



Squore 17.1.11

Reference Manual

Reference : REF_Squore
Version : 17.1.11
Date : 27/09/2019

Reference Manual

Copyright © 2019 Squoring Technologies

Abstract

This edition of the Reference Manual applies to Squore 17.1.11 and to all subsequent releases and modifications until otherwise indicated in new editions.

Licence

No part of this publication may be reproduced, transmitted, stored in a retrieval system, nor translated into any human or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual or otherwise, without the prior written permission of the copyright owner, Squoring Technologies.

Squoring Technologies reserves the right to revise this publication and to make changes from time to time without obligation to notify authorised users of such changes. Consult Squoring Technologies to determine whether any such changes have been made.

The terms and conditions governing the licensing of Squoring Technologies software consist solely of those set forth in the written contracts between Squoring Technologies and its customers.

All third-party products are trademarks or registered trademarks of their respective companies.

Warranty

Squoring Technologies makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Squoring Technologies shall not be liable for errors contained herein nor for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Table of Contents

1. Introduction	1
1.1. Foreword	1
1.2. About This Document	1
1.3. Contacting Squoring Technologies Product Support	1
1.4. Responsibilities	1
1.5. Getting the Latest Version of this Manual	2
2. Coding Standards	3
2.1. ABAP	3
2.1.1. ABAP Metrics	3
2.1.2. ABAP Ruleset	8
2.2. ADA	13
2.2.1. ADA Metrics	13
2.2.2. ADA Ruleset	21
2.3. C	23
2.3.1. C Metrics	23
2.3.2. C Ruleset	31
2.4. COBOL	35
2.4.1. COBOL Metrics	35
2.4.2. COBOL Ruleset	41
2.5. C++	48
2.5.1. C++ Metrics	48
2.5.2. C++ Ruleset	57
2.6. C#	60
2.6.1. C# Metrics	60
2.6.2. C# Ruleset	69
2.7. Fortran	72
2.7.1. Fortran Metrics	72
2.7.2. Fortran Ruleset	77
2.8. Java	80
2.8.1. Java Metrics	80
2.8.2. Java Ruleset	87
2.9. Javascript	89
2.9.1. Javascript Metrics	89
2.9.2. Javascript Ruleset	94
2.10. MindC	97
2.10.1. MindC Metrics	97
2.10.2. MindC Ruleset	105
2.11. Objective-C	109
2.11.1. Objective-C Metrics	109
2.11.2. Objective-C Ruleset	117
2.12. PHP	120
2.12.1. PHP Metrics	120
2.12.2. PHP Ruleset	127
2.13. Python	130
2.13.1. Python Metrics	130
2.13.2. Python Ruleset	135
2.14. PL/SQL	138
2.14.1. PL/SQL Metrics	138
2.14.2. PL/SQL Ruleset	144
2.15. TSQL	146

2.15.1. TSQL Metrics	146
2.15.2. TSQL Ruleset	150
2.16. VB.net	152
2.16.1. VB.net Metrics	152
2.16.2. VB.net Ruleset	161
2.17. Xaml	164
2.17.1. Xaml Metrics	164
2.17.2. Xaml Ruleset	169
3. Repository Connectors	173
3.1. Folder Path	173
3.1.1. Description	173
3.1.2. Usage	173
3.2. Zip Upload	173
3.2.1. Description	173
3.2.2. Usage	173
3.3. CVS	173
3.3.1. Description	173
3.3.2. Usage	174
3.4. ClearCase	174
3.4.1. Description	174
3.4.2. Usage	174
3.5. Perforce	175
3.5.1. Description	175
3.5.2. Usage	175
3.6. Git	176
3.6.1. Description	176
3.6.2. Usage	176
3.7. PTC Integrity	176
3.7.1. Description	177
3.7.2. Usage	177
3.8. TFS	177
3.8.1. Description	177
3.8.2. Usage	178
3.9. Synergy	178
3.9.1. Description	178
3.9.2. Usage	179
3.10. SVN	179
3.10.1. Description	179
3.10.2. Usage	180
3.11. Using Multiple Nodes	180
4. Data Providers	181
4.1. AntiC	181
4.1.1. Description	181
4.1.2. Usage	181
4.2. Automotive Coverage Import	181
4.2.1. Description	181
4.2.2. Usage	181
4.3. Automotive Tag Import	181
4.3.1. Description	181
4.3.2. Usage	182
4.4. BullseyeCoverage Code Coverage Analyzer	182
4.4.1. Description	182
4.4.2. Usage	182

4.5. CPD	182
4.5.1. Description	182
4.5.2. Usage	182
4.6. Cppcheck	182
4.6.1. Description	183
4.6.2. Usage	183
4.7. Cppcheck (plugin)	183
4.7.1. Description	183
4.7.2. Usage	183
4.8. CPPTest	183
4.8.1. Description	183
4.8.2. Usage	184
4.9. Cantata	184
4.9.1. Description	184
4.9.2. Usage	184
4.10. CheckStyle	184
4.10.1. Description	184
4.10.2. Usage	184
4.11. CheckStyle (plugin)	184
4.11.1. Description	185
4.11.2. Usage	185
4.12. CheckStyle for SQALE (plugin)	185
4.12.1. Description	185
4.12.2. Usage	185
4.13. Cobertura	186
4.13.1. Description	186
4.13.2. Usage	186
4.14. CodeSonar	186
4.14.1. Description	186
4.14.2. Usage	186
4.15. Compiler	186
4.15.1. Description	186
4.15.2. Usage	187
4.16. Coverity	187
4.16.1. Description	187
4.16.2. Usage	187
4.17. FindBugs	187
4.17.1. Description	187
4.17.2. Usage	187
4.18. FindBugs (plugin)	187
4.18.1. Description	188
4.18.2. Usage	188
4.19. Function Relaxer	188
4.19.1. Description	188
4.19.2. Usage	188
4.20. FxCop	188
4.20.1. Description	188
4.20.2. Usage	189
4.21. GCov	189
4.21.1. Description	189
4.21.2. Usage	189
4.22. GNATcheck	189
4.22.1. Description	189

4.22.2. Usage	189
4.23. GNATCompiler	190
4.23.1. Description	190
4.23.2. Usage	190
4.24. JUnit	190
4.24.1. Description	190
4.24.2. Usage	190
4.25. JaCoCo	190
4.25.1. Description	190
4.25.2. Usage	191
4.26. Klocwork	191
4.26.1. Description	191
4.26.2. Usage	191
4.27. Rational Logiscope	191
4.27.1. Description	191
4.27.2. Usage	191
4.28. MemUsage	191
4.28.1. Description	192
4.28.2. Usage	192
4.29. NCover	192
4.29.1. Description	192
4.29.2. Usage	192
4.30. Oracle PLSQL compiler Warning checker	192
4.30.1. Description	192
4.30.2. Usage	192
4.31. MISRA Rule Checking using PC-lint	193
4.31.1. Description	193
4.31.2. Usage	193
4.32. PMD	193
4.32.1. Description	193
4.32.2. Usage	193
4.33. PMD (plugin)	193
4.33.1. Description	194
4.33.2. Usage	194
4.34. Polyspace	194
4.34.1. Description	194
4.34.2. Usage	194
4.35. Polyspace MISRA	194
4.35.1. Description	194
4.35.2. Usage	195
4.36. Polyspace (plugin)	195
4.36.1. Description	195
4.36.2. Usage	195
4.37. MISRA Rule Checking with QAC	195
4.37.1. Description	196
4.37.2. Usage	196
4.38. Unit Test Code Coverage from Rational Test RealTime	196
4.38.1. Description	196
4.38.2. Usage	196
4.39. ReqIF	196
4.39.1. Description	196
4.39.2. Usage	197
4.40. SQL Code Guard	197

4.40.1. Description	197
4.40.2. Usage	197
4.41. Squan Sources	197
4.41.1. Description	197
4.41.2. Usage	198
4.42. Squore Import	199
4.42.1. Description	200
4.42.2. Usage	200
4.43. Squore Virtual Project	200
4.43.1. Description	200
4.43.2. Usage	200
4.44. StyleCop	200
4.44.1. Description	200
4.44.2. Usage	200
4.45. StyleCop (plugin)	201
4.45.1. Description	201
4.45.2. Usage	201
4.46. Tessy	201
4.46.1. Description	201
4.46.2. Usage	201
4.47. VectorCAST	201
4.47.1. Description	202
4.47.2. Usage	202
4.48. CodeSniffer	202
4.48.1. Description	202
4.48.2. Usage	202
4.49. Configuration Checker	202
4.49.1. Description	202
4.49.2. Usage	202
4.50. Csv Coverage Import	203
4.50.1. Description	203
4.50.2. Usage	203
4.51. CSV Findings	203
4.51.1. Description	203
4.51.2. Usage	203
4.52. Csv Tag Import	203
4.52.1. Description	203
4.52.2. Usage	203
4.53. Csv Test Results Import	204
4.53.1. Description	204
4.53.2. Usage	204
4.54. OSLC	204
4.54.1. Description	204
4.54.2. Usage	204
4.55. pep8	204
4.55.1. Description	205
4.55.2. Usage	205
4.56. pycodestyle / pep8 (plugin)	205
4.56.1. Description	205
4.56.2. Usage	205
4.57. PHP Code Coverage	205
4.57.1. Description	205
4.57.2. Usage	205

4.58. pylint	206
4.58.1. Description	206
4.58.2. Usage	206
4.59. pylint (plugin)	206
4.59.1. Description	206
4.59.2. Usage	206
4.60. Qac_8_2	206
4.60.1. Description	206
4.60.2. Usage	207
4.61. Advanced COBOL Parsing	207
4.62. Using Data Provider Input Files From Version Control	207
4.63. Providing a catalog file to a Data Provider for Offline XSL Transformations	208
4.64. Creating your own Data Providers	209
4.64.1. Choosing the Right Data Provider Framework	210
4.64.2. Extending a Framework	211
4.64.3. Creating a Freestyle Data Provider	212
4.64.4. Data Provider Parameters	213
4.64.5. Localising your Data Provider	215
5. Cloning Detection	218
5.1. Cloning Metrics	218
5.1.1. CC - Code Cloned	218
5.1.2. CFTC - Control Flow Token (CFT) Cloned	218
5.1.3. CAC - Children Artefact Cloned	218
5.1.4. CN - Clones Number	219
5.1.5. RS - Repeated Substrings (Repeated Code Blocks)	219
5.1.6. CFTRS - Repeated Substrings in Control Flow Token	219
5.1.7. ICC - Inner Code Cloned	219
5.1.8. ICFTC - Inner Control Flow Token Cloned	219
5.2. Cloning Violations	220
5.2.1. CC (R_NOCC)	220
5.2.2. CFTC (R_NOCFTC)	220
5.2.3. CAC (R_NOCAC)	220
5.2.4. RS (R_NORS)	220
5.2.5. CFTRS (R_NOCFTRS)	220
6. Glossary	221
6.1. Acceptance Testing	221
6.2. Accessibility	221
6.3. Accuracy	221
6.4. Accuracy of Measurement	222
6.5. Acquirer	222
6.6. Action	223
6.7. Activity	223
6.8. Actor	224
6.9. Adaptability	224
6.10. Agreement	225
6.11. Analysability	225
6.12. Analysis Model	225
6.13. Architecture	225
6.14. Attractiveness	226
6.15. Attribute	226
6.16. Availability	227
6.17. Base Measure	227
6.18. Baseline	228

6.19. Branch	228
6.20. Branch Coverage	229
6.21. Branch Testing	229
6.22. Budget	230
6.23. Build	230
6.24. Call Graph	230
6.25. Capability Maturity Model	231
6.26. Certification	231
6.27. Certification Criteria	231
6.28. Change Control Board	232
6.29. Change Control System	232
6.30. Change Management	232
6.31. Changeability	233
6.32. Co-existence	233
6.33. Code	233
6.34. Code Coverage	234
6.35. Code Freeze	234
6.36. Code Review	234
6.37. Code Verification	234
6.38. Coding	235
6.39. Cohesion	235
6.40. Commercial-Off-The-Shelf (COTS)	236
6.41. Commit	236
6.42. Commitment	236
6.43. Compatibility	237
6.44. Complexity	237
6.45. Component	238
6.46. Conciseness	238
6.47. Condition	239
6.48. Configuration	239
6.49. Configuration Control	239
6.50. Configuration Item	240
6.51. Configuration Management	241
6.52. Configuration Management System	241
6.53. Conflict	242
6.54. Conformance	242
6.55. Connectivity	243
6.56. Consistency	243
6.57. Constraint	243
6.58. Content Coupling	244
6.59. Context of Use	244
6.60. Contract	244
6.61. Control Coupling	245
6.62. Control Flow	245
6.63. Control Flow Diagram	245
6.64. Convention	246
6.65. Correctability	246
6.66. Correctness	246
6.67. Coupling	247
6.68. Coverage	247
6.69. Criteria	248
6.70. Criticality	248
6.71. Custom Software	248

6.72. Customer	249
6.73. Data	249
6.74. Data Coupling	250
6.75. Data Flow	250
6.76. Data Flow Diagram	251
6.77. Data Management	251
6.78. Data Model	251
6.79. Data Processing	252
6.80. Data Provider	252
6.81. Data Store	253
6.82. Data Type	253
6.83. Database	253
6.84. Decision Criteria	254
6.85. Decoupling	254
6.86. Defect	254
6.87. Degree of Confidence	255
6.88. Deliverable	255
6.89. Delivery	255
6.90. Dependability	256
6.91. Deployment	256
6.92. Derived Measure	256
6.93. Design	257
6.94. Design Pattern	257
6.95. Developer	258
6.96. Development	258
6.97. Development Testing	259
6.98. Direct Measure	259
6.99. Direct Metric	260
6.100. Document	260
6.101. Documentation	261
6.102. Dynamic Analysis	262
6.103. Earned Value	262
6.104. Effectiveness	262
6.105. Efficiency	262
6.106. Efficiency Compliance	263
6.107. Effort	263
6.108. Encapsulation	263
6.109. End User	264
6.110. Entity	264
6.111. Entry Point	265
6.112. Environment	265
6.113. Error	266
6.114. Error Tolerance	266
6.115. Evaluation	266
6.116. Evaluation Activity	267
6.117. Evaluation Group	267
6.118. Evaluation Method	268
6.119. Evaluation Module	268
6.120. Evaluation Technology	269
6.121. Evaluation Tool	269
6.122. Execute	269
6.123. Execution Efficiency	270
6.124. Execution Time	270

6.125. Exit	270
6.126. Expandability	270
6.127. Extendability	271
6.128. External Attribute	271
6.129. External Measure	271
6.130. External Quality	272
6.131. External Software Quality	272
6.132. Facility	273
6.133. Failure	273
6.134. Failure Rate	274
6.135. Fault	275
6.136. Fault Tolerance	275
6.137. Feasibility	276
6.138. Feature	276
6.139. Feature Freeze	276
6.140. Finite State Machine	277
6.141. Flexibility	277
6.142. Frozen Branch	277
6.143. Function	277
6.144. Functional Analysis	278
6.145. Functional Requirement	278
6.146. Functional Size	279
6.147. Functional Testing	279
6.148. Functional Unit	279
6.149. Functionality	279
6.150. Functionality Compliance	280
6.151. Generality	280
6.152. Generic Practice	280
6.153. Glossary	281
6.154. Goal	281
6.155. Granularity	281
6.156. Historical Information	281
6.157. Hybrid Coupling	281
6.158. Impact Analysis	282
6.159. Implementation	282
6.160. Implied Needs	283
6.161. Incremental Development	283
6.162. Indicator	283
6.163. Indicator Value	284
6.164. Indirect Measure	284
6.165. Indirect Metric	285
6.166. Information	285
6.167. Information Analysis	285
6.168. Information Management	285
6.169. Information Need	286
6.170. Information Product	286
6.171. Inspection	286
6.172. Installability	287
6.173. Installation Manual	287
6.174. Integration	287
6.175. Integration Test	288
6.176. Integrity	288
6.177. Interface Testing	288

6.178. Intermediate Software Product	288
6.179. Internal Attribute	289
6.180. Internal Measure	289
6.181. Internal Quality	290
6.182. Internal Software Quality	290
6.183. Interoperability	291
6.184. Interoperability Testing	291
6.185. Interval Scale	291
6.186. Item	292
6.187. Iteration	292
6.188. Key Practices	292
6.189. Key Process Area	293
6.190. Knowledge Base	293
6.191. Learnability	293
6.192. Lessons Learned	294
6.193. Level of Performance	294
6.194. Life Cycle	294
6.195. Life Cycle Model	295
6.196. Maintainability	295
6.197. Maintainability Compliance	296
6.198. Maintainer	296
6.199. Maintenance	296
6.200. Maintenance Manual	297
6.201. Maturity	298
6.202. Measurable Concept	298
6.203. Measurand	298
6.204. Measure	298
6.205. Measurement	299
6.206. Measurement Analyst	300
6.207. Measurement Experience Base	300
6.208. Measurement Function	300
6.209. Measurement Method	301
6.210. Measurement Procedure	301
6.211. Measurement Process	302
6.212. Measurement Process Owner	302
6.213. Measurement Sponsor	302
6.214. Measurement User	302
6.215. Metric	303
6.216. Milestone	303
6.217. Mock Object	304
6.218. Model	304
6.219. Modifiability	304
6.220. Modifiable	305
6.221. Modularity	305
6.222. Module	305
6.223. Moke Object	306
6.224. Multidimensional Analysis	306
6.225. Network	306
6.226. Nonfunctional Requirement	307
6.227. Nontechnical Requirement	307
6.228. Object	307
6.229. Object Model	308
6.230. Object Oriented Design	308

6.231. Observation	308
6.232. Observation Period	309
6.233. Operability	309
6.234. Operand	309
6.235. Operational Testing	310
6.236. Operator	310
6.237. Operator Manual	311
6.238. Optional Attribute	311
6.239. Optional Requirement	311
6.240. Organisational Unit	312
6.241. Path	312
6.242. Path Analysis	312
6.243. Path Testing	313
6.244. Pathological Coupling	313
6.245. Peer Review	313
6.246. Performance	314
6.247. Performance Indicator	314
6.248. Performance Testing	315
6.249. Pilot Project	315
6.250. Portability	315
6.251. Portability Compliance	315
6.252. Practice	316
6.253. Precision	316
6.254. Predictive Metric	317
6.255. Procedure	317
6.256. Process	317
6.257. Process Assessment	318
6.258. Process Assessment Model	318
6.259. Process Capability	319
6.260. Process Capability Determination	319
6.261. Process Capability Level	319
6.262. Process Context	319
6.263. Process Improvement	320
6.264. Process Improvement Objective	320
6.265. Process Improvement Program	320
6.266. Process Improvement Project	321
6.267. Process Metric	321
6.268. Process Outcome	321
6.269. Process Performance	322
6.270. Process Purpose	322
6.271. Product	323
6.272. Product Line	324
6.273. Product Metric	324
6.274. Productivity	324
6.275. Programmer Manual	324
6.276. Project	325
6.277. Project Management	325
6.278. Project Phase	326
6.279. Prototype	326
6.280. Qualification	327
6.281. Qualification Testing	327
6.282. Quality	327
6.283. Quality Assurance	328

6.284. Quality Control	329
6.285. Quality Evaluation	329
6.286. Quality Factor	330
6.287. Quality Management	330
6.288. Quality Measure Element	330
6.289. Quality Metric	331
6.290. Quality Model	331
6.291. Quality in Use	331
6.292. Rating	332
6.293. Rating Level	333
6.294. Readability	333
6.295. Recoverability	333
6.296. Recovery	334
6.297. Reengineering	334
6.298. Regression Testing	334
6.299. Release	335
6.300. Reliability	335
6.301. Reliability Compliance	336
6.302. Repeatability of Results of Measurements	336
6.303. Replaceability	337
6.304. Reproducibility of Results of Measurements	337
6.305. Request For Change	338
6.306. Request For Information	338
6.307. Request For Proposal	338
6.308. Requirement	338
6.309. Requirements Analysis	339
6.310. Requirements Derivation	340
6.311. Requirements Document	340
6.312. Requirements Engineering	340
6.313. Requirements Partitioning	341
6.314. Requirements Review	341
6.315. Requirements Specification	341
6.316. Requirements Traceability	342
6.317. Requirements Traceability Matrix	342
6.318. Resource	343
6.319. Resource Utilisation	343
6.320. Result	343
6.321. Retirement	344
6.322. Reverse Engineering	344
6.323. Risk	345
6.324. Risk Acceptance	345
6.325. Risk Analysis	346
6.326. Robustness	346
6.327. Role	346
6.328. Routine	347
6.329. Run	347
6.330. Safety	348
6.331. Satisfaction	348
6.332. Scale	348
6.333. Security	349
6.334. Service	350
6.335. Service Level Agreement	350
6.336. Simplicity	350

6.337. Software	351
6.338. Software Asset Management	351
6.339. Software Development Process	352
6.340. Software Engineering	352
6.341. Software Item	352
6.342. Software Life Cycle	353
6.343. Software Product Evaluation	353
6.344. Software Quality	354
6.345. Software Quality Characteristic	354
6.346. Software Quality Evaluation	354
6.347. Software Quality Measure	355
6.348. Software Repository	355
6.349. Software Unit	355
6.350. Source Code	356
6.351. Specification	356
6.352. Stability	356
6.353. Stage	357
6.354. Stakeholder	357
6.355. Standard	358
6.356. Standard Process	358
6.357. Statement	358
6.358. Statement Testing	359
6.359. Statement of Work	359
6.360. Static Analysis	359
6.361. Statistical Process Control	360
6.362. Step	361
6.363. Stress Testing	361
6.364. Structural Testing	361
6.365. Stub	362
6.366. Suitability	362
6.367. Supplier	363
6.368. Support	363
6.369. Support Manual	364
6.370. System	364
6.371. System Testing	364
6.372. Task	365
6.373. Technical Requirement	365
6.374. Technique	366
6.375. Test	366
6.376. Test Case	367
6.377. Test Case Suite	367
6.378. Test Coverage	367
6.379. Test Documentation	368
6.380. Test Environment	368
6.381. Test Objective	369
6.382. Test Plan	369
6.383. Test Procedure	369
6.384. Testability	370
6.385. Testing	370
6.386. Testing Description	371
6.387. Time Behaviour	371
6.388. Tool	372
6.389. Total Quality Management	372

6.390. Traceability	372
6.391. Traceable	373
6.392. Trunk	373
6.393. Understandability	373
6.394. Unit Test	374
6.395. Unit of Measurement	374
6.396. Usability	375
6.397. Usability Compliance	375
6.398. User	375
6.399. User Documentation	376
6.400. User Manual	377
6.401. Validation	377
6.402. Value	379
6.403. Verification	379
6.404. Version	380
6.405. Work Breakdown Structure	380
6.406. Work Product	381
7. Standards	382
7.1. CMMi	382
7.2. DOD-STD-2167A	382
7.3. IEC 61508	383
7.4. IEC 61508-3	383
7.5. IEC 61508-7	383
7.6. IEEE 1012	384
7.7. IEEE 1058	384
7.8. IEEE 1061	384
7.9. IEEE 1074	384
7.10. IEEE 1220	385
7.11. IEEE 1233	385
7.12. IEEE 1320	385
7.13. IEEE 1362	385
7.14. IEEE 1490	386
7.15. IEEE 610.12	386
7.16. IEEE 829	386
7.17. IEEE 830	386
7.18. IEEE 982	387
7.19. ISO 5806	387
7.20. ISO 8402	387
7.21. ISO 9001	387
7.22. ISO 9127	387
7.23. ISO 9241	388
7.24. ISO 9241-10	389
7.25. ISO 9241-11	390
7.26. ISO/IEC 12119	391
7.27. ISO/IEC 12207	391
7.28. ISO/IEC 14143	391
7.29. ISO/IEC 14143-1	392
7.30. ISO/IEC 14143-3	392
7.31. ISO/IEC 14598	393
7.32. ISO/IEC 14598-1	393
7.33. ISO/IEC 14598-2	394
7.34. ISO/IEC 14598-3	394
7.35. ISO/IEC 14598-4	395

7.36. ISO/IEC 14598-5	395
7.37. ISO/IEC 14598-6	396
7.38. ISO/IEC 14756	396
7.39. ISO/IEC 14764	396
7.40. ISO/IEC 15026	397
7.41. ISO/IEC 15026-1	397
7.42. ISO/IEC 15026-2	397
7.43. ISO/IEC 15288	398
7.44. ISO/IEC 15289	398
7.45. ISO/IEC 15414	399
7.46. ISO/IEC 15474	399
7.47. ISO/IEC 15474-1	399
7.48. ISO/IEC 15474-2	400
7.49. ISO/IEC 15504	400
7.50. ISO/IEC 15504-1	401
7.51. ISO/IEC 15504-2	401
7.52. ISO/IEC 15504-3	402
7.53. ISO/IEC 15504-4	402
7.54. ISO/IEC 15504-5	403
7.55. ISO/IEC 15504-6	404
7.56. ISO/IEC 15504-7	404
7.57. ISO/IEC 15846	405
7.58. ISO/IEC 15910	405
7.59. ISO/IEC 15939	406
7.60. ISO/IEC 19759	406
7.61. ISO/IEC 19770	406
7.62. ISO/IEC 19770-1	407
7.63. ISO/IEC 19770-2	407
7.64. ISO/IEC 20000	408
7.65. ISO/IEC 2382	408
7.66. ISO/IEC 2382-1	409
7.67. ISO/IEC 25000	410
7.68. ISO/IEC 25001	410
7.69. ISO/IEC 25010	411
7.70. ISO/IEC 25012	411
7.71. ISO/IEC 25020	411
7.72. ISO/IEC 25021	412
7.73. ISO/IEC 25030	412
7.74. ISO/IEC 25040	413
7.75. ISO/IEC 25045	413
7.76. ISO/IEC 25051	413
7.77. ISO/IEC 25060	414
7.78. ISO/IEC 25062	414
7.79. ISO/IEC 26514	415
7.80. ISO/IEC 29881	415
7.81. ISO/IEC 90003	415
7.82. ISO/IEC 9126	415
7.83. ISO/IEC 9126-1	416
7.84. ISO/IEC 9126-2	417
7.85. ISO/IEC 9126-3	417
7.86. ISO/IEC 9126-4	418
7.87. ISO/IEC 9294	418
7.88. ISO/IEC 99	419

7.89. ISO/IEC SQuaRE	419
7.90. ISO/IEC/IEEE 15289	420
7.91. ISO/IEC/IEEE 24765	421
7.92. RTCA/EUROCAE	421
7.93. SIGIST	421
7.94. Team Software Process	421
A. Data Provider Frameworks	423
Csv Reference	423
csv_findings Reference	426
CsvPerl Reference	427
Generic Reference	429
GenericPerl Reference	433
FindingsPerl Reference	436
ExcelMetrics Reference	440
B. Squore XML Schemas	446
input-data-2.xsd	446
properties-1.2.xsd	448
config-1.2.xsd	449
analysis.xsd	451
decision.xsd	456
description.xsd	457
exports.xsd	458
highlights.xsd	459
properties.xsd	463
tutorials.xsd	464
wizards.xsd	470
C. Licences	475
Software Licence Agreement	475
Redistributed Software	481
Licences	482
Index	548

1. Introduction

1.1. Foreword

This document was released by Squoring Technologies.

It is part of the user documentation of the Squore software product edited and distributed by Squoring Technologies.

1.2. About This Document

The Reference Manual provides a complete reference for the metrics, glossary and standards used in Squore 17.1.11.

This manual is intended for Squore administrators and end-users. It gives useful information about the technical background of Squore and important knowledge basis to understand what is measured and how.

If you are already familiar with Squore, you can navigate this manual by looking for what has changed since the previous version. New functionality is tagged with **(new in 17.1)** throughout this manual. A summary of the new features described in this manual is available in the entry * **What's New in Squore 17.1?** of this manual's Index.

For information on how to use and configure Squore, the full suite of manuals includes:

- Squore Installation Checklist
- Squore Installation and Administration Guide
- Squore Getting Started Guide
- Squore Command Line Interface
- Squore Configuration Guide
- Squore Eclipse Plugin Guide
- Squore Reference Manual


1.3. Contacting Squoring Technologies Product Support

If the information provided in this manual is erroneous or inaccurate, or if you encounter problems during your installation, contact Squoring Technologies Product Support: <https://support.squoring.com/>

You will need a valid Squore customer account to submit a support request. You can create an account on the support website if you do not have one already.

For any communication:

 support@squoring.com

 **Squoring Technologies Product Support**
76, allées Jean Jaurès / 31000 Toulouse - FRANCE

1.4. Responsibilities

Approval of this version of the document and any further updates are the responsibility of Squoring Technologies.

1.5. Getting the Latest Version of this Manual

The version of this manual included in your Squore installation may have been updated. If you would like to check for updated user guides, consult the Squoring Technologies documentation site to consult or download the latest Squore manuals at <https://support.squoring.com/documentation/17.1.11>. Manuals are constantly updated and published as soon as they are available.

2. Coding Standards

This chapter describes the list of metrics and rules for each language supported by . Note that this is not the complete list of metrics and rules in , only the ones generated by our source code parser. Some of the rules may also be disabled by default in your configuration. For more information about your analysis model, consult 's and , which provide more information about each metric and rule.

2.1. ABAP

2.1.1. ABAP Metrics

Andthen Operators

- **Mnemonic** ANTH
- **Description** Number of 'andthen' operators

Number of comment blocks

- **Mnemonic** BCOM
- **Description** Number of comment blocks.

Header Blocks Of Comment

- **Mnemonic** BHCO
- **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

- **Mnemonic** BLAN
- **Description** Number of blank lines of code in the source file(s).

Brace Lines

- **Mnemonic** BRAC
- **Description** Number of lines of code containing only a brace in the source file(s).

Case Blocks

- **Mnemonic** CABL
- **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

- **Mnemonic** CASE
- **Description** Number of 'case' labels in the function

Catch Statements

- **Mnemonic** CATC
- **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

- **Mnemonic** CCN

→ **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

→ **Mnemonic** CFT

→ **Description** Number of tokens in the control flow of functions

Call Graph Depth

→ **Mnemonic** CGDM

→ **Description** Maximum depth of the call graph.

Number of Check instruction

→ **Mnemonic** CHECK

→ **Description** Number of Check instruction

Comment Lines

→ **Mnemonic** CLOC

→ **Description** Number of lines of comments in the source file(s).

Continue Statements

→ **Mnemonic** CONT

→ **Description** Number of 'continue' statements in the function

Public Constant

→ **Mnemonic** CPBL

→ **Description** Public Constant

Protected Constant

→ **Mnemonic** CPRT

→ **Description** Protected Constant

Private Constant

→ **Mnemonic** CPRV

→ **Description** Private Constant

Commented Statements

→ **Mnemonic** CSTAT

→ **Description** Number of Commented Statements.

Minimum Number of Cycles

→ **Mnemonic** CYCL

→ **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Default Statement

→ **Mnemonic** DEFT

→ **Description** Number of 'default' blocks in 'switch' in the function

Distinct Operands

→ **Mnemonic** DOPD

→ **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Public Data

→ **Mnemonic** DPBL

→ **Description** Public Data

Protected Data

→ **Mnemonic** DPRT

→ **Description** Protected Data

Private Data

→ **Mnemonic** DPRV

→ **Description** Private Data

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

Call to exit

→ **Mnemonic** EXIT

→ **Description** Number of calls to the exit function

For Statements

→ **Mnemonic** FOR

→ **Description** Number of 'for' statements in the function

File Type Count

→ **Mnemonic** FTYP

→ **Description** File Type Count

Structures Added

→ **Mnemonic** SADD

→ **Description** Number of control structures added since the previous version.

Structures Modified

→ **Mnemonic** SMOD

→ **Description** Number of control structures modified since the previous version.

Structures Removed

→ **Mnemonic** SREM

→ **Description** Number of control structures removed since the previous version.

Number of Structures

→ **Mnemonic** SSIZ

→ **Description** Number of control structures: iterations, selections, sequences

Header Lines Of Comment

→ **Mnemonic** HCOM

→ **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

→ **Mnemonic** HLOC

→ **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

→ **Mnemonic** ICC

→ **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

→ **Mnemonic** ICFTC

→ **Description** Number of duplicated tokens in control flow of functions

If Statements

→ **Mnemonic** IF

→ **Description** Number of 'if' statements

Line Count

→ **Mnemonic** LC

→ **Description** Number of lines.

Loop Statements

→ **Mnemonic** LOOP

→ **Description** Number of loop statements in the function

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Non-Cyclic Paths

→ **Mnemonic** PATH

→ **Description** Number of non-cyclic paths in the function.

Orelse operators

→ **Mnemonic** OREL

→ **Description** Number of 'orelse' operators

Return Statements

→ **Mnemonic** RETURN

→ **Description** Number of 'return' statements in the function

Repeated Code Blocks

→ **Mnemonic** RS

→ **Description** Duplicated blocks in the function

Skipped Lines of Comment code

→ **Mnemonic** SKLC

→ **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

→ **Mnemonic** SLOC

→ **Description** Number of lines of source code in the source file(s).

Executable Statements

→ **Mnemonic** STAT

→ **Description** Total number of executable statements.

Switch Statements

→ **Mnemonic** SWIT

→ **Description** Number of 'switch' statements in the function

Throw Statements

→ **Mnemonic** THRO

→ **Description** Number of 'throw' statements in the function

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

→ **Mnemonic** TRY

→ **Description** Number of 'try' statements in the function

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

While Statements

→ **Mnemonic** WHIL

→ **Description** Number of 'while' statements in the function

2.1.2. ABAP Ruleset

Avoid using APPEND statements in loops

→ **Mnemonic** AVOIDAPPENDINLOOP

→ **Description** Avoid using COMMIT WORK statements in loops.

Avoid using APPEND in SQL SELECT statements

→ **Mnemonic** AVOIDAPPENDINSELECT

→ **Description** Avoid using APPEND in SQL SELECT statements.

Avoid using BREAK-POINT

→ **Mnemonic** AVOIDBREAKPOINT

→ **Description** Avoid using BREAK-POINT

Avoid using CHECK in SQL SELECT statements

→ **Mnemonic** AVOIDCHECKINSELECT

→ **Description** Avoid using CHECK in SQL SELECT statements

Avoid using COMMIT WORK statements in loops

→ **Mnemonic** AVOIDCOMMITWORKINLOOP

→ **Description** Avoid using COMMIT WORK statements in loops.

Avoid obsolete DATA BEGIN OF OCCURS statement

- **Mnemonic** AVOIDDATAOCCURS
- **Description** Avoid obsolete DATA BEGIN OF OCCURS statement

Avoid using INSERT statements in loops

- **Mnemonic** AVOIDINSERTINLOOP
- **Description** Avoid using INSERT statements in loops. Querying in a loop can lead to performance issues.

Avoid using INSERT in SQL SELECT statements

- **Mnemonic** AVOIDINSERTINSELECT
- **Description** Avoid using INSERT in SQL SELECT statements

Avoid using SQL INTO statements in loops

- **Mnemonic** AVOIDINTOINLOOP
- **Description** Avoid using SQL INTO statements in loops.

Avoid using SELECT *

- **Mnemonic** AVOIDSELECTALL
- **Description** SELECT * should be avoided as it does not enable to keep control on the flow return and could therefore be error prone and potentially lead to performance issues.

Avoid using the SQL "BYPASSING BUFFER" clause

- **Mnemonic** AVOIDSELECTBYPASS
- **Description** The BYPASSING BUFFER clause causes the SELECT statement to avoid the SAP buffering and to read directly from the database and not from the buffer on the application server.

Avoid using SELECT DISTINCT Statement

- **Mnemonic** AVOIDSELECTDISTINCT
- **Description** The SQL DISTINCT clause causes the SELECT statement to avoid the SAP buffering and to read directly from the database and not from the buffer on the application server.

Avoid SELECT SQL statement without a WHERE clause

- **Mnemonic** AVOIDSELECTNOWHERE
- **Description** Avoid SELECT SQL statement without a WHERE clause

Avoid SELECT SQL statement with a WHERE clause containing the NOT EQUAL operator

- **Mnemonic** AVOIDSELECTWHERENOTEQ
- **Description** Avoid SELECT SQL statement with a WHERE clause containing the NOT EQUAL operator.

Avoid using SQL Aggregate Functions

- **Mnemonic** AVOIDSQLAGGREGATEFUNC
- **Description** SQL COUNT(..) , MIN(..) , MAX(..) , SUM(..) , AVG(..) functions cause the SAP table buffer to be bypassed and so usage of such functions can lead to some performance issues.

Avoid using SUBMIT statements in loops

→ **Mnemonic** AVOIDSUBMITINLOOP

→ **Description** Avoid using SUBMIT statements in loops.

Avoid using UPDATE, MODIFY, DELETE statements in loops

→ **Mnemonic** AVOIDUPDELINLOOP

→ **Description** Avoid using UPDATE, MODIFY, DELETE statements in loops. Querying in a loop can lead to performance issues.

Avoid UPDATE or DELETE SQL Statement without a WHERE clause

→ **Mnemonic** AVOIDUPDELNOWHERE

→ **Description** Avoid UPDATE or DELETE SQL Statement without a WHERE clause

Avoid using GROUP BY in queries

→ **Mnemonic** AVOIDUSINGSQLGROUPBY

→ **Description** Using GROUP BY in SQL queries can lead to performance issues.

Avoid using the JOIN SQL clause

→ **Mnemonic** AVOIDUSINGSQLJOIN

→ **Description** Using the SQL JOIN clause leads to bypass the SAP table buffer.

Avoid using LIKE in SQL queries

→ **Mnemonic** AVOIDUSINGSQLLIKE

→ **Description** Using LIKE in SQL queries can lead to performance issues.

Avoid using the WAIT statement

→ **Mnemonic** AVOIDWAIT

→ **Description** Avoid using the WAIT statement.

The class name should conform to the defined standard

→ **Mnemonic** CLASSNAMINGCONVENTION

→ **Description** The class name should conform to the defined standard

Commented-out Source Code is not allowed

→ **Mnemonic** R_CSTAT

→ **Description** Commented-out Source Code is not allowed

Missing Default

→ **Mnemonic** DEFAULT

→ **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Missing final else

→ **Mnemonic** ELSEFINAL

→ **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Forbid call to a system function

→ **Mnemonic** FORBIDCALLCFUNC

→ **Description** Call of a System Function: CALL 'cfunc' is only intended for internal usage. Incompatible changes and further development is possible at any time and without warning or notice.

Forbid calls to dialog transactions

→ **Mnemonic** FORBIDCALLDIALTRANS

→ **Description** Forbid calls to dialog transactions.

Forbid use of GENERATE REPORT / SUBROUTINE POOL / DYNPRO

→ **Mnemonic** FORBIDGENERATEPROG

→ **Description** This statement is exclusively for internal use within SAP Technology Development. Incompatible changes or developments are possible at any time without prior warning or notes.

Forbid calls to GET RUN TIME.

→ **Mnemonic** FORBIDGETRUNTIME

→ **Description** Forbid calls to GET RUN TIME.

Forbid use of INSERT/DELETE REPORT/TEXTPOOL

→ **Mnemonic** FORBIDINSERTPROG

→ **Description** This statement is exclusively for internal use within SAP Technology Development. Incompatible changes or developments are possible at any time without prior warning or notice.

Forbid uses of OFFSET in ASSIGN

→ **Mnemonic** FORBIDOFFSETINASSIGN

→ **Description** Forbid uses of OFFSET in ASSIGN.

Forbid use of SYSTEM-CALL

→ **Mnemonic** FORBIDSYSTEMCALL

→ **Description** This statement is only for !Internal use in SAP Basis development!. Its use is subject to various restrictions, not all of which may be listed in the documentation. Changes and further development, which may be incompatible, may occur at any time, without warning or notice!

The form name should conform to the defined standard

→ **Mnemonic** FORMNAMINGCONVENTION

→ **Description** The form name should conform to the defined standard

The function name should conform to the defined standard

→ **Mnemonic** FUNCTIONNAMINGCONVENTION

→ **Description** The function name should conform to the defined standard

Avoid calling a function module without handling exceptions

→ **Mnemonic** HANDLEERRORCALLFUNC

→ **Description** Handling the exceptions when calling a function module is optional but should be mandatory to correctly handle errors in production.

The macro name should conform to the defined standard

- **Mnemonic** MACRONAMINGCONVENTION
- **Description** The macro name should conform to the defined standard

The method name should conform to the defined standard

- **Mnemonic** METHODNAMINGCONVENTION
- **Description** The method name should conform to the defined standard

Factorizable Classes

- **Mnemonic** CAC_CL
- **Description** Consider classes refactorization

Factorizable Files

- **Mnemonic** CAC_FI
- **Description** Consider files refactorization

Factorizable Functions

- **Mnemonic** CAC_FN
- **Description** Consider functions refactorization

Factorizable Packages

- **Mnemonic** CAC_PKG
- **Description** Consider packages refactorization

Cloned Classes

- **Mnemonic** CC_CL
- **Description** There shall be no duplicated classes

Cloned Files

- **Mnemonic** CC_FI
- **Description** There shall be no duplicated files

Cloned Functions

- **Mnemonic** CC_FN
- **Description** There shall be no duplicated functions

Cloned Algorithmic

- **Mnemonic** CFTC_FN
- **Description** There shall be no algorithmic cloning

Continue shall not be used

- **Mnemonic** NOCONT
- **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

Do not use "Native SQL" instructions

→ **Mnemonic** NONNATIVESQL

→ **Description** Native SQL should not be used.

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Prevent use of EDITOR-CALLS

→ **Mnemonic** PREVENTEDITORCALL

→ **Description** This statement bypasses the authority checks that are performed when calling the ABAP editor via transaction code.

The program or report name should conform to the defined standard

→ **Mnemonic** PROGREPORTNAMINGCONVENTION

→ **Description** The program or report name should conform to the defined standard

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

2.2. ADA

2.2.1. ADA Metrics

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Header Blocks Of Comment

→ **Mnemonic** BHCO

→ **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

→ **Mnemonic** BLAN

→ **Description** Number of blank lines of code in the source file(s).

Brace Lines

→ **Mnemonic** BRAC

→ **Description** Number of lines of code containing only a brace in the source file(s).

Case Blocks

→ **Mnemonic** CABL

→ **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

→ **Mnemonic** CASE

→ **Description** Number of 'case' labels in the function

Cyclomatic Complexity

→ **Mnemonic** CCN

→ **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

→ **Mnemonic** CFT

→ **Description** Number of tokens in the control flow of functions

Call Graph Depth

→ **Mnemonic** CGDM

→ **Description** Maximum depth of the call graph.

Comment Lines

→ **Mnemonic** CLOC

→ **Description** Number of lines of comments in the source file(s).

Commented Statements

→ **Mnemonic** CSTAT

→ **Description** Number of Commented Statements.

Minimum Number of Cycles

→ **Mnemonic** CYCL

→ **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Declare operators

→ **Mnemonic** DECBL

→ **Description** Number of Declare operators

Default Statement

→ **Mnemonic** DEFT

→ **Description** Number of 'default' blocks in 'switch' in the function

Distinct Operands

→ **Mnemonic** DOPD

→ **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

Entry Statements

→ **Mnemonic** ENTRY

→ **Description** Number of Entry statements

Exception When blocks

→ **Mnemonic** EXBL

→ **Description** Number of 'when' blocks in 'exception handler'.

Exception handlers

→ **Mnemonic** EXGR

→ **Description** Number of Exception handlers

For Statements

→ **Mnemonic** FOR

→ **Description** Number of 'for' statements in the function

Structures Added

→ **Mnemonic** SADD

→ **Description** Number of control structures added since the previous version.

Structures Modified

→ **Mnemonic** SMOD

→ **Description** Number of control structures modified since the previous version.

Structures Removed

→ **Mnemonic** SREM

→ **Description** Number of control structures removed since the previous version.

Number of Structures

→ **Mnemonic** SSIZ

→ **Description** Number of control structures: iterations, selections, sequences

Goto Statements

→ **Mnemonic** GOTO

→ **Description** Number of 'goto' statements

Header Lines Of Comment

→ **Mnemonic** HCOM

→ **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

→ **Mnemonic** HLOC

→ **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

→ **Mnemonic** ICC

→ **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

→ **Mnemonic** ICFTC

→ **Description** Number of duplicated tokens in control flow of functions

If Statements

→ **Mnemonic** IF

→ **Description** Number of 'if' statements

Generic object

→ **Mnemonic** ISGEN

→ **Description** The object is declared generic

Label Statements

→ **Mnemonic** LABEL

→ **Description** Number of Label statements

Line Count

→ **Mnemonic** LC

→ **Description** Number of lines.

Loop Statements

→ **Mnemonic** LOOP

→ **Description** Number of loop statements in the function

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

AND operators

→ **Mnemonic** NBAND

→ **Description** Number of AND operators

Constants

→ **Mnemonic** NBCONST

→ **Description** Number of Constants

Private constant

→ **Mnemonic** NBCONSTPRIV

→ **Description** Number of Private constants

Public constants

→ **Mnemonic** NBCONSTPUB

→ **Description** Number of Public constants

Declared functions

→ **Mnemonic** NBDFUNC

→ **Description** Number of Declared functions/procedures

Private functions/Procedures

→ **Mnemonic** NBDFUNCPRIV

→ **Description** Number of Private function/Procedure

Public functions

→ **Mnemonic** NBDFUNCPUB

→ **Description** Number of Public functions/procedures

Exceptions

→ **Mnemonic** NBEXCEPT

→ **Description** Number of Declared Exceptions

Private exceptions

→ **Mnemonic** NBEXCEPTPRIV

→ **Description** Number of Private exceptions

Public exceptions

→ **Mnemonic** NBEXCEPTPUB

→ **Description** Number of Public exceptions

Separate functions/procedures

→ **Mnemonic** NBFUNCDSEP

→ **Description** Number of Separate functions/procedures

OR operators

→ **Mnemonic** NBOR

→ **Description** Number of OR operators

Separate packages

→ **Mnemonic** NBPACKDSEP

→ **Description** Number of package declared Separate

Protected objects

→ **Mnemonic** NBPROTOBJDSEP

→ **Description** Number of Declred Protected objects

Renamed objects

→ **Mnemonic** NBRENA

→ **Description** Number of Renamed object

Subtypes

→ **Mnemonic** NBSTYP

→ **Description** Number of Subtypes

Separate tasks

→ **Mnemonic** NBTASKDSEP

→ **Description** Number of task declared Separate

Types

→ **Mnemonic** NBTYP

→ **Description** Number of Types

Derived types

→ **Mnemonic** NBTYPDRV

→ **Description** Number of Derived types

Private types

→ **Mnemonic** NBTYPPRIV

→ **Description** Number of Private types

Public types

→ **Mnemonic** NBTYPPUB

→ **Description** Number of Public types

Variables

→ **Mnemonic** NBVAR

→ **Description** Number of Variables

Private variables

→ **Mnemonic** NBVARPRIV

→ **Description** Number of Private variables

Public variables

→ **Mnemonic** NBVARPUB

→ **Description** Number of Public variables

With statements

→ **Mnemonic** NBWITH

→ **Description** Number of With statements

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Non-Cyclic Paths

→ **Mnemonic** PATH

→ **Description** Number of non-cyclic paths in the function.

Orelse operators

→ **Mnemonic** OREL

→ **Description** Number of 'orelse' operators

Raise statements

→ **Mnemonic** RAISE

→ **Description** Number of Raise statements

Return Statements

→ **Mnemonic** RETURN

→ **Description** Number of 'return' statements in the function

Repeated Code Blocks

→ **Mnemonic** RS

→ **Description** Duplicated blocks in the function

Skipped Lines of Comment code

→ **Mnemonic** SKLC

→ **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

→ **Mnemonic** SLOC

→ **Description** Number of lines of source code in the source file(s).

Executable Statements

→ **Mnemonic** STAT

→ **Description** Total number of executable statements.

Switch Statements

→ **Mnemonic** SWIT

→ **Description** Number of 'switch' statements in the function

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

While Statements

- **Mnemonic** WHIL
- **Description** Number of 'while' statements in the function

2.2.2. ADA Ruleset

Backward Goto shall not be used

- **Mnemonic** BWGOTO
- **Description** Backward gotos shall not be used.

Commented-out Source Code is not allowed

- **Mnemonic** R_CSTAT
- **Description** Commented-out Source Code is not allowed

Missing final else

- **Mnemonic** ELSEFINAL
- **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Exit Label shall be named

- **Mnemonic** EXTLABEL
- **Description** Each exit label shall be named.

Use 'exit when' instead of if... exit syntax

- **Mnemonic** EXTWHEN
- **Description** Use 'exit when' instead of if... exit syntax.

Each loop shall be named

- **Mnemonic** LOOPNAMED
- **Description** Each loop shall be named.

Abort shall not be used

- **Mnemonic** NOABORT
- **Description** Use of 'abort'

Factorizable Classes

- **Mnemonic** CAC_CL
- **Description** Consider classes refactorization

Factorizable Files

- **Mnemonic** CAC_FI
- **Description** Consider files refactorization

Factorizable Functions

- **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

Delay shall not be used

→ **Mnemonic** NODELAY

→ **Description** Use of 'delay'

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

→ **Mnemonic** NOGOTO

→ **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

There shall be no 'when others' in exception handler

→ **Mnemonic** NOWHEN_OTHERS

→ **Description** There shall be no 'when others' in exception handler.

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Parameters shall be ordered: 'IN', 'OUT', 'IN OUT'.

→ **Mnemonic** PARAMORDER

→ **Description** Parameters shall be ordered: 'IN', 'OUT', 'IN OUT'.

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Multiple Exit in loop

→ **Mnemonic** SGLEXT

→ **Description** There shall be a single exit by loop.

2.3. C

2.3.1. C Metrics

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Assignment Operators

→ **Mnemonic** ASOP

→ **Description** Number of assignment operators used in the source file

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Header Blocks Of Comment

→ **Mnemonic** BHCO

→ **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

→ **Mnemonic** BLAN

→ **Description** Number of blank lines of code in the source file(s).

Brace Lines

→ **Mnemonic** BRAC

→ **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

→ **Mnemonic** BRKL

→ **Description** Number of 'break' statements in loop in the function

Break in Switch

→ **Mnemonic** BRKS

→ **Description** Number of 'break' statements in 'switch' in the function

Case Blocks

→ **Mnemonic** CABL

→ **Description** Number of 'case' blocks in 'switch' in the function

Calls To

→ **Mnemonic** CAL2

→ **Description** Number of explicit calls to the function.

Called Functions

→ **Mnemonic** CALD

→ **Description** Number of distinct functions defined in the project source file and called by the function.

Calls From

→ **Mnemonic** CALF

→ **Description** Number of explicit calls from the function.

Calling Functions

→ **Mnemonic** CALI

→ **Description** Number of distinct functions calling the function.

Called External Functions

→ **Mnemonic** CALX

→ **Description** Number of distinct external functions called by the function - external i.e. not defined in the project

Case Labels

→ **Mnemonic** CASE

→ **Description** Number of 'case' labels in the function

Cyclomatic Complexity

→ Mnemonic CCN**→ Description** Number of linearly independent paths in the function control graph.**Recursive Calls****→ Mnemonic CDRI****→ Description** Number of directly recursive calls in the function.**Control Flow Token****→ Mnemonic CFT****→ Description** Number of tokens in the control flow of functions**Called Depth****→ Mnemonic CGDD****→ Description** Maximum depth of called functions.**Calling Depth****→ Mnemonic CGDI****→ Description** Maximum depth of calling functions.**Call Graph Depth****→ Mnemonic CGDM****→ Description** Maximum depth of the call graph.**Minimum Number of Indirect Cycles****→ Mnemonic CIRI****→ Description** Minimum number of indirect call graph cycles in which the function is involved (excluding recursive calls).**Comment Lines****→ Mnemonic CLOC****→ Description** Number of lines of comments in the source file(s).**Continue Statements****→ Mnemonic CONT****→ Description** Number of 'continue' statements in the function**Comparison Operators****→ Mnemonic CPOP****→ Description** Number of comparison operators used in the source file**Commented Statements****→ Mnemonic CSTAT****→ Description** Number of Commented Statements.**Minimum Number of Cycles**

→ **Mnemonic** CYCL

→ **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Default Statement

→ **Mnemonic** DEFT

→ **Description** Number of 'default' blocks in 'switch' in the function

Distinct Operands

→ **Mnemonic** DOPD

→ **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

→ **Mnemonic** DOWH

→ **Description** Number of 'do...while' statements in the function

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

For Statements

→ **Mnemonic** FOR

→ **Description** Number of 'for' statements in the function

Structures Added

→ **Mnemonic** SADD

→ **Description** Number of control structures added since the previous version.

Structures Modified

→ **Mnemonic** SMOD

→ **Description** Number of control structures modified since the previous version.

Structures Removed

→ **Mnemonic** SREM

→ **Description** Number of control structures removed since the previous version.

Number of Structures

→ **Mnemonic** SSIZ

→ **Description** Number of control structures: iterations, selections, sequences

Goto Statements

→ **Mnemonic** GOTO

→ **Description** Number of 'goto' statements

Header Lines Of Comment

→ **Mnemonic** HCOM

→ **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

→ **Mnemonic** HLOC

→ **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

→ **Mnemonic** ICC

→ **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

→ **Mnemonic** ICFTC

→ **Description** Number of duplicated tokens in control flow of functions

If Statements

→ **Mnemonic** IF

→ **Description** Number of 'if' statements

Line Count

→ **Mnemonic** LC

→ **Description** Number of lines.

Use of longjump

→ **Mnemonic** LONGJMP

→ **Description** Use of longjump

Loop Statements

→ **Mnemonic** LOOP

→ **Description** Number of loop statements in the function

Memory Allocation

→ **Mnemonic** MEMALLOC

→ **Description** Memory Allocation

Memory Freeing

→ **Mnemonic** MEMFREE

→ **Description** Memory Freeing

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Number of Parameters

→ **Mnemonic** NOP

→ **Description** Number of formal parameters in the function

Non-Cyclic Paths

→ **Mnemonic** PATH

→ **Description** Number of non-cyclic paths in the function.

Use of offsetof

→ **Mnemonic** OFFSETOF

→ **Description** Use of offsetof

Orelse operators

→ **Mnemonic** OREL

→ **Description** Number of 'orelse' operators

Number of #DEFINE

→ **Mnemonic** P_DEFINE

→ **Description** Number of #DEFINE

Number of #ELIF

→ **Mnemonic** P_ELIF

→ **Description** Number of #ELIF

Number of #ELSE

→ **Mnemonic** P_ELSE

→ **Description** Number of #ELSE

Number of #ENDIF

→ **Mnemonic** P_ENDIF

→ **Description** Number of #ENDIF

Number of #ERROR

→ **Mnemonic** P_ERROR

→ **Description** Number of #ERROR

Number of #IF

→ **Mnemonic** P_IF

→ **Description** Number of #IF

Number of #IFDEF

→ **Mnemonic** P_IFDEF

→ **Description** Number of #IFDEF

Number of #IFDEF

→ **Mnemonic** P_IFNDEF

→ **Description** Number of #IFNDEF

Number of Include

→ **Mnemonic** P_INCLUDE

→ **Description** Number of Include

Compiler FLAG Nested Level

→ **Mnemonic** P_NEST

→ **Description** Compiler FLAG Nested Level

Number of #PRAGMA

→ **Mnemonic** P_PRAGMA

→ **Description** Number of #PRAGMA

Number of #UNDEF

→ **Mnemonic** P_UNDEF

→ **Description** Number of #UNDEF

Number of #WARNING

→ **Mnemonic** P_WARNING

→ **Description** Number of #WARNING

Return Statements

→ **Mnemonic** RETURN

→ **Description** Number of 'return' statements in the function

Repeated Code Blocks

→ **Mnemonic** RS

→ **Description** Duplicated blocks in the function

Use of setjump

→ **Mnemonic** SETJMP

→ **Description** Use of setjump

Signal Functions

→ **Mnemonic** SIGNAL

→ **Description** Use of signal Functions

Skipped Lines of Comment code

→ **Mnemonic** SKLC

→ **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

→ **Mnemonic** SLOC

→ **Description** Number of lines of source code in the source file(s).

Special Operators

→ **Mnemonic** SPOP

→ **Description** Number of special operators used in the source file

Executable Statements

→ **Mnemonic** STAT

→ **Description** Total number of executable statements.

IO Functions

→ **Mnemonic** STDIO

→ **Description** Use IO Functions

String Conversions

→ **Mnemonic** STRINGCONV

→ **Description** Use of String Conversions

Switch Statements

→ **Mnemonic** SWIT

→ **Description** Number of 'switch' statements in the function

System Functions

→ **Mnemonic** SYSCOM

→ **Description** Use of system Functions

Ternary operators

→ **Mnemonic** TERN

→ **Description** Number of ternary operators i.e. ?:

Time Handling

→ **Mnemonic** TIMEHDL

→ **Description** Use of Time Handling

Operand Occurrences

- **Mnemonic** TOPD
- **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

- **Mnemonic** TOPT
- **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Lines Added

- **Mnemonic** LADD
- **Description** Number of lines added since the previous version.

Lines Modified

- **Mnemonic** LMOD
- **Description** Number of lines modified since the previous version.

Lines Removed

- **Mnemonic** LREM
- **Description** Number of lines removed since the previous version.

While Statements

- **Mnemonic** WHIL
- **Description** Number of 'while' statements in the function

2.3.2. C Ruleset

Missing Break

- **Mnemonic** BRKFINAL
- **Description** An unconditional break statement shall terminate every non-empty switch clause (see [MISRA-C:2004]: RULE 15.2).

Backward Goto shall not be used

- **Mnemonic** BWGOTO
- **Description** Backward gotos shall not be used.

Missing compound statement

- **Mnemonic** COMPOUND
- **Description** The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if

- **Mnemonic** COMPOUNDIF
- **Description** An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Commented-out Source Code is not allowed

- **Mnemonic** R_CSTAT
- **Description** Commented-out Source Code is not allowed

Missing Default

- **Mnemonic** DEFAULT
- **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Dynamic Memory Allocation shall not be used

- **Mnemonic** DYNMEMALLOC
- **Description** Dynamic heap memory allocation shall not be used. This precludes the use of the functions calloc, malloc, realloc and free (see [MISRA-C:2004]: RULE 20.4)

Missing final else

- **Mnemonic** ELSEFINAL
- **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Macro longjmp or setjmp shall not be used

- **Mnemonic** JUMP
- **Description** (The setjmp macro and the longjmp function shall not be used (see [MISRA-C:2004]: RULE 20.7).

Nesting Level of Preprocessing directives is too high

- **Mnemonic** R_MAXPNEST
- **Description** Nesting Level of Preprocessing directives is too high

Assignment in Boolean

- **Mnemonic** NOASGCOND
- **Description** Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison

- **Mnemonic** NOASGINBOOL
- **Description** Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes

- **Mnemonic** CAC_CL
- **Description** Consider classes refactorization

Factorizable Files

- **Mnemonic** CAC_FI
- **Description** Consider files refactorization

Factorizable Functions

→ **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

There shall be a no code before first case

→ **Mnemonic** NOCODEBEFORECASE

→ **Description** There shall be a no code before the first case of a switch statement.

Continue shall not be used

→ **Mnemonic** NOCONT

→ **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Fallthrough shall be avoided

→ **Mnemonic** NOFALLTHROUGH

→ **Description** There shall be no fallthrough the next case in a switch statement.

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

→ **Mnemonic** NOGOTO

→ **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Label out a switch

→ **Mnemonic** NOLABEL

→ **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Recursion are not allowed

→ **Mnemonic** NORECURSION

→ **Description** Functions shall not called themselves either directly or indirectly (see [MISRA-C:2004]: RULE 16.2).

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Macro offsetof shall not be used

→ **Mnemonic** OFFSETOF

→ **Description** The macro offsetof, in library <stddef.h>, shall not be used (see [MISRA-C:2004]: RULE 20.6).

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

→ **Mnemonic** RISKYEMPTY

→ **Description** Risky Empty Statement

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

Signal or Raise shall not be used

→ **Mnemonic** SIGNAL

→ **Description** The signal handling facilities of <signal.h> shall not be used (see [MISRA-C:2004]: RULE 20.8).

IO Functions shall not be used

→ **Mnemonic** STDIO

→ **Description** The input/output library <stdio.h> shall not be used in production code (see [MISRA-C:2004]: RULE 20.9).

'atof, atoi or atol' shall not be used

→ **Mnemonic** STRINGCONV

→ **Description** The library functions atof, atoi and atol from library <stdlib.h> shall not be used (see [MISRA-C:2004]: RULE 20.10).

'abort, exit, getenv or system' shall not be used

→ **Mnemonic** SYSCOM

→ **Description** The library functions abort, exit, getenv and system from library <stdlib.h> shall not be used (see [MISRA-C:2004]: RULE 20.11).

Time Handling Functions shall not be used

→ **Mnemonic** TIMEHDL

→ **Description** The time handling functions of library <time.h> shall not be used: time, strftime, clock, difftime, mktime (see [MISRA-C:2004]: RULE 20.12).

2.4. COBOL

2.4.1. COBOL Metrics

Arithmetic Operators

→ **Mnemonic** AROP

→ **Description** Number of arithmetic operators

Blank Lines

→ **Mnemonic** BLAN

→ **Description** Number of blank lines of code in the source file(s).

CALL Statements

→ **Mnemonic** CALL

→ **Description** Number of CALL statements

Cyclomatic Complexity

→ **Mnemonic** CCN

→ **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

→ **Mnemonic** CFT

→ **Description** Number of tokens in the control flow of functions

Call Graph Depth

→ **Mnemonic** CGDM

→ **Description** Maximum depth of the call graph.

Comment Lines

→ **Mnemonic** CLOC

→ **Description** Number of lines of comments in the source file(s).

Comment lines with code

→ **Mnemonic** CLOC_CODE

→ **Description** Number of lines of comments in the source file(s) whose first word is a keyword.

Comment lines without alphabetic characters

→ **Mnemonic** CLOC_NULL

→ **Description** Number of lines of comments in the source file(s) without alphabetic character.

Real comment lines with alphabetic characters

→ **Mnemonic** CLOC_REAL

→ **Description** Number of real lines of comments in the source file(s) with alphabetic characters.

Conditions

→ **Mnemonic** COND

→ **Description** Number of conditions

Commented Statements

→ **Mnemonic** CSTAT

→ **Description** Number of Commented Statements.

Minimum Number of Cycles

→ **Mnemonic** CYCL

→ **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Debug lines

→ **Mnemonic** DEBUG

→ **Description** Number of lines of debug in the source file(s).

DISPLAY statements

→ **Mnemonic** DISPLAY

→ **Description** Number of DISPLAY statements

Distinct Operands

→ **Mnemonic** DOPD

→ **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operands in Data Div.

→ **Mnemonic** DOPD_DD

→ **Description** Number of distinct operands in Data Division: variables and constants ([Halstead,76]: n2)

Distinct Operands in Procedure Div.

→ **Mnemonic** DOPD_PD

→ **Description** Number of distinct operands in Procedure Division: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Distinct Operators in Data Div.

→ **Mnemonic** DOPT_DD

→ **Description** Number of distinct operators in Data Division: language keywords ([Halstead,76]: n1)

Distinct Operators in Procedure Div.

→ **Mnemonic** DOPT_PD

→ **Description** Number of distinct operators in Procedure Division: language keywords ([Halstead,76]: n1)

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

EVALUATE Statements

→ **Mnemonic** EVAL

→ **Description** Number of EVALUATE statements

Call to exit

→ **Mnemonic** EXIT

→ **Description** Number of calls to the exit function

File Declarations

→ **Mnemonic** FD

→ **Description** Number of file declarations

Files Used

→ **Mnemonic** FDUS

→ **Description** Number of references to files

Goto Statements

→ **Mnemonic** GOTO

→ **Description** Number of 'goto' statements

Cloned Code

→ **Mnemonic** ICC

→ **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

→ **Mnemonic** ICFTC

→ **Description** Number of duplicated tokens in control flow of functions

Is IDMS active

→ **Mnemonic** IDMS_ACTIVE

→ **Description** Is IDMS active in program

IDMS instructions called

→ **Mnemonic** IDMS_CALLBD

→ **Description** Number of IDMS instructions called

IDMS records called

→ **Mnemonic** IDMS_CALLREC

→ **Description** Number of IDMS records called

IDMS calls for modification

→ **Mnemonic** IDMS_MOD

→ **Description** Number of calls for modification

IDMS calls for reading/searching

→ **Mnemonic** IDMS_READ

→ **Description** Number of calls for reading/searching

IDMS subschema definition

→ **Mnemonic** IDMS_SSCH

→ **Description** Number of IDMS subschema definition

If Statements

→ **Mnemonic** IF

→ **Description** Number of 'if' statements

Line Count

→ **Mnemonic** LC

→ **Description** Number of lines.

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Number of paragraphs

→ **Mnemonic** PARA

→ **Description** Number of paragraphs.

PERFORM Statements

→ **Mnemonic** PERF

→ **Description** Number of PERFORM statements

Repeated Code Blocks

→ **Mnemonic** RS

→ **Description** Duplicated blocks in the function

Data Declarations

→ **Mnemonic** SD

→ **Description** Number of data declarations

Data Used

→ **Mnemonic** SDUS

→ **Description** Number of used data

Number of Sections

→ **Mnemonic** SECT

→ **Description** Number of sections.

Source Lines Of Code

→ **Mnemonic** SLOC

→ **Description** Number of lines of source code in the source file(s).

Executable Statements

→ **Mnemonic** STAT

→ **Description** Total number of executable statements.

STOP Statements

→ **Mnemonic** STOP

→ **Description** Number of STOP statements

TIMES Clauses

→ **Mnemonic** TIME

→ **Description** Number of TIMES clauses in PERFORM statements

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operand Occurrences in Data Div.

→ **Mnemonic** TOPD_DD

→ **Description** Number of occurrences of operands in Data Division: variables and constants ([Halstead,76]: N2)

Operand Occurrences in Procedure Div.

→ **Mnemonic** TOPD_PD

→ **Description** Number of occurrences of operands in Procedure Division: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Operator Occurrences in Data Div.

→ **Mnemonic** TOPT_DD

→ **Description** Number of occurrences of operators in Data Division: language keywords ([Halstead,76]: N1)

Operator Occurrences in Procedure Div.

→ **Mnemonic** TOPT_PD

→ **Description** Number of occurrences of operators in Procedure Division: language keywords ([Halstead,76]: N1)

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

UNTIL Clauses

→ **Mnemonic** UNTL

→ **Description** Number of UNTIL clauses in PERFORM statements

VARYING Clauses

→ **Mnemonic** VARY

→ **Description** Number of VARYING clauses in PERFORM statements

WHEN Clauses

- **Mnemonic** WHEN
- **Description** Number of WHEN and WHENOTHER clauses in EVALUATE Statements

2.4.2. COBOL Ruleset

BLOCK Clause

- **Mnemonic** BLOCKSIZE
- **Description** In the FILE-DESCRIPTION section, each file description shall always use the BLOCK CONTAINS 0 RECORDS clause. The system will assign the BLOCK-SIZE automatically when allocating the file.

Column 7 for * and D Only

- **Mnemonic** COLUMN7
- **Description** Only * and D shall be used in column 7.

Comment Division

- **Mnemonic** COMMENT_DIVISION
- **Description** A comment is recommended before each division.

Comment FD

- **Mnemonic** COMMENT_FD
- **Description** A comment is recommended before each file description.

Comment First Level

- **Mnemonic** COMMENT_FIRST_LEVEL
- **Description** A comment is recommended before each first level of IF or PERFORM.

Comment Variable 01 and 77

- **Mnemonic** COMMENT_FIRST_VARIABLE
- **Description** A comment is recommended before each variable 01 and 77.

Empty lines around DIVISION

- **Mnemonic** CPRS_DIVISION
- **Description** An empty line shall precede and follow a DIVISION.

Empty line after EXIT

- **Mnemonic** CPRS_EXIT
- **Description** An empty line shall follow an EXIT statement.

Bad statement indentation

- **Mnemonic** CPRS_INDENT
- **Description** The nested statements shall be indented.

Bad indentation of scope terminator

→ **Mnemonic** CPRS_SCOPE_TERMINATOR

→ **Description** Scope terminators must be on the same column as the beginning of the block to facilitate program readability.

Empty line after SECTION

→ **Mnemonic** CPRS_SECTION

→ **Description** An empty line shall follow a SECTION.

Variable declaration format

→ **Mnemonic** DCLWS

→ **Description** A variable shall be declared in the WORKING STORAGE using the format ^W

Paragraphs having exact same name

→ **Mnemonic** R_DUPPARA

→ **Description** Paragraphs having exact same name in the same PROGRAM-ID is forbidden.

Missing END-EVALUATE

→ **Mnemonic** EVALWITHENDEVAL

→ **Description** An EVALUATE statement shall be closed by END-EVALUATE

Close file once

→ **Mnemonic** FILECLOSEONCE

→ **Description** A file shall be closed only once

Close open file

→ **Mnemonic** FILEOPENCLOSE

→ **Description** A file shall be opened and closed in the same program

Open file once

→ **Mnemonic** FILEOPENONCE

→ **Description** A file shall be opened only once

Use FILE STATUS

→ **Mnemonic** FILESTATUS

→ **Description** FILE STATUS shall be used to manage I/O errors.

Single GOBACK

→ **Mnemonic** GOBACK

→ **Description** Only a single GOBACK shall be used in a subprogram.

IDMS FIND CURRENT

→ **Mnemonic** IDMSFINDCURRENT

→ **Description** IDMS FIND CURRENT is forbidden

IDMS One modify by PERFORM

→ **Mnemonic** IDMSONEMODFORPERF

→ **Description** Each IDMS modify statement (MODIFY/ERASE/STORE) should be in a specific perform

IDMS One same call

→ **Mnemonic** IDMSONESAMECALL

→ **Description** Avoid duplicated IDMS call.

IDMS Ready Protected Update

→ **Mnemonic** IDMSREADYPRTUPD

→ **Description** Each IDMS Ready Update statement should be defined in PROTECTED mode.

IDMS Return Code

→ **Mnemonic** IDMSRETURNCODE

→ **Description** After each IDMS statement, return code should be checked.

Missing END-IF

→ **Mnemonic** IFWITHENDIF

→ **Description** An IF statement shall be closed by an END-IF

Avoid using inline PERFORM with too many lines of code

→ **Mnemonic** INLINE_PERFORM_SIZE

→ **Description** Avoid Cobol programs containing PERFORM - END-PERFORM loops with more than 80 lines.

Standard Label

→ **Mnemonic** LABELSTD

→ **Description** In the FILE-DESCRIPTION section, each file description shall always use the LABEL RECORD STANDARD clause. Only the standard labels are checked by the system.

Missing END-ADD

→ **Mnemonic** MISSING_END_ADD

→ **Description** An ADD statement shall be closed by an END-ADD.

Missing END-CALL

→ **Mnemonic** MISSING_END_CALL

→ **Description** An CALL statement shall be closed by an END-CALL.

Missing END-COMPUTE

→ **Mnemonic** MISSING_END_COMPUTE

→ **Description** An COMPUTE statement shall be closed by an END-COMPUTE.

Missing END-DELETE

→ **Mnemonic** MISSING_END_DELETE

→ **Description** An DELETE statement shall be closed by an END-DELETE.

Missing END-DIVIDE

→ **Mnemonic** MISSING_END_DIVIDE

→ **Description** An DIVIDE statement shall be closed by an END-DIVIDE.

Missing END-MULTIPLY

→ **Mnemonic** MISSING_END_MULTIPLY

→ **Description** An MULTIPLY statement shall be closed by an END-MULTIPLY.

Missing END-READ

→ **Mnemonic** MISSING_END_READ

→ **Description** An READ statement shall be closed by an END-READ.

Missing END-RETURN

→ **Mnemonic** MISSING_END_RETURN

→ **Description** An RETURN statement shall be closed by an END-RETURN.

Missing END-REWRITE

→ **Mnemonic** MISSING_END_REWRITE

→ **Description** An REWRITE statement shall be closed by an END-REWRITE.

Missing END-SEARCH

→ **Mnemonic** MISSING_END_SEARCH

→ **Description** An SEARCH statement shall be closed by an END-SEARCH.

Missing END-START

→ **Mnemonic** MISSING_END_START

→ **Description** An START statement shall be closed by an END-START.

Missing END-STRING

→ **Mnemonic** MISSING_END_STRING

→ **Description** An STRING statement shall be closed by an END-STRING.

Missing END-SUBTRACT

→ **Mnemonic** MISSING_END_SUBTRACT

→ **Description** An SUBTRACT statement shall be closed by an END-SUBTRACT.

Missing END-UNSTRING

→ **Mnemonic** MISSING_END_UNSTRING

→ **Description** An UNSTRING statement shall be closed by an END-UNSTRING.

Missing END-WRITE

→ **Mnemonic** MISSING_END_WRITE

→ **Description** An WRITE statement shall be closed by an END-WRITE.

Missing FILLER

→ **Mnemonic** MISSING_FILLER

→ **Description** Even the 'FILLER' word is optional since Cobol85, it is recommended to write it.

No more than 3 nested IF

→ **Mnemonic** NESTEDIF

→ **Description** There shall be no more than 3 nested IF statements

Nested Program

→ **Mnemonic** NESTED_PROGRAM

→ **Description** Nested program is not recommended

ALTER shall not be used

→ **Mnemonic** NOALTER

→ **Description** The ALTER statement shall not be used. Labels are decided only at execution time.

Factorizable Classes

→ **Mnemonic** CAC_CL

→ **Description** Consider classes refactorization

Factorizable Files

→ **Mnemonic** CAC_FI

→ **Description** Consider files refactorization

Factorizable Functions

→ **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

- **Mnemonic** CFTC_FN
- **Description** There shall be no algorithmic cloning

No Conditional GOTO

- **Mnemonic** NOCONDGOTO
- **Description** Conditional GO TO shall not be used. Use EVALUATE instead.

No MOVE CORRESPONDING

- **Mnemonic** NOCORRESPONDING
- **Description** MOVE CORRESPONDING shall not be used.

COMPUTE instead of ADD

- **Mnemonic** NOCPXADD
- **Description** COMPUTE shall be used to add more than 2 data instead of ADD.

COMPUTE instead of SUBTRACT

- **Mnemonic** NOCPXSUBTRACT
- **Description** COMPUTE shall be used to add more than 2 data instead of SUBTRACT.

No DEBUG MODE

- **Mnemonic** NODEBUG
- **Description** DEBUGGING-MODE shall not be used

COMPUTE instead of DIVIDE

- **Mnemonic** NODIVIDE
- **Description** COMPUTE shall be used instead of DIVIDE.

FIXME shall not be committed in sources code

- **Mnemonic** R_NOFIXME
- **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

No INITIALIZE

- **Mnemonic** NOINITIALIZE
- **Description** INITIALIZE shall not be used. Use MOVE to initialize variable.

COMPUTE instead of MULTIPLY

- **Mnemonic** NOMULTIPLY
- **Description** COMPUTE shall be used instead of MULTIPLY.

No procedural COPY

- **Mnemonic** NOPROCCOPY
- **Description** Procedural COPY clauses shall not be used. Use subprograms instead.

No RENAMES

- **Mnemonic** NORENAMES
- **Description** The RENAMES clause shall not be used.

Avoid Duplicated Blocks in Function

- **Mnemonic** RS_FN
- **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

- **Mnemonic** R_NOTODO
- **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

No Variables S9(9)

- **Mnemonic** NOVARS9
- **Description** The variables shall not be declared in S9(9) COMP. It implies a conversion

Avoid GOTO jumps out of PERFORM range

- **Mnemonic** NO_GOTO_OUT_OF_PERFORM_RANGE
- **Description** Avoid Cobol Programs containing sections or paragraphs that are called by PERFORM statements and that contain a GO TO statement to another section or paragraph that is not in the scope of the initial PERFORM.

Avoid OPEN/CLOSE inside loops

- **Mnemonic** NO_OPEN_CLOSE_INSIDE_LOOP
- **Description** Avoid Cobol programs using OPEN or CLOSE in loops. Following loops are taken into account:
 - PERFORM TIMES / UNTIL / VARYING

Avoid accessing data by using the position and length

- **Mnemonic** NO_REFERENCE_ACCESS
- **Description** Avoid Cobol programs accessing part of data by using a position and a length.

Use COMP for OCCURS

- **Mnemonic** OCCURSCOMP
- **Description** For the OCCURS DEPENDING ON clause, the corresponding item shall be declared using COMP or BINARY.

Avoid mixing paragraphs and sections

- **Mnemonic** PARA_OR_SECT_ONLY
- **Description** A program should not mix paragraphs and sections.

Perform with no THRU

- **Mnemonic** PERFORMWITHTHRU
- **Description** The call of a paragraph shall be made in the use of PERFORM paragraphName THRU paragraphNameExit.

Bad paragraph position used in PERFORM

→ **Mnemonic** POSITION_OF_PERFORM_RANGE

→ **Description** On a PERFORM range: P1 THRU P2, P1 must be declared before P2.

READ-WRITE Instruction

→ **Mnemonic** READWRITE

→ **Description** READ A INTO B or WRITE A FROM B forms shall be used for reading/writing a file.

Avoid using READ statement without AT END clause

→ **Mnemonic** READ_AT_END

→ **Description** Avoid Cobol programs using READ statements without the AT END clause.

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Statement shall be in uppercase

→ **Mnemonic** UPPERCASE

→ **Description** A COBOL statement shall be written in uppercase to keep the program readable.

Use SYNCHRONIZED

→ **Mnemonic** USESYNCH

→ **Description** SYNCHRONIZED shall be used for COMP, COMP-1, COMP-2, POINTER and INDEX variables.

Homonymous variable shall not be used

→ **Mnemonic** VARNAME

→ **Description** There shall be no homonymous variables.

Use WHEN OTHER

→ **Mnemonic** WHENOTHER

→ **Description** EVALUATE shall end by a WHEN OTHER clause.

2.5. C++

2.5.1. C++ Metrics

Constant Data

→ **Mnemonic** ACST

→ **Description** Number of constant data

Number of Attributes

→ **Mnemonic** ANBR

→ **Description** Number of attributes

Number of data without accessibility

- **Mnemonic** ANON
- **Description** Number of data without accessibility

Andthen Operators

- **Mnemonic** ANTH
- **Description** Number of 'andthen' operators

Public Data

- **Mnemonic** APBL
- **Description** Number of public data

Protected Data

- **Mnemonic** APRT
- **Description** Number of protected data

Private data

- **Mnemonic** APRV
- **Description** Number of private data

Assignment Operators

- **Mnemonic** ASOP
- **Description** Number of assignment operators used in the source file

Static Data

- **Mnemonic** ASTA
- **Description** Number of static data

Number of comment blocks

- **Mnemonic** BCOM
- **Description** Number of comment blocks.

Header Blocks Of Comment

- **Mnemonic** BHCO
- **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

- **Mnemonic** BLAN
- **Description** Number of blank lines of code in the source file(s).

Brace Lines

- **Mnemonic** BRAC
- **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

- **Mnemonic** BRKL
- **Description** Number of 'break' statements in loop in the function

Break in Switch

- **Mnemonic** BRKS
- **Description** Number of 'break' statements in 'switch' in the function

Case Blocks

- **Mnemonic** CABL
- **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

- **Mnemonic** CASE
- **Description** Number of 'case' labels in the function

Catch Statements

- **Mnemonic** CATC
- **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

- **Mnemonic** CCN
- **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

- **Mnemonic** CFT
- **Description** Number of tokens in the control flow of functions

Call Graph Depth

- **Mnemonic** CGDM
- **Description** Maximum depth of the call graph.

Comment Lines

- **Mnemonic** CLOC
- **Description** Number of lines of comments in the source file(s).

Continue Statements

- **Mnemonic** CONT
- **Description** Number of 'continue' statements in the function

Comparison Operators

- **Mnemonic** CPOP
- **Description** Number of comparison operators used in the source file

Commented Statements

- **Mnemonic** CSTAT
- **Description** Number of Commented Statements.

Minimum Number of Cycles

- **Mnemonic** CYCL
- **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Depth of Descendant Tree

- **Mnemonic** DDT
- **Description** Maximun depth of the inheritance tree from the class

Default Statement

- **Mnemonic** DEFT
- **Description** Number of 'default' blocks in 'switch' in the function

Depth of Inheritance Tree

- **Mnemonic** DIT
- **Description** Maximun depth of the class inheritance tree

Distinct Operands

- **Mnemonic** DOPD
- **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

- **Mnemonic** DOPT
- **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

- **Mnemonic** DOWH
- **Description** Number of 'do...while' statements in the function

Else Statements

- **Mnemonic** ELSE
- **Description** Number of 'else' statements

For Statements

- **Mnemonic** FOR
- **Description** Number of 'for' statements in the function

Structures Added

- **Mnemonic** SADD
- **Description** Number of control structures added since the previous version.

Structures Modified

- **Mnemonic** SMOD
- **Description** Number of control structures modified since the previous version.

Structures Removed

- **Mnemonic** SREM
- **Description** Number of control structures removed since the previous version.

Number of Structures

- **Mnemonic** SSIZ
- **Description** Number of control structures: iterations, selections, sequences

Goto Statements

- **Mnemonic** GOTO
- **Description** Number of 'goto' statements

Header Lines Of Comment

- **Mnemonic** HCOM
- **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

- **Mnemonic** HLOC
- **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

- **Mnemonic** ICC
- **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

- **Mnemonic** ICFTC
- **Description** Number of duplicated tokens in control flow of functions

If Statements

- **Mnemonic** IF
- **Description** Number of 'if' statements

Line Count

- **Mnemonic** LC
- **Description** Number of lines.

Loop Statements

- **Mnemonic** LOOP
- **Description** Number of loop statements in the function

Multiple Inheritance Indicator

→ **Mnemonic** MII

→ **Description** Number of classes from which the class inherits directly

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Methods without Accessibility

→ **Mnemonic** MNON

→ **Description** Number of methods without any accessibility specifier

Public Methods

→ **Mnemonic** MPBL

→ **Description** Number of public methods

Protected Methods

→ **Mnemonic** MPRT

→ **Description** Number of protected methods

Private Methods

→ **Mnemonic** MPRV

→ **Description** Number of private methods

Static Methods

→ **Mnemonic** MSTA

→ **Description** Number of static methods

Number of Ancestors

→ **Mnemonic** NAC

→ **Description** Number of classes from which the class inherits directly or indirectly

Number of Descendants

→ **Mnemonic** NDC

→ **Description** Number of classes which inherit from the class directly or indirectly

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Number Of Children

→ **Mnemonic** NOC

→ **Description** Number of classes which inherit directly from the class

Number of Parameters

- **Mnemonic** NOP
- **Description** Number of formal parameters in the function

Non-Cyclic Paths

- **Mnemonic** PATH
- **Description** Number of non-cyclic paths in the function.

Orelse operators

- **Mnemonic** OREL
- **Description** Number of 'orelse' operators

Number of #DEFINE

- **Mnemonic** P_DEFINE
- **Description** Number of #DEFINE

Number of #ELIF

- **Mnemonic** P_ELIF
- **Description** Number of #ELIF

Number of #ELSE

- **Mnemonic** P_ELSE
- **Description** Number of #ELSE

Number of #ENDIF

- **Mnemonic** P_ENDIF
- **Description** Number of #ENDIF

Number of #ERROR

- **Mnemonic** P_ERROR
- **Description** Number of #ERROR

Number of #IF

- **Mnemonic** P_IF
- **Description** Number of #IF

Number of #IFDEF

- **Mnemonic** P_IFDEF
- **Description** Number of #IFDEF

Number of #IFDEF

- **Mnemonic** P_IFNDEF
- **Description** Number of #IFNDEF

Number of Include

- **Mnemonic** P_INCLUDE
- **Description** Number of Include

Compiler FLAG Nested Level

- **Mnemonic** P_NEST
- **Description** Compiler FLAG Nested Level

Number of #PRAGMA

- **Mnemonic** P_PRAGMA
- **Description** Number of #PRAGMA

Number of #UNDEF

- **Mnemonic** P_UNDEF
- **Description** Number of #UNDEF

Number of #WARNING

- **Mnemonic** P_WARNING
- **Description** Number of #WARNING

Return Statements

- **Mnemonic** RETURN
- **Description** Number of 'return' statements in the function

Repeated Code Blocks

- **Mnemonic** RS
- **Description** Duplicated blocks in the function

Skipped Lines of Comment code

- **Mnemonic** SKLC
- **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

- **Mnemonic** SLOC
- **Description** Number of lines of source code in the source file(s).

Special Operators

- **Mnemonic** SPOP
- **Description** Number of special operators used in the source file

Executable Statements

- **Mnemonic** STAT

→ **Description** Total number of executable statements.

Switch Statements

→ **Mnemonic** SWIT

→ **Description** Number of 'switch' statements in the function

Ternary operators

→ **Mnemonic** TERN

→ **Description** Number of ternary operators i.e. ?:

Throw Statements

→ **Mnemonic** THRO

→ **Description** Number of 'throw' statements in the function

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

→ **Mnemonic** TRY

→ **Description** Number of 'try' statements in the function

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

While Statements

→ **Mnemonic** WHIL

→ **Description** Number of 'while' statements in the function

Weighted Method per Class

→ **Mnemonic** XWMC

- **Description** Sum of cyclomatic complexities of methods implemented outside the class definition

2.5.2. C++ Ruleset

Missing Break

- **Mnemonic** BRKFINAL
- **Description** An unconditional break statement shall terminate every non-empty switch clause (see [MISRA-C:2004]: RULE 15.2).

Backward Goto shall not be used

- **Mnemonic** BWGOTO
- **Description** Backward gotos shall not be used.

Missing compound statement

- **Mnemonic** COMPOUND
- **Description** The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if

- **Mnemonic** COMPOUNDIF
- **Description** An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Commented-out Source Code is not allowed

- **Mnemonic** R_CSTAT
- **Description** Commented-out Source Code is not allowed

Missing Default

- **Mnemonic** DEFAULT
- **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Missing final else

- **Mnemonic** ELSEFINAL
- **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Nesting Level of Preprocessing directives is too high

- **Mnemonic** R_MAXPNEST
- **Description** Nesting Level of Preprocessing directives is too high

Assignment in Boolean

- **Mnemonic** NOASGCOND
- **Description** Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison

→ **Mnemonic** NOASGINBOOL

→ **Description** Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes

→ **Mnemonic** CAC_CL

→ **Description** Consider classes refactorization

Factorizable Files

→ **Mnemonic** CAC_FI

→ **Description** Consider files refactorization

Factorizable Functions

→ **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

There shall be a no code before first case

→ **Mnemonic** NOCODEBEFORECASE

→ **Description** There shall be a no code before the first case of a switch statement.

Continue shall not be used

→ **Mnemonic** NOCONT

→ **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Fallthrough shall be avoided

→ **Mnemonic** NOFALLTHROUGH

→ **Description** There shall be no fallthrough the next case in a switch statement.

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

→ **Mnemonic** NOGOTO

→ **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Label out a switch

→ **Mnemonic** NOLABEL

→ **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

→ **Mnemonic** RISKYEMPTY

→ **Description** Risky Empty Statement

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

2.6. C#

2.6.1. C# Metrics

Constant Data

- **Mnemonic** ACST
- **Description** Number of constant data

Internal Data

- **Mnemonic** AINT
- **Description** Number of internal data (only applicable to C#)

Number of Attributes

- **Mnemonic** ANBR
- **Description** Number of attributes

Number of data without accessibility

- **Mnemonic** ANON
- **Description** Number of data without accessibility

Andthen Operators

- **Mnemonic** ANTH
- **Description** Number of 'andthen' operators

Public Data

- **Mnemonic** APBL
- **Description** Number of public data

Protected Internal Data

- **Mnemonic** APIN
- **Description** Number of protected internal data (only applicable to C#)

Protected Data

- **Mnemonic** APRT
- **Description** Number of protected data

Private data

- **Mnemonic** APRV
- **Description** Number of private data

Assignment Operators

- **Mnemonic** ASOP
- **Description** Number of assignment operators used in the source file

Static Data

- **Mnemonic** ASTA
- **Description** Number of static data

Number of comment blocks

- **Mnemonic** BCOM
- **Description** Number of comment blocks.

Header Blocks Of Comment

- **Mnemonic** BHCO
- **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

- **Mnemonic** BLAN
- **Description** Number of blank lines of code in the source file(s).

Brace Lines

- **Mnemonic** BRAC
- **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

- **Mnemonic** BRKL
- **Description** Number of 'break' statements in loop in the function

Break in Switch

- **Mnemonic** BRKS
- **Description** Number of 'break' statements in 'switch' in the function

Case Blocks

- **Mnemonic** CABL
- **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

- **Mnemonic** CASE
- **Description** Number of 'case' labels in the function

Catch Statements

- **Mnemonic** CATC
- **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

- **Mnemonic** CCN
- **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

- **Mnemonic** CFT
- **Description** Number of tokens in the control flow of functions

Call Graph Depth

- **Mnemonic** CGDM
- **Description** Maximum depth of the call graph.

Comment Lines

- **Mnemonic** CLOC
- **Description** Number of lines of comments in the source file(s).

Continue Statements

- **Mnemonic** CONT
- **Description** Number of 'continue' statements in the function

Comparison Operators

- **Mnemonic** CPOP
- **Description** Number of comparison operators used in the source file

Commented Statements

- **Mnemonic** CSTAT
- **Description** Number of Commented Statements.

Minimum Number of Cycles

- **Mnemonic** CYCL
- **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Depth of Descendant Tree

- **Mnemonic** DDT
- **Description** Maximum depth of the inheritance tree from the class

Default Statement

- **Mnemonic** DEFT
- **Description** Number of 'default' blocks in 'switch' in the function

Depth of Inheritance Tree

- **Mnemonic** DIT
- **Description** Maximum depth of the class inheritance tree

Distinct Operands

- **Mnemonic** DOPD
- **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

→ **Mnemonic** DOWH

→ **Description** Number of 'do...while' statements in the function

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

For Statements

→ **Mnemonic** FOR

→ **Description** Number of 'for' statements in the function

Foreach Statements

→ **Mnemonic** FORE

→ **Description** Number of 'foreach' statements in the function

Structures Added

→ **Mnemonic** SADD

→ **Description** Number of control structures added since the previous version.

Structures Modified

→ **Mnemonic** SMOD

→ **Description** Number of control structures modified since the previous version.

Structures Removed

→ **Mnemonic** SREM

→ **Description** Number of control structures removed since the previous version.

Number of Structures

→ **Mnemonic** SSIZ

→ **Description** Number of control structures: iterations, selections, sequences

Goto Statements

→ **Mnemonic** GOTO

→ **Description** Number of 'goto' statements

Header Lines Of Comment

→ **Mnemonic** HCOM

→ **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

- **Mnemonic** HLOC
- **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

- **Mnemonic** ICC
- **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

- **Mnemonic** ICFTC
- **Description** Number of duplicated tokens in control flow of functions

If Statements

- **Mnemonic** IF
- **Description** Number of 'if' statements

Line Count

- **Mnemonic** LC
- **Description** Number of lines.

Loop Statements

- **Mnemonic** LOOP
- **Description** Number of loop statements in the function

Multiple Inheritance Indicator

- **Mnemonic** MII
- **Description** Number of classes from which the class inherits directly

Internal Methods

- **Mnemonic** MINT
- **Description** Number of internal methods (only applicable to C#)

Mixed Lines

- **Mnemonic** MLOC
- **Description** Number of lines containing both code and comment in the source files.

Methods without Accessibility

- **Mnemonic** MNON
- **Description** Number of methods without any accessibility specifier

Public Methods

- **Mnemonic** MPBL
- **Description** Number of public methods

Protected Internal Methods

- **Mnemonic** MPIN
- **Description** Number of protected internal methods(only applicable to C#)

Protected Methods

- **Mnemonic** MPRT
- **Description** Number of protected methods

Private Methods

- **Mnemonic** MPRV
- **Description** Number of private methods

Static Methods

- **Mnemonic** MSTA
- **Description** Number of static methods

Number of Ancestors

- **Mnemonic** NAC
- **Description** Number of classes from which the class inherits directly or indirectly

Number of Descendants

- **Mnemonic** NDC
- **Description** Number of classes which inherit from the class directly or indirectly

Maximum Nested Structures

- **Mnemonic** NEST
- **Description** Maximum number of nested structures

Number Of Children

- **Mnemonic** NOC
- **Description** Number of classes which inherit directly from the class

Number of Parameters

- **Mnemonic** NOP
- **Description** Number of formal parameters in the function

Non-Cyclic Paths

- **Mnemonic** PATH
- **Description** Number of non-cyclic paths in the function.

Or else operators

- **Mnemonic** OREL
- **Description** Number of 'or else' operators

Constant Properties

- **Mnemonic** PCST
- **Description** Number of constant properties

Properties with Get

- **Mnemonic** PGET
- **Description** Number of properties with a setter (only applicable to C#)

Internal Properties

- **Mnemonic** PINT
- **Description** Number of internal properties (only applicable to C#)

Properties

- **Mnemonic** PNBR
- **Description** Total number of properties

Properties without Accessibility

- **Mnemonic** PNON
- **Description** Number of properties without accessibility specifier

Public Properties

- **Mnemonic** PPBL
- **Description** Number of public properties

Protected Internal Properties

- **Mnemonic** PPIN
- **Description** Number of protected internal properties (only applicable to C#)

Protected Properties

- **Mnemonic** PPRT
- **Description** Number of protected properties

Private Properties

- **Mnemonic** PPRV
- **Description** Number of private properties

Properties with Set

- **Mnemonic** PSET
- **Description** Number of properties with a getter (only applicable to C#)

Static Properties

- **Mnemonic** PSTA
- **Description** Number of static properties in the class

Number of #DEFINE

- **Mnemonic** P_DEFINE
- **Description** Number of #DEFINE

Number of #ELIF

- **Mnemonic** P_ELIF
- **Description** Number of #ELIF

Number of #ELSE

- **Mnemonic** P_ELSE
- **Description** Number of #ELSE

Number of #ENDIF

- **Mnemonic** P_ENDIF
- **Description** Number of #ENDIF

Number of #ENDREGION

- **Mnemonic** P_ENDREGION
- **Description** Number of #ENDREGION

Number of #ERROR

- **Mnemonic** P_ERROR
- **Description** Number of #ERROR

Number of #IF

- **Mnemonic** P_IF
- **Description** Number of #IF

Number of #IFDEF

- **Mnemonic** P_IFDEF
- **Description** Number of #IFDEF

Number of #IFNDEF

- **Mnemonic** P_IFNDEF
- **Description** Number of #IFNDEF

Compiler FLAG Nested Level

- **Mnemonic** P_NEST
- **Description** Compiler FLAG Nested Level

Number of #PRAGMA

- **Mnemonic** P_PRAGMA
- **Description** Number of #PRAGMA

Number of #REGION

- **Mnemonic** P_REGION
- **Description** Number of #REGION

Number of #UNDEF

- **Mnemonic** P_UNDEF
- **Description** Number of #UNDEF

Number of #WARNING

- **Mnemonic** P_WARNING
- **Description** Number of #WARNING

Return Statements

- **Mnemonic** RETURN
- **Description** Number of 'return' statements in the function

Repeated Code Blocks

- **Mnemonic** RS
- **Description** Duplicated blocks in the function

Skipped Lines of Comment code

- **Mnemonic** SKLC
- **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

- **Mnemonic** SLOC
- **Description** Number of lines of source code in the source file(s).

Special Operators

- **Mnemonic** SPOP
- **Description** Number of special operators used in the source file

Executable Statements

- **Mnemonic** STAT
- **Description** Total number of executable statements.

Switch Statements

- **Mnemonic** SWIT
- **Description** Number of 'switch' statements in the function

Ternary operators

- **Mnemonic** TERN

→ **Description** Number of ternary operators i.e. ?:

Throw Statements

→ **Mnemonic** THRO

→ **Description** Number of 'throw' statements in the function

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

→ **Mnemonic** TRY

→ **Description** Number of 'try' statements in the function

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

While Statements

→ **Mnemonic** WHIL

→ **Description** Number of 'while' statements in the function

2.6.2. C# Ruleset

Backward Goto shall not be used

→ **Mnemonic** BWGOTO

→ **Description** Backward gotos shall not be used.

Missing compound statement

→ **Mnemonic** COMPOUND

→ **Description** The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if

→ **Mnemonic** COMPOUNDIF

→ **Description** An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Commented-out Source Code is not allowed

→ **Mnemonic** R_CSTAT

→ **Description** Commented-out Source Code is not allowed

Missing Default

→ **Mnemonic** DEFAULT

→ **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Missing final else

→ **Mnemonic** ELSEFINAL

→ **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Assignment in Boolean

→ **Mnemonic** NOASGCOND

→ **Description** Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison

→ **Mnemonic** NOASGINBOOL

→ **Description** Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes

→ **Mnemonic** CAC_CL

→ **Description** Consider classes refactorization

Factorizable Files

→ **Mnemonic** CAC_FI

→ **Description** Consider files refactorization

Factorizable Functions

→ **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

- **Mnemonic** CC_CL
- **Description** There shall be no duplicated classes

Cloned Files

- **Mnemonic** CC_FI
- **Description** There shall be no duplicated files

Cloned Functions

- **Mnemonic** CC_FN
- **Description** There shall be no duplicated functions

Cloned Algorithmic

- **Mnemonic** CFTC_FN
- **Description** There shall be no algorithmic cloning

There shall be a no code before first case

- **Mnemonic** NOCODEBEFORECASE
- **Description** There shall be a no code before the first case of a switch statement.

Continue shall not be used

- **Mnemonic** NOCONT
- **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Fallthrough shall be avoided

- **Mnemonic** NOFALLTHROUGH
- **Description** There shall be no fallthrough the next case in a switch statement.

FIXME shall not be committed in sources code

- **Mnemonic** R_NOFIXME
- **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

- **Mnemonic** NOGOTO
- **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Label out a switch

- **Mnemonic** NOLABEL
- **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Avoid Duplicated Blocks in Function

- **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

→ **Mnemonic** RISKYEMPTY

→ **Description** Risky Empty Statement

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

2.7. Fortran

2.7.1. Fortran Metrics

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Number of arithmetic if

→ **Mnemonic** ARIF

→ **Description** Count number of arithmetic if

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Header Blocks Of Comment

→ **Mnemonic** BHCO

→ **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

→ **Mnemonic** BLAN

→ **Description** Number of blank lines of code in the source file(s).

Brace Lines

→ **Mnemonic** BRAC

→ **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

→ **Mnemonic** BRKL

→ **Description** Number of 'break' statements in loop in the function

Case Blocks

→ **Mnemonic** CABL

→ **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

→ **Mnemonic** CASE

→ **Description** Number of 'case' labels in the function

Cyclomatic Complexity

→ **Mnemonic** CCN

→ **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

→ **Mnemonic** CFT

→ **Description** Number of tokens in the control flow of functions

Call Graph Depth

→ **Mnemonic** CGDM

→ **Description** Maximum depth of the call graph.

Comment Lines

→ **Mnemonic** CLOC

→ **Description** Number of lines of comments in the source file(s).

Continue Statements

→ **Mnemonic** CONT

→ **Description** Number of 'continue' statements in the function

Commented Statements

→ **Mnemonic** CSTAT

→ **Description** Number of Commented Statements.

Minimum Number of Cycles

→ **Mnemonic** CYCL

→ **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Number of declarative statements

→ **Mnemonic** DECL

→ **Description** Count number of declarative statements

Default Statement

→ **Mnemonic** DEFT

→ **Description** Number of 'default' blocks in 'switch' in the function

Distinct Operands

→ **Mnemonic** DOPD

→ **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

→ **Mnemonic** DOWH

→ **Description** Number of 'do...while' statements in the function

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

For Statements

→ **Mnemonic** FOR

→ **Description** Number of 'for' statements in the function

Structures Added

→ **Mnemonic** SADD

→ **Description** Number of control structures added since the previous version.

Structures Modified

→ **Mnemonic** SMOD

→ **Description** Number of control structures modified since the previous version.

Structures Removed

→ **Mnemonic** SREM

→ **Description** Number of control structures removed since the previous version.

Number of Structures

→ **Mnemonic** SSIZ

→ **Description** Number of control structures: iterations, selections, sequences

Goto Statements

→ **Mnemonic** GOTO

→ **Description** Number of 'goto' statements

Header Lines Of Comment

→ **Mnemonic** HCOM

→ **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

→ **Mnemonic** HLOC

→ **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

→ **Mnemonic** ICC

→ **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

→ **Mnemonic** ICFTC

→ **Description** Number of duplicated tokens in control flow of functions

If Statements

→ **Mnemonic** IF

→ **Description** Number of 'if' statements

Line Count

→ **Mnemonic** LC

→ **Description** Number of lines.

Loop Statements

→ **Mnemonic** LOOP

→ **Description** Number of loop statements in the function

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Non-Cyclic Paths

→ **Mnemonic** PATH

→ **Description** Number of non-cyclic paths in the function.

Orelse operators

→ **Mnemonic** OREL

→ **Description** Number of 'orelse' operators

% of parsed tokens

→ **Mnemonic** PARSE

→ **Description** Percent of parsed tokens

Return Statements

→ **Mnemonic** RETURN

→ **Description** Number of 'return' statements in the function

Repeated Code Blocks

→ **Mnemonic** RS

→ **Description** Duplicated blocks in the function

Skipped Lines of Comment code

→ **Mnemonic** SKLC

→ **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

→ **Mnemonic** SLOC

→ **Description** Number of lines of source code in the source file(s).

Executable Statements

→ **Mnemonic** STAT

→ **Description** Total number of executable statements.

Switch Statements

→ **Mnemonic** SWIT

→ **Description** Number of 'switch' statements in the function

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

2.7.2. Fortran Ruleset

Backward Goto shall not be used

→ **Mnemonic** BWGOTO

→ **Description** Backward gotos shall not be used.

Missing Default

→ **Mnemonic** DEFAULT

→ **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Use of continue is deprecated (Fortran)

→ **Mnemonic** NOCONTINUE

→ **Description** The 'continue' statement is deprecated.

Missing final else

→ **Mnemonic** ELSEFINAL

→ **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Incorrect Function Name

→ **Mnemonic** NAMING_FUNCTION

→ **Description** Function name does not fit the convention.

Incorrect Module Name

→ **Mnemonic** NAMING_MODULE

→ **Description** Module name does not fit the convention.

Incorrect Program Name

- **Mnemonic** NAMING_PROGRAM
- **Description** Program name does not fit the convention.

Incorrect Subroutine Name

- **Mnemonic** NAMING_SUBROUTINE
- **Description** Subroutine name does not fit the convention.

Factorizable Classes

- **Mnemonic** CAC_CL
- **Description** Consider classes refactorization

Factorizable Files

- **Mnemonic** CAC_FI
- **Description** Consider files refactorization

Factorizable Functions

- **Mnemonic** CAC_FN
- **Description** Consider functions refactorization

Factorizable Packages

- **Mnemonic** CAC_PKG
- **Description** Consider packages refactorization

Cloned Classes

- **Mnemonic** CC_CL
- **Description** There shall be no duplicated classes

Cloned Files

- **Mnemonic** CC_FI
- **Description** There shall be no duplicated files

Cloned Functions

- **Mnemonic** CC_FN
- **Description** There shall be no duplicated functions

Cloned Algorithmic

- **Mnemonic** CFTC_FN
- **Description** There shall be no algorithmic cloning

Continue shall not be used

- **Mnemonic** NOCONT
- **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

'cycle' shall not be used

- **Mnemonic** NOCYCL
- **Description** The 'cycle' statement shall not be used.

FIXME shall not be committed in sources code

- **Mnemonic** R_NOFIXME
- **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

- **Mnemonic** NOGOTO
- **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Label out a switch

- **Mnemonic** NOLABEL
- **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Avoid Duplicated Blocks in Function

- **Mnemonic** RS_FN
- **Description** There shall be no duplicated parts in functions

'stop' shall not be used

- **Mnemonic** NOSTOP
- **Description** The 'stop' statement shall not be used.

TODO shall not be committed in sources code

- **Mnemonic** R_NOTODO
- **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

- **Mnemonic** ONECASE
- **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

- **Mnemonic** RELAX
- **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

- **Mnemonic** RETURN
- **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Use of SAVE and DATA

- **Mnemonic** SAVE_DATA_USE

→ **Description** A function must not use the SAVE and DATA statements.

Multiple exit

→ **Mnemonic** SGLEXIT

→ **Description** For any iteration statement there shall be at most one 'exit' statement used for loop termination.

2.8. Java

2.8.1. Java Metrics

Constant Data

→ **Mnemonic** ACST

→ **Description** Number of constant data

Number of Attributes

→ **Mnemonic** ANBR

→ **Description** Number of attributes

Number of data without accessibility

→ **Mnemonic** ANON

→ **Description** Number of data without accessibility

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Public Data

→ **Mnemonic** APBL

→ **Description** Number of public data

Protected Data

→ **Mnemonic** APRT

→ **Description** Number of protected data

Private data

→ **Mnemonic** APRV

→ **Description** Number of private data

Assignment Operators

→ **Mnemonic** ASOP

→ **Description** Number of assignment operators used in the source file

Static Data

→ **Mnemonic** ASTA

→ **Description** Number of static data

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Header Blocks Of Comment

→ **Mnemonic** BHCO

→ **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

→ **Mnemonic** BLAN

→ **Description** Number of blank lines of code in the source file(s).

Brace Lines

→ **Mnemonic** BRAC

→ **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

→ **Mnemonic** BRKL

→ **Description** Number of 'break' statements in loop in the function

Break in Switch

→ **Mnemonic** BRKS

→ **Description** Number of 'break' statements in 'switch' in the function

Case Blocks

→ **Mnemonic** CABL

→ **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

→ **Mnemonic** CASE

→ **Description** Number of 'case' labels in the function

Catch Statements

→ **Mnemonic** CATC

→ **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

→ **Mnemonic** CCN

→ **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

→ **Mnemonic** CFT

→ **Description** Number of tokens in the control flow of functions

Call Graph Depth

→ **Mnemonic** CGDM

→ **Description** Maximum depth of the call graph.

Comment Lines

→ **Mnemonic** CLOC

→ **Description** Number of lines of comments in the source file(s).

Continue Statements

→ **Mnemonic** CONT

→ **Description** Number of 'continue' statements in the function

Comparison Operators

→ **Mnemonic** CPOP

→ **Description** Number of comparison operators used in the source file

Commented Statements

→ **Mnemonic** CSTAT

→ **Description** Number of Commented Statements.

Minimum Number of Cycles

→ **Mnemonic** CYCL

→ **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Depth of Descendant Tree

→ **Mnemonic** DDT

→ **Description** Maximum depth of the inheritance tree from the class

Default Statement

→ **Mnemonic** DEFT

→ **Description** Number of 'default' blocks in 'switch' in the function

Depth of Inheritance Tree

→ **Mnemonic** DIT

→ **Description** Maximum depth of the class inheritance tree

Distinct Operands

→ **Mnemonic** DOPD

→ **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

→ **Mnemonic** DOWH

→ **Description** Number of 'do...while' statements in the function

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

For Statements

→ **Mnemonic** FOR

→ **Description** Number of 'for' statements in the function

Structures Added

→ **Mnemonic** SADD

→ **Description** Number of control structures added since the previous version.

Structures Modified

→ **Mnemonic** SMOD

→ **Description** Number of control structures modified since the previous version.

Structures Removed

→ **Mnemonic** SREM

→ **Description** Number of control structures removed since the previous version.

Number of Structures

→ **Mnemonic** SSIZ

→ **Description** Number of control structures: iterations, selections, sequences

Header Lines Of Comment

→ **Mnemonic** HCOM

→ **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

→ **Mnemonic** HLOC

→ **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

→ **Mnemonic** ICC

→ **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

→ **Mnemonic** ICFTC

→ **Description** Number of duplicated tokens in control flow of functions

If Statements

→ **Mnemonic** IF

→ **Description** Number of 'if' statements

Line Count

→ **Mnemonic** LC

→ **Description** Number of lines.

Loop Statements

→ **Mnemonic** LOOP

→ **Description** Number of loop statements in the function

Multiple Inheritance Indicator

→ **Mnemonic** MII

→ **Description** Number of classes from which the class inherits directly

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Methods without Accessibility

→ **Mnemonic** MNON

→ **Description** Number of methods without any accessibility specifier

Public Methods

→ **Mnemonic** MPBL

→ **Description** Number of public methods

Protected Methods

→ **Mnemonic** MPRT

→ **Description** Number of protected methods

Private Methods

→ **Mnemonic** MPRV

→ **Description** Number of private methods

Static Methods

→ **Mnemonic** MSTA

→ **Description** Number of static methods

Number of Ancestors

→ **Mnemonic** NAC

→ **Description** Number of classes from which the class inherits directly or indirectly

Number of Descendants

→ **Mnemonic** NDC

→ **Description** Number of classes which inherit from the class directly or indirectly

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Number Of Children

→ **Mnemonic** NOC

→ **Description** Number of classes which inherit directly from the class

Number of Parameters

→ **Mnemonic** NOP

→ **Description** Number of formal parameters in the function

Non-Cyclic Paths

→ **Mnemonic** PATH

→ **Description** Number of non-cyclic paths in the function.

Or else operators

→ **Mnemonic** OREL

→ **Description** Number of 'or else' operators

Return Statements

→ **Mnemonic** RETURN

→ **Description** Number of 'return' statements in the function

Repeated Code Blocks

→ **Mnemonic** RS

→ **Description** Duplicated blocks in the function

Skipped Lines of Comment code

→ **Mnemonic** SKLC

→ **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

→ **Mnemonic** SLOC

→ **Description** Number of lines of source code in the source file(s).

Special Operators

- **Mnemonic** SPOP
- **Description** Number of special operators used in the source file

Executable Statements

- **Mnemonic** STAT
- **Description** Total number of executable statements.

Switch Statements

- **Mnemonic** SWIT
- **Description** Number of 'switch' statements in the function

Ternary operators

- **Mnemonic** TERN
- **Description** Number of ternary operators i.e. ?:

Throw Statements

- **Mnemonic** THRO
- **Description** Number of 'throw' statements in the function

Operand Occurrences

- **Mnemonic** TOPD
- **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

- **Mnemonic** TOPT
- **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

- **Mnemonic** TRY
- **Description** Number of 'try' statements in the function

Lines Added

- **Mnemonic** LADD
- **Description** Number of lines added since the previous version.

Lines Modified

- **Mnemonic** LMOD
- **Description** Number of lines modified since the previous version.

Lines Removed

- **Mnemonic** LREM
- **Description** Number of lines removed since the previous version.

While Statements

- **Mnemonic** WHIL
- **Description** Number of 'while' statements in the function

2.8.2. Java Ruleset

Missing Break

- **Mnemonic** BRKFINAL
- **Description** An unconditional break statement shall terminate every non-empty switch clause (see [MISRA-C:2004]: RULE 15.2).

Missing compound statement

- **Mnemonic** COMPOUND
- **Description** The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if

- **Mnemonic** COMPOUNDIF
- **Description** An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Commented-out Source Code is not allowed

- **Mnemonic** R_CSTAT
- **Description** Commented-out Source Code is not allowed

Missing Default

- **Mnemonic** DEFAULT
- **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Missing final else

- **Mnemonic** ELSEFINAL
- **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Assignment in Boolean

- **Mnemonic** NOASGCOND
- **Description** Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison

- **Mnemonic** NOASGINBOOL
- **Description** Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes

→ **Mnemonic** CAC_CL

→ **Description** Consider classes refactorization

Factorizable Files

→ **Mnemonic** CAC_FI

→ **Description** Consider files refactorization

Factorizable Functions

→ **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

There shall be a no code before first case

→ **Mnemonic** NOCODEBEFORECASE

→ **Description** There shall be a no code before the first case of a switch statement.

Continue shall not be used

→ **Mnemonic** NOCONT

→ **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Fallthrough shall be avoided

→ **Mnemonic** NOFALLTHROUGH

→ **Description** There shall be no fallthrough the next case in a switch statement.

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

→ **Mnemonic** RISKYEMPTY

→ **Description** Risky Empty Statement

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

2.9. Javascript

2.9.1. Javascript Metrics

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Header Blocks Of Comment

- **Mnemonic** BHCO
- **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

- **Mnemonic** BLAN
- **Description** Number of blank lines of code in the source file(s).

Brace Lines

- **Mnemonic** BRAC
- **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

- **Mnemonic** BRKL
- **Description** Number of 'break' statements in loop in the function

Break in Switch

- **Mnemonic** BRKS
- **Description** Number of 'break' statements in 'switch' in the function

Case Blocks

- **Mnemonic** CABL
- **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

- **Mnemonic** CASE
- **Description** Number of 'case' labels in the function

Catch Statements

- **Mnemonic** CATC
- **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

- **Mnemonic** CCN
- **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

- **Mnemonic** CFT
- **Description** Number of tokens in the control flow of functions

Call Graph Depth

- **Mnemonic** CGDM
- **Description** Maximum depth of the call graph.

Comment Lines

- **Mnemonic** CLOC
- **Description** Number of lines of comments in the source file(s).

Continue Statements

- **Mnemonic** CONT
- **Description** Number of 'continue' statements in the function

Commented Statements

- **Mnemonic** CSTAT
- **Description** Number of Commented Statements.

Minimum Number of Cycles

- **Mnemonic** CYCL
- **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Default Statement

- **Mnemonic** DEFT
- **Description** Number of 'default' blocks in 'switch' in the function

Distinct Operands

- **Mnemonic** DOPD
- **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

- **Mnemonic** DOPT
- **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

- **Mnemonic** DOWH
- **Description** Number of 'do...while' statements in the function

Else Statements

- **Mnemonic** ELSE
- **Description** Number of 'else' statements

Max Nested Functions

- **Mnemonic** FNST
- **Description** Max Nested Functions

For Statements

- **Mnemonic** FOR
- **Description** Number of 'for' statements in the function

Structures Added

- **Mnemonic** SADD
- **Description** Number of control structures added since the previous version.

Structures Modified

- **Mnemonic** SMOD
- **Description** Number of control structures modified since the previous version.

Structures Removed

- **Mnemonic** SREM
- **Description** Number of control structures removed since the previous version.

Number of Structures

- **Mnemonic** SSIZ
- **Description** Number of control structures: iterations, selections, sequences

Header Lines Of Comment

- **Mnemonic** HCOM
- **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

- **Mnemonic** HLOC
- **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

- **Mnemonic** ICC
- **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

- **Mnemonic** ICFTC
- **Description** Number of duplicated tokens in control flow of functions

If Statements

- **Mnemonic** IF
- **Description** Number of 'if' statements

Line Count

- **Mnemonic** LC
- **Description** Number of lines.

Loop Statements

- **Mnemonic** LOOP
- **Description** Number of loop statements in the function

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Number of Parameters

→ **Mnemonic** NOP

→ **Description** Number of formal parameters in the function

Non-Cyclic Paths

→ **Mnemonic** PATH

→ **Description** Number of non-cyclic paths in the function.

Orelse operators

→ **Mnemonic** OREL

→ **Description** Number of 'orelse' operators

Return Statements

→ **Mnemonic** RETURN

→ **Description** Number of 'return' statements in the function

Repeated Code Blocks

→ **Mnemonic** RS

→ **Description** Duplicated blocks in the function

Skipped Lines of Comment code

→ **Mnemonic** SKLC

→ **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

→ **Mnemonic** SLOC

→ **Description** Number of lines of source code in the source file(s).

Executable Statements

→ **Mnemonic** STAT

→ **Description** Total number of executable statements.

Switch Statements

→ **Mnemonic** SWIT

→ **Description** Number of 'switch' statements in the function

Ternary operators

→ **Mnemonic** TERN

→ **Description** Number of ternary operators i.e. ?:

Throw Statements

→ **Mnemonic** THRO

→ **Description** Number of 'throw' statements in the function

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

→ **Mnemonic** TRY

→ **Description** Number of 'try' statements in the function

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

While Statements

→ **Mnemonic** WHIL

→ **Description** Number of 'while' statements in the function

2.9.2. Javascript Ruleset

Missing Break

→ **Mnemonic** BRKFINAL

→ **Description** An unconditional break statement shall terminate every non-empty switch clause (see [MISRA-C:2004]: RULE 15.2).

Missing compound statement

→ **Mnemonic** COMPOUND

→ **Description** The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if

→ **Mnemonic** COMPOUNDIF

→ **Description** An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Commented-out Source Code is not allowed

→ **Mnemonic** R_CSTAT

→ **Description** Commented-out Source Code is not allowed

Missing Default

→ **Mnemonic** DEFAULT

→ **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Missing final else

→ **Mnemonic** ELSEFINAL

→ **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Assignment in Boolean

→ **Mnemonic** NOASGCOND

→ **Description** Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison

→ **Mnemonic** NOASGINBOOL

→ **Description** Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes

→ **Mnemonic** CAC_CL

→ **Description** Consider classes refactorization

Factorizable Files

→ **Mnemonic** CAC_FI

→ **Description** Consider files refactorization

Factorizable Functions

→ **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

- **Mnemonic** CAC_PKG
- **Description** Consider packages refactorization

Cloned Classes

- **Mnemonic** CC_CL
- **Description** There shall be no duplicated classes

Cloned Files

- **Mnemonic** CC_FI
- **Description** There shall be no duplicated files

Cloned Functions

- **Mnemonic** CC_FN
- **Description** There shall be no duplicated functions

Cloned Algorithmic

- **Mnemonic** CFTC_FN
- **Description** There shall be no algorithmic cloning

There shall be a no code before first case

- **Mnemonic** NOCODEBEFORECASE
- **Description** There shall be a no code before the first case of a switch statement.

Continue shall not be used

- **Mnemonic** NOCONT
- **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Fallthrough shall be avoided

- **Mnemonic** NOFALLTHROUGH
- **Description** There shall be no fallthrough the next case in a switch statement.

FIXME shall not be committed in sources code

- **Mnemonic** R_NOFIXME
- **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

Label out a switch

- **Mnemonic** NOLABEL
- **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Avoid Duplicated Blocks in Function

- **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

→ **Mnemonic** RISKYEMPTY

→ **Description** Risky Empty Statement

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

2.10. MindC

2.10.1. MindC Metrics

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Assignment Operators

→ **Mnemonic** ASOP

→ **Description** Number of assignment operators used in the source file

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Header Blocks Of Comment

→ **Mnemonic** BHCO

→ **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

→ **Mnemonic** BLAN

→ **Description** Number of blank lines of code in the source file(s).

Brace Lines

→ **Mnemonic** BRAC

→ **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

→ **Mnemonic** BRKL

→ **Description** Number of 'break' statements in loop in the function

Break in Switch

→ **Mnemonic** BRKS

→ **Description** Number of 'break' statements in 'switch' in the function

Case Blocks

→ **Mnemonic** CABL

→ **Description** Number of 'case' blocks in 'switch' in the function

Calls To

→ **Mnemonic** CAL2

→ **Description** Number of explicit calls to the function.

Called Functions

→ **Mnemonic** CALD

→ **Description** Number of distinct functions defined in the project source file and called by the function.

Calls From

→ **Mnemonic** CALF

→ **Description** Number of explicit calls from the function.

Calling Functions

→ **Mnemonic** CALI

→ **Description** Number of distinct functions calling the function.

Called External Functions

→ **Mnemonic** CALX

→ **Description** Number of distinct external functions called by the function - external i.e. not defined in the project

Case Labels

→ **Mnemonic** CASE

→ **Description** Number of 'case' labels in the function

Cyclomatic Complexity

→ **Mnemonic** CCN

→ **Description** Number of linearly independent paths in the function control graph.

Recursive Calls

→ **Mnemonic** CDRI

→ **Description** Number of directly recursive calls in the function.

Control Flow Token

→ **Mnemonic** CFT

→ **Description** Number of tokens in the control flow of functions

Called Depth

→ **Mnemonic** CGDD

→ **Description** Maximum depth of called functions.

Calling Depth

→ **Mnemonic** CGDI

→ **Description** Maximum depth of calling functions.

Call Graph Depth

→ **Mnemonic** CGDM

→ **Description** Maximum depth of the call graph.

Minimum Number of Indirect Cycles

→ **Mnemonic** CIRI

→ **Description** Minimum number of indirect call graph cycles in which the function is involved (excluding recursive calls).

Comment Lines

→ **Mnemonic** CLOC

→ **Description** Number of lines of comments in the source file(s).

Continue Statements

→ **Mnemonic** CONT

→ **Description** Number of 'continue' statements in the function

Comparison Operators

→ **Mnemonic** CPOP

→ **Description** Number of comparison operators used in the source file

Commented Statements

→ **Mnemonic** CSTAT

→ **Description** Number of Commented Statements.

Minimum Number of Cycles

→ **Mnemonic** CYCL

→ **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Default Statement

→ **Mnemonic** DEFT

→ **Description** Number of 'default' blocks in 'switch' in the function

Distinct Operands

→ **Mnemonic** DOPD

→ **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

→ **Mnemonic** DOWH

→ **Description** Number of 'do...while' statements in the function

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

For Statements

→ **Mnemonic** FOR

→ **Description** Number of 'for' statements in the function

Structures Added

→ **Mnemonic** SADD

→ **Description** Number of control structures added since the previous version.

Structures Modified

→ **Mnemonic** SMOD

→ **Description** Number of control structures modified since the previous version.

Structures Removed

→ **Mnemonic** SREM

→ **Description** Number of control structures removed since the previous version.

Number of Structures

→ **Mnemonic** SSIZ

→ **Description** Number of control structures: iterations, selections, sequences

Goto Statements

→ **Mnemonic** GOTO

→ **Description** Number of 'goto' statements

Header Lines Of Comment

→ **Mnemonic** HCOM

→ **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

→ **Mnemonic** HLOC

→ **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

→ **Mnemonic** ICC

→ **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

→ **Mnemonic** ICFTC

→ **Description** Number of duplicated tokens in control flow of functions

If Statements

→ **Mnemonic** IF

→ **Description** Number of 'if' statements

Line Count

→ **Mnemonic** LC

→ **Description** Number of lines.

Use of longjump

→ **Mnemonic** LONGJMP

→ **Description** Use of longjump

Loop Statements

→ **Mnemonic** LOOP

→ **Description** Number of loop statements in the function

Memory Allocation

→ **Mnemonic** MEMALLOC

→ **Description** Memory Allocation

Memory Freeing

→ **Mnemonic** MEMFREE

→ **Description** Memory Freeing

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Number of Parameters

→ **Mnemonic** NOP

→ **Description** Number of formal parameters in the function

Non-Cyclic Paths

→ **Mnemonic** PATH

→ **Description** Number of non-cyclic paths in the function.

Use of offsetof

→ **Mnemonic** OFFSETOF

→ **Description** Use of offsetof

Or else operators

→ **Mnemonic** OREL

→ **Description** Number of 'or else' operators

Number of #DEFINE

→ **Mnemonic** P_DEFINE

→ **Description** Number of #DEFINE

Number of #ELIF

→ **Mnemonic** P_ELIF

→ **Description** Number of #ELIF

Number of #ELSE

→ **Mnemonic** P_ELSE

→ **Description** Number of #ELSE

Number of #ENDIF

→ **Mnemonic** P_ENDIF

→ **Description** Number of #ENDIF

Number of #ERROR

→ **Mnemonic** P_ERROR

→ **Description** Number of #ERROR

Number of #IF

→ **Mnemonic** P_IF

→ **Description** Number of #IF

Number of #IFDEF

→ **Mnemonic** P_IFDEF

→ **Description** Number of #IFDEF

Number of #IFNDEF

→ **Mnemonic** P_IFNDEF

→ **Description** Number of #IFNDEF

Number of Include

→ **Mnemonic** P_INCLUDE

→ **Description** Number of Include

Compiler FLAG Nested Level

→ **Mnemonic** P_NEST

→ **Description** Compiler FLAG Nested Level

Number of #PRAGMA

→ **Mnemonic** P_PRAGMA

→ **Description** Number of #PRAGMA

Number of #UNDEF

→ **Mnemonic** P_UNDEF

→ **Description** Number of #UNDEF

Number of #WARNING

→ **Mnemonic** P_WARNING

→ **Description** Number of #WARNING

Return Statements

→ **Mnemonic** RETURN

→ **Description** Number of 'return' statements in the function

Repeated Code Blocks

→ **Mnemonic** RS

→ **Description** Duplicated blocks in the function

Use of setjump

→ **Mnemonic** SETJMP

→ **Description** Use of setjump

Signal Functions

→ **Mnemonic** SIGNAL

→ **Description** Use of signal Functions

Skipped Lines of Comment code

→ **Mnemonic** SKLC

→ **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

→ **Mnemonic** SLOC

→ **Description** Number of lines of source code in the source file(s).

Special Operators

→ **Mnemonic** SPOP

→ **Description** Number of special operators used in the source file

Executable Statements

→ **Mnemonic** STAT

→ **Description** Total number of executable statements.

IO Functions

→ **Mnemonic** STDIO

→ **Description** Use IO Functions

String Conversions

→ **Mnemonic** STRINGCONV

→ **Description** Use of String Conversions

Switch Statements

→ **Mnemonic** SWIT

→ **Description** Number of 'switch' statements in the function

System Functions

→ **Mnemonic** SYSCOM

→ **Description** Use of system Functions

Ternary operators

→ **Mnemonic** TERN

→ **Description** Number of ternary operators i.e. ?:

Time Handling

→ **Mnemonic** TIMEHDL

→ **Description** Use of Time Handling

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

While Statements

→ **Mnemonic** WHIL

→ **Description** Number of 'while' statements in the function

2.10.2. MindC Ruleset

Missing Break

→ **Mnemonic** BRKFINAL

→ **Description** An unconditional break statement shall terminate every non-empty switch clause (see [MISRA-C:2004]: RULE 15.2).

Backward Goto shall not be used

→ **Mnemonic** BWGOTO

→ **Description** Backward gotos shall not be used.

Missing compound statement

→ **Mnemonic** COMPOUND

→ **Description** The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if

→ **Mnemonic** COMPOUNDIF

- **Description** An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Commented-out Source Code is not allowed

- **Mnemonic** R_CSTAT
- **Description** Commented-out Source Code is not allowed

Missing Default

- **Mnemonic** DEFAULT
- **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Dynamic Memory Allocation shall not be used

- **Mnemonic** DYNMEMALLOC
- **Description** Dynamic heap memory allocation shall not be used. This precludes the use of the functions calloc, malloc, realloc and free (see [MISRA-C:2004]: RULE 20.4)

Missing final else

- **Mnemonic** ELSEFINAL
- **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Macro longjmp or setjmp shall not be used

- **Mnemonic** JUMP
- **Description** (The setjmp macro and the longjmp function shall not be used (see [MISRA-C:2004]: RULE 20.7).

Nesting Level of Preprocessing directives is too high

- **Mnemonic** R_MAXPNEST
- **Description** Nesting Level of Preprocessing directives is too high

Assignment in Boolean

- **Mnemonic** NOASGCOND
- **Description** Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison

- **Mnemonic** NOASGINBOOL
- **Description** Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes

- **Mnemonic** CAC_CL
- **Description** Consider classes refactorization

Factorizable Files

- **Mnemonic** CAC_FI

→ **Description** Consider files refactorization

Factorizable Functions

→ **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

There shall be a no code before first case

→ **Mnemonic** NOCODEBEFORECASE

→ **Description** There shall be a no code before the first case of a switch statement.

Continue shall not be used

→ **Mnemonic** NOCONT

→ **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Fallthrough shall be avoided

→ **Mnemonic** NOFALLTHROUGH

→ **Description** There shall be no fallthrough the next case in a switch statement.

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

→ **Mnemonic** NOGOTO

→ **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Label out a switch

→ **Mnemonic** NOLABEL

→ **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Recursion are not allowed

→ **Mnemonic** NORECURSION

→ **Description** Functions shall not called themselves either directly or indirectly (see [MISRA-C:2004]: RULE 16.2).

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Macro offsetof shall not be used

→ **Mnemonic** OFFSETOF

→ **Description** The macro offsetof, in library <stddef.h>, shall not be used (see [MISRA-C:2004]: RULE 20.6).

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

→ **Mnemonic** RISKYEMPTY

→ **Description** Risky Empty Statement

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

Signal or Raise shall not be used

→ **Mnemonic** SIGNAL

→ **Description** The signal handling facilities of <signal.h> shall not be used (see [MISRA-C:2004]: RULE 20.8).

IO Functions shall not be used

→ **Mnemonic** STDIO

→ **Description** The input/output library <stdio.h> shall not be used in production code (see [MISRA-C:2004]: RULE 20.9).

'atof, atoi or atol' shall not be used

→ **Mnemonic** STRINGCONV

→ **Description** The library functions atof, atoi and atol from library <stdlib.h> shall not be used (see [MISRA-C:2004]: RULE 20.10).

'abort, exit, getenv or system' shall not be used

→ **Mnemonic** SYSCOM

→ **Description** The library functions abort, exit, getenv and system from library <stdlib.h> shall not be used (see [MISRA-C:2004]: RULE 20.11).

Time Handling Functions shall not be used

→ **Mnemonic** TIMEHDL

→ **Description** The time handling functions of library <time.h> shall not be used: time, strftime, clock, difftime, mktime (see [MISRA-C:2004]: RULE 20.12).

2.11. Objective-C

2.11.1. Objective-C Metrics

Constant Data

→ **Mnemonic** ACST

→ **Description** Number of constant data

Number of Attributes

→ **Mnemonic** ANBR

→ **Description** Number of attributes

Number of data without accessibility

→ **Mnemonic** ANON

→ **Description** Number of data without accessibility

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Public Data

- **Mnemonic** APBL
- **Description** Number of public data

Protected Data

- **Mnemonic** APRT
- **Description** Number of protected data

Private data

- **Mnemonic** APRV
- **Description** Number of private data

Assignment Operators

- **Mnemonic** ASOP
- **Description** Number of assignment operators used in the source file

Static Data

- **Mnemonic** ASTA
- **Description** Number of static data

Number of comment blocks

- **Mnemonic** BCOM
- **Description** Number of comment blocks.

Header Blocks Of Comment

- **Mnemonic** BHCO
- **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

- **Mnemonic** BLAN
- **Description** Number of blank lines of code in the source file(s).

Brace Lines

- **Mnemonic** BRAC
- **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

- **Mnemonic** BRKL
- **Description** Number of 'break' statements in loop in the function

Break in Switch

- **Mnemonic** BRKS
- **Description** Number of 'break' statements in 'switch' in the function

Case Blocks

- **Mnemonic** CABL
- **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

- **Mnemonic** CASE
- **Description** Number of 'case' labels in the function

Catch Statements

- **Mnemonic** CATC
- **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

- **Mnemonic** CCN
- **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

- **Mnemonic** CFT
- **Description** Number of tokens in the control flow of functions

Call Graph Depth

- **Mnemonic** CGDM
- **Description** Maximum depth of the call graph.

Comment Lines

- **Mnemonic** CLOC
- **Description** Number of lines of comments in the source file(s).

Continue Statements

- **Mnemonic** CONT
- **Description** Number of 'continue' statements in the function

Comparison Operators

- **Mnemonic** CPOP
- **Description** Number of comparison operators used in the source file

Commented Statements

- **Mnemonic** CSTAT
- **Description** Number of Commented Statements.

Minimum Number of Cycles

- **Mnemonic** CYCL
- **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Depth of Descendant Tree

→ **Mnemonic** DDT

→ **Description** Maximun depth of the inheritance tree from the class

Default Statement

→ **Mnemonic** DEFT

→ **Description** Number of 'default' blocks in 'switch' in the function

Depth of Inheritance Tree

→ **Mnemonic** DIT

→ **Description** Maximun depth of the class inheritance tree

Distinct Operands

→ **Mnemonic** DOPD

→ **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

→ **Mnemonic** DOWH

→ **Description** Number of 'do...while' statements in the function

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

For Statements

→ **Mnemonic** FOR

→ **Description** Number of 'for' statements in the function

Structures Added

→ **Mnemonic** SADD

→ **Description** Number of control structures added since the previous version.

Structures Modified

→ **Mnemonic** SMOD

→ **Description** Number of control structures modified since the previous version.

Structures Removed

→ **Mnemonic** SREM

→ **Description** Number of control structures removed since the previous version.

Number of Structures

- **Mnemonic** SSIZ
- **Description** Number of control structures: iterations, selections, sequences

Goto Statements

- **Mnemonic** GOTO
- **Description** Number of 'goto' statements

Header Lines Of Comment

- **Mnemonic** HCOM
- **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

- **Mnemonic** HLOC
- **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

- **Mnemonic** ICC
- **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

- **Mnemonic** ICFTC
- **Description** Number of duplicated tokens in control flow of functions

If Statements

- **Mnemonic** IF
- **Description** Number of 'if' statements

Line Count

- **Mnemonic** LC
- **Description** Number of lines.

Loop Statements

- **Mnemonic** LOOP
- **Description** Number of loop statements in the function

Multiple Inheritance Indicator

- **Mnemonic** MII
- **Description** Number of classes from which the class inherits directly

Mixed Lines

- **Mnemonic** MLOC
- **Description** Number of lines containing both code and comment in the source files.

Methods without Accessibility

→ **Mnemonic** MNON

→ **Description** Number of methods without any accessibility specifier

Public Methods

→ **Mnemonic** MPBL

→ **Description** Number of public methods

Protected Methods

→ **Mnemonic** MPRT

→ **Description** Number of protected methods

Private Methods

→ **Mnemonic** MPRV

→ **Description** Number of private methods

Static Methods

→ **Mnemonic** MSTA

→ **Description** Number of static methods

Number of Ancestors

→ **Mnemonic** NAC

→ **Description** Number of classes from which the class inherits directly or indirectly

Number of Descendants

→ **Mnemonic** NDC

→ **Description** Number of classes which inherit from the class directly or indirectly

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Number Of Children

→ **Mnemonic** NOC

→ **Description** Number of classes which inherit directly from the class

Number of Parameters

→ **Mnemonic** NOP

→ **Description** Number of formal parameters in the function

Non-Cyclic Paths

→ **Mnemonic** PATH

→ **Description** Number of non-cyclic paths in the function.

Orelse operators

- **Mnemonic** OREL
- **Description** Number of 'orelse' operators

Properties

- **Mnemonic** PNBR
- **Description** Total number of properties

Number of #DEFINE

- **Mnemonic** P_DEFINE
- **Description** Number of #DEFINE

Number of #ELIF

- **Mnemonic** P_ELIF
- **Description** Number of #ELIF

Number of #ELSE

- **Mnemonic** P_ELSE
- **Description** Number of #ELSE

Number of #ENDIF

- **Mnemonic** P_ENDIF
- **Description** Number of #ENDIF

Number of #ERROR

- **Mnemonic** P_ERROR
- **Description** Number of #ERROR

Number of #IF

- **Mnemonic** P_IF
- **Description** Number of #IF

Number of #IFDEF

- **Mnemonic** P_IFDEF
- **Description** Number of #IFDEF

Number of #IFDEF

- **Mnemonic** P_IFNDEF
- **Description** Number of #IFDEF

Number of Include

- **Mnemonic** P_INCLUDE
- **Description** Number of Include

Compiler FLAG Nested Level

- **Mnemonic** P_NEST
- **Description** Compiler FLAG Nested Level

Number of #PRAGMA

- **Mnemonic** P_PRAGMA
- **Description** Number of #PRAGMA

Number of #UNDEF

- **Mnemonic** P_UNDEF
- **Description** Number of #UNDEF

Number of #WARNING

- **Mnemonic** P_WARNING
- **Description** Number of #WARNING

Return Statements

- **Mnemonic** RETURN
- **Description** Number of 'return' statements in the function

Repeated Code Blocks

- **Mnemonic** RS
- **Description** Duplicated blocks in the function

Skipped Lines of Comment code

- **Mnemonic** SKLC
- **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

- **Mnemonic** SLOC
- **Description** Number of lines of source code in the source file(s).

Special Operators

- **Mnemonic** SPOP
- **Description** Number of special operators used in the source file

Executable Statements

- **Mnemonic** STAT
- **Description** Total number of executable statements.

Switch Statements

- **Mnemonic** SWIT

→ **Description** Number of 'switch' statements in the function

Ternary operators

→ **Mnemonic** TERN

→ **Description** Number of ternary operators i.e. ?:

Throw Statements

→ **Mnemonic** THRO

→ **Description** Number of 'throw' statements in the function

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

→ **Mnemonic** TRY

→ **Description** Number of 'try' statements in the function

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

While Statements

→ **Mnemonic** WHIL

→ **Description** Number of 'while' statements in the function

Weighted Method per Class

→ **Mnemonic** XWMC

→ **Description** Sum of cyclomatic complexities of methods implemented outside the class definition

2.11.2. Objective-C Ruleset

Missing Break

→ **Mnemonic** BRKFINAL

→ **Description** An unconditional break statement shall terminate every non-empty switch clause (see [MISRA-C:2004]: RULE 15.2).

Backward Goto shall not be used

→ **Mnemonic** BWGOTO

→ **Description** Backward gotos shall not be used.

Missing compound statement

→ **Mnemonic** COMPOUND

→ **Description** The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if

→ **Mnemonic** COMPOUNDIF

→ **Description** An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Commented-out Source Code is not allowed

→ **Mnemonic** R_CSTAT

→ **Description** Commented-out Source Code is not allowed

Missing Default

→ **Mnemonic** DEFAULT

→ **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Missing final else

→ **Mnemonic** ELSEFINAL

→ **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Assignment in Boolean

→ **Mnemonic** NOASGCOND

→ **Description** Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison

→ **Mnemonic** NOASGINBOOL

→ **Description** Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes

→ **Mnemonic** CAC_CL

→ **Description** Consider classes refactorization

Factorizable Files

→ **Mnemonic** CAC_FI

→ **Description** Consider files refactorization

Factorizable Functions

→ **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

There shall be a no code before first case

→ **Mnemonic** NOCODEBEFORECASE

→ **Description** There shall be a no code before the first case of a switch statement.

Continue shall not be used

→ **Mnemonic** NOCONT

→ **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Fallthrough shall be avoided

→ **Mnemonic** NOFALLTHROUGH

→ **Description** There shall be no fallthrough the next case in a switch statement.

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

→ **Mnemonic** NOGOTO

→ **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Label out a switch

→ **Mnemonic** NOLABEL

→ **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

→ **Mnemonic** RISKYEMPTY

→ **Description** Risky Empty Statement

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

2.12. PHP

2.12.1. PHP Metrics

Constant Data

→ **Mnemonic** ACST

→ **Description** Number of constant data

Number of Attributes

- **Mnemonic** ANBR
- **Description** Number of attributes

Number of data without accessibility

- **Mnemonic** ANON
- **Description** Number of data without accessibility

Andthen Operators

- **Mnemonic** ANTH
- **Description** Number of 'andthen' operators

Public Data

- **Mnemonic** APBL
- **Description** Number of public data

Protected Data

- **Mnemonic** APRT
- **Description** Number of protected data

Private data

- **Mnemonic** APRV
- **Description** Number of private data

Static Data

- **Mnemonic** ASTA
- **Description** Number of static data

Number of comment blocks

- **Mnemonic** BCOM
- **Description** Number of comment blocks.

Header Blocks Of Comment

- **Mnemonic** BHCO
- **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

- **Mnemonic** BLAN
- **Description** Number of blank lines of code in the source file(s).

Brace Lines

- **Mnemonic** BRAC
- **Description** Number of lines of code containing only a brace in the source file(s).

Case Labels

- **Mnemonic** CASE
- **Description** Number of 'case' labels in the function

Catch Statements

- **Mnemonic** CATC
- **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

- **Mnemonic** CCN
- **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

- **Mnemonic** CFT
- **Description** Number of tokens in the control flow of functions

Call Graph Depth

- **Mnemonic** CGDM
- **Description** Maximum depth of the call graph.

Comment Lines

- **Mnemonic** CLOC
- **Description** Number of lines of comments in the source file(s).

Continue Statements

- **Mnemonic** CONT
- **Description** Number of 'continue' statements in the function

Commented Statements

- **Mnemonic** CSTAT
- **Description** Number of Commented Statements.

Minimum Number of Cycles

- **Mnemonic** CYCL
- **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Depth of Descendant Tree

- **Mnemonic** DDT
- **Description** Maximum depth of the inheritance tree from the class

Default Statement

- **Mnemonic** DEFT
- **Description** Number of 'default' blocks in 'switch' in the function

Depth of Inheritance Tree

- **Mnemonic** DIT
- **Description** Maximun depth of the class inheritance tree

Distinct Operands

- **Mnemonic** DOPD
- **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

- **Mnemonic** DOPT
- **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

- **Mnemonic** DOWH
- **Description** Number of 'do...while' statements in the function

Else Statements

- **Mnemonic** ELSE
- **Description** Number of 'else' statements

Call to exit

- **Mnemonic** EXIT
- **Description** Number of calls to the exit function

For Statements

- **Mnemonic** FOR
- **Description** Number of 'for' statements in the function

Foreach Statements

- **Mnemonic** FORE
- **Description** Number of 'foreach' statements in the function

Structures Added

- **Mnemonic** SADD
- **Description** Number of control structures added since the previous version.

Structures Modified

- **Mnemonic** SMOD
- **Description** Number of control structures modified since the previous version.

Structures Removed

- **Mnemonic** SREM
- **Description** Number of control structures removed since the previous version.

Number of Structures

- **Mnemonic** SSIZ
- **Description** Number of control structures: iterations, selections, sequences

Goto Statements

- **Mnemonic** GOTO
- **Description** Number of 'goto' statements

Header Lines Of Comment

- **Mnemonic** HCOM
- **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

- **Mnemonic** HLOC
- **Description** Number of lines between the function or class definition and the first opening brace.

HTML Lines of Code

- **Mnemonic** HTML
- **Description** Number of HTML lines of code in the source file(s).

Cloned Code

- **Mnemonic** ICC
- **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

- **Mnemonic** ICFTC
- **Description** Number of duplicated tokens in control flow of functions

If Statements

- **Mnemonic** IF
- **Description** Number of 'if' statements

Line Count

- **Mnemonic** LC
- **Description** Number of lines.

Loop Statements

- **Mnemonic** LOOP
- **Description** Number of loop statements in the function

Constant Methods

- **Mnemonic** MCST
- **Description** Number of 'constant' methods i.e. which do not modify the object

Multiple Inheritance Indicator

→ **Mnemonic** MII

→ **Description** Number of classes from which the class inherits directly

PHP/HTML Mixed Lines

→ **Mnemonic** MIXL

→ **Description** Number of lines containing both PHP and HTML in the source files.

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Methods without Accessibility

→ **Mnemonic** MNON

→ **Description** Number of methods without any accessibility specifier

Public Methods

→ **Mnemonic** MPBL

→ **Description** Number of public methods

Protected Methods

→ **Mnemonic** MPRT

→ **Description** Number of protected methods

Private Methods

→ **Mnemonic** MPRV

→ **Description** Number of private methods

Static Methods

→ **Mnemonic** MSTA

→ **Description** Number of static methods

Number of Ancestors

→ **Mnemonic** NAC

→ **Description** Number of classes from which the class inherits directly or indirectly

Number of Descendants

→ **Mnemonic** NDC

→ **Description** Number of classes which inherit from the class directly or indirectly

Maximum Nested Structures

→ **Mnemonic** NEST

→ **Description** Maximum number of nested structures

Number Of Children

- **Mnemonic** NOC
- **Description** Number of classes which inherit directly from the class

Non-Cyclic Paths

- **Mnemonic** PATH
- **Description** Number of non-cyclic paths in the function.

Or else operators

- **Mnemonic** OREL
- **Description** Number of 'or else' operators

PHP Lines of Code

- **Mnemonic** PHPL
- **Description** Number of PHP lines of code in the source file(s).

Return Statements

- **Mnemonic** RETURN
- **Description** Number of 'return' statements in the function

Repeated Code Blocks

- **Mnemonic** RS
- **Description** Duplicated blocks in the function

Skipped Lines of Comment code

- **Mnemonic** SKLC
- **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

- **Mnemonic** SLOC
- **Description** Number of lines of source code in the source file(s).

Executable Statements

- **Mnemonic** STAT
- **Description** Total number of executable statements.

Switch Statements

- **Mnemonic** SWIT
- **Description** Number of 'switch' statements in the function

Ternary operators

- **Mnemonic** TERN
- **Description** Number of ternary operators i.e. ?:

Throw Statements

- **Mnemonic** THRO
- **Description** Number of 'throw' statements in the function

Operand Occurrences

- **Mnemonic** TOPD
- **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

- **Mnemonic** TOPT
- **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

- **Mnemonic** TRY
- **Description** Number of 'try' statements in the function

Lines Added

- **Mnemonic** LADD
- **Description** Number of lines added since the previous version.

Lines Modified

- **Mnemonic** LMOD
- **Description** Number of lines modified since the previous version.

Lines Removed

- **Mnemonic** LREM
- **Description** Number of lines removed since the previous version.

While Statements

- **Mnemonic** WHIL
- **Description** Number of 'while' statements in the function

2.12.2. PHP Ruleset

Missing Break

- **Mnemonic** BRKFINAL
- **Description** An unconditional break statement shall terminate every non-empty switch clause (see [MISRA-C:2004]: RULE 15.2).

Backward Goto shall not be used

- **Mnemonic** BWGOTO
- **Description** Backward gotos shall not be used.

Missing compound statement

→ Mnemonic COMPOUND

→ Description The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if**→ Mnemonic COMPOUNDIF**

→ Description An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Commented-out Source Code is not allowed**→ Mnemonic R_CSTAT**

→ Description Commented-out Source Code is not allowed

Missing Default**→ Mnemonic DEFAULT**

→ Description The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Missing final else**→ Mnemonic ELSEFINAL**

→ Description All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Assignment in Boolean**→ Mnemonic NOASGCOND**

→ Description Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison**→ Mnemonic NOASGINBOOL**

→ Description Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes**→ Mnemonic CAC_CL**

→ Description Consider classes refactorization

Factorizable Files**→ Mnemonic CAC_FI**

→ Description Consider files refactorization

Factorizable Functions**→ Mnemonic CAC_FN**

→ Description Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

There shall be a no code before first case

→ **Mnemonic** NOCODEBEFORECASE

→ **Description** There shall be a no code before the first case of a switch statement.

Continue shall not be used

→ **Mnemonic** NOCONT

→ **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Fallthrough shall be avoided

→ **Mnemonic** NOFALLTHROUGH

→ **Description** There shall be no fallthrough the next case in a switch statement.

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

→ **Mnemonic** NOGOTO

→ **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Label out a switch

→ **Mnemonic** NOLABEL

→ **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

→ **Mnemonic** RISKYEMPTY

→ **Description** Risky Empty Statement

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

2.13. Python

2.13.1. Python Metrics

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Header Blocks Of Comment

→ **Mnemonic** BHCO

→ **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

→ **Mnemonic** BLAN

→ **Description** Number of blank lines of code in the source file(s).

Brace Lines

→ **Mnemonic** BRAC

→ **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

→ **Mnemonic** BRKL

→ **Description** Number of 'break' statements in loop in the function

Catch Statements

→ **Mnemonic** CATC

→ **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

→ **Mnemonic** CCN

→ **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

→ **Mnemonic** CFT

→ **Description** Number of tokens in the control flow of functions

Call Graph Depth

→ **Mnemonic** CGDM

→ **Description** Maximum depth of the call graph.

Comment Lines

→ **Mnemonic** CLOC

→ **Description** Number of lines of comments in the source file(s).

Continue Statements

→ **Mnemonic** CONT

→ **Description** Number of 'continue' statements in the function

Commented Statements

→ **Mnemonic** CSTAT

→ **Description** Number of Commented Statements.

Minimum Number of Cycles

→ **Mnemonic** CYCL

→ **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Depth of Descendant Tree

→ **Mnemonic** DDT

→ **Description** Maximum depth of the inheritance tree from the class

Depth of Inheritance Tree

- **Mnemonic** DIT
- **Description** Maximun depth of the class inheritance tree

Number of DocString lines

- **Mnemonic** DOCL
- **Description** Count number of lines of python DocString

Distinct Operands

- **Mnemonic** DOPD
- **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

- **Mnemonic** DOPT
- **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Else Statements

- **Mnemonic** ELSE
- **Description** Number of 'else' statements

Call to exit

- **Mnemonic** EXIT
- **Description** Number of calls to the exit function

For Statements

- **Mnemonic** FOR
- **Description** Number of 'for' statements in the function

Structures Added

- **Mnemonic** SADD
- **Description** Number of control structures added since the previous version.

Structures Modified

- **Mnemonic** SMOD
- **Description** Number of control structures modified since the previous version.

Structures Removed

- **Mnemonic** SREM
- **Description** Number of control structures removed since the previous version.

Number of Structures

- **Mnemonic** SSIZ
- **Description** Number of control structures: iterations, selections, sequences

Header Lines Of Comment

→ **Mnemonic** HCOM

→ **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

→ **Mnemonic** HLOC

→ **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

→ **Mnemonic** ICC

→ **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

→ **Mnemonic** ICFTC

→ **Description** Number of duplicated tokens in control flow of functions

If Statements

→ **Mnemonic** IF

→ **Description** Number of 'if' statements

Line Count

→ **Mnemonic** LC

→ **Description** Number of lines.

Loop Statements

→ **Mnemonic** LOOP

→ **Description** Number of loop statements in the function

Multiple Inheritance Indicator

→ **Mnemonic** MII

→ **Description** Number of classes from which the class inherits directly

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Number of Ancestors

→ **Mnemonic** NAC

→ **Description** Number of classes from which the class inherits directly or indirectly

Number of Descendants

→ **Mnemonic** NDC

→ **Description** Number of classes which inherit from the class directly or indirectly

Maximum Nested Structures

- **Mnemonic** NEST
- **Description** Maximum number of nested structures

Number Of Children

- **Mnemonic** NOC
- **Description** Number of classes which inherit directly from the class

Non-Cyclic Paths

- **Mnemonic** PATH
- **Description** Number of non-cyclic paths in the function.

% of parsed tokens

- **Mnemonic** PARSE
- **Description** Percent of parsed tokens

Return Statements

- **Mnemonic** RETURN
- **Description** Number of 'return' statements in the function

Repeated Code Blocks

- **Mnemonic** RS
- **Description** Duplicated blocks in the function

Skipped Lines of Comment code

- **Mnemonic** SKLC
- **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

- **Mnemonic** SLOC
- **Description** Number of lines of source code in the source file(s).

Executable Statements

- **Mnemonic** STAT
- **Description** Total number of executable statements.

Throw Statements

- **Mnemonic** THRO
- **Description** Number of 'throw' statements in the function

Operand Occurrences

- **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

→ **Mnemonic** TRY

→ **Description** Number of 'try' statements in the function

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

While Statements

→ **Mnemonic** WHIL

→ **Description** Number of 'while' statements in the function

2.13.2. Python Ruleset

There shall be a `__init__` method in the class.

→ **Mnemonic** CLASSNOINIT

→ **Description** There shall be a `__init__` method in the class.

Missing compound statement

→ **Mnemonic** COMPOUND

→ **Description** The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if

→ **Mnemonic** COMPOUNDIF

→ **Description** An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Missing final else

→ **Mnemonic** ELSEFINAL

- **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Method should have "self" as first argument

- **Mnemonic** METHODSELFFIRST
- **Description** Method has an attribute different the "self" as first argument.

Method without parameter

- **Mnemonic** METHODWITHOUTPARAM
- **Description** Method without parameter.

Assignment in Boolean

- **Mnemonic** NOASGCOND
- **Description** Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison

- **Mnemonic** NOASGINBOOL
- **Description** Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes

- **Mnemonic** CAC_CL
- **Description** Consider classes refactorization

Factorizable Files

- **Mnemonic** CAC_FI
- **Description** Consider files refactorization

Factorizable Functions

- **Mnemonic** CAC_FN
- **Description** Consider functions refactorization

Factorizable Packages

- **Mnemonic** CAC_PKG
- **Description** Consider packages refactorization

Cloned Classes

- **Mnemonic** CC_CL
- **Description** There shall be no duplicated classes

Cloned Files

- **Mnemonic** CC_FI
- **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

Continue shall not be used

→ **Mnemonic** NOCONT

→ **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Exec shall not be used.

→ **Mnemonic** NOEXEC

→ **Description** Use of 'exec'

EXIT PROGRAM shall not be used

→ **Mnemonic** NOEXIT

→ **Description** EXIT PROGRAM shall not be used in a subprogram. Use GOBACK instead

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

Label out a switch

→ **Mnemonic** NOLABEL

→ **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Print shall not be used.

→ **Mnemonic** NOPRINT

→ **Description** Use of 'print'

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

'star' parameter shall not be used.

→ **Mnemonic** NOSTARPARAM

→ **Description** Use of star parameter

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

There shall be only one Statement per line

→ **Mnemonic** ONESTMTPERLINE

→ **Description** There shall be only one Statement per line

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

→ **Mnemonic** RISKYEMPTY

→ **Description** Risky Empty Statement

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

2.14. PL/SQL

2.14.1. PL/SQL Metrics

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Header Blocks Of Comment

→ **Mnemonic** BHCO

→ **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

→ **Mnemonic** BLAN

→ **Description** Number of blank lines of code in the source file(s).

Brace Lines

→ **Mnemonic** BRAC

→ **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

- **Mnemonic** BRKL
- **Description** Number of 'break' statements in loop in the function

Case Blocks

- **Mnemonic** CABL
- **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

- **Mnemonic** CASE
- **Description** Number of 'case' labels in the function

Catch Statements

- **Mnemonic** CATC
- **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

- **Mnemonic** CCN
- **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

- **Mnemonic** CFT
- **Description** Number of tokens in the control flow of functions

Call Graph Depth

- **Mnemonic** CGDM
- **Description** Maximum depth of the call graph.

Comment Lines

- **Mnemonic** CLOC
- **Description** Number of lines of comments in the source file(s).

Continue Statements

- **Mnemonic** CONT
- **Description** Number of 'continue' statements in the function

Commented Statements

- **Mnemonic** CSTAT
- **Description** Number of Commented Statements.

Minimum Number of Cycles

- **Mnemonic** CYCL
- **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Default Statement

- **Mnemonic** DEFT
- **Description** Number of 'default' blocks in 'switch' in the function

Distinct Operands

- **Mnemonic** DOPD
- **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

- **Mnemonic** DOPT
- **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Else Statements

- **Mnemonic** ELSE
- **Description** Number of 'else' statements

For Statements

- **Mnemonic** FOR
- **Description** Number of 'for' statements in the function

Structures Added

- **Mnemonic** SADD
- **Description** Number of control structures added since the previous version.

Structures Modified

- **Mnemonic** SMOD
- **Description** Number of control structures modified since the previous version.

Structures Removed

- **Mnemonic** SREM
- **Description** Number of control structures removed since the previous version.

Number of Structures

- **Mnemonic** SSIZ
- **Description** Number of control structures: iterations, selections, sequences

Goto Statements

- **Mnemonic** GOTO
- **Description** Number of 'goto' statements

Header Lines Of Comment

- **Mnemonic** HCOM
- **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

- **Mnemonic** HLOC
- **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

- **Mnemonic** ICC
- **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

- **Mnemonic** ICFTC
- **Description** Number of duplicated tokens in control flow of functions

If Statements

- **Mnemonic** IF
- **Description** Number of 'if' statements

Line Count

- **Mnemonic** LC
- **Description** Number of lines.

Loop Statements

- **Mnemonic** LOOP
- **Description** Number of loop statements in the function

Mixed Lines

- **Mnemonic** MLOC
- **Description** Number of lines containing both code and comment in the source files.

Maximum Nested Structures

- **Mnemonic** NEST
- **Description** Maximum number of nested structures

Number of Parameters

- **Mnemonic** NOP
- **Description** Number of formal parameters in the function

Non-Cyclic Paths

- **Mnemonic** PATH
- **Description** Number of non-cyclic paths in the function.

Or else operators

- **Mnemonic** OREL
- **Description** Number of 'or else' operators

Number of #DEFINE

- **Mnemonic** P_DEFINE
- **Description** Number of #DEFINE

Number of #ELIF

- **Mnemonic** P_ELIF
- **Description** Number of #ELIF

Number of #ELSE

- **Mnemonic** P_ELSE
- **Description** Number of #ELSE

Number of #ENDIF

- **Mnemonic** P_ENDIF
- **Description** Number of #ENDIF

Number of #ERROR

- **Mnemonic** P_ERROR
- **Description** Number of #ERROR

Number of #IF

- **Mnemonic** P_IF
- **Description** Number of #IF

Number of #IFDEF

- **Mnemonic** P_IFDEF
- **Description** Number of #IFDEF

Number of #IFDEF

- **Mnemonic** P_IFNDEF
- **Description** Number of #IFNDEF

Number of Include

- **Mnemonic** P_INCLUDE
- **Description** Number of Include

Compiler FLAG Nested Level

- **Mnemonic** P_NEST
- **Description** Compiler FLAG Nested Level

Number of #PRAGMA

- **Mnemonic** P_PRAGMA
- **Description** Number of #PRAGMA

Number of #UNDEF

- **Mnemonic** P_UNDEF
- **Description** Number of #UNDEF

Number of #WARNING

- **Mnemonic** P_WARNING
- **Description** Number of #WARNING

Return Statements

- **Mnemonic** RETURN
- **Description** Number of 'return' statements in the function

Repeated Code Blocks

- **Mnemonic** RS
- **Description** Duplicated blocks in the function

Skipped Lines of Comment code

- **Mnemonic** SKLC
- **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

- **Mnemonic** SLOC
- **Description** Number of lines of source code in the source file(s).

Executable Statements

- **Mnemonic** STAT
- **Description** Total number of executable statements.

Switch Statements

- **Mnemonic** SWIT
- **Description** Number of 'switch' statements in the function

Operand Occurrences

- **Mnemonic** TOPD
- **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

- **Mnemonic** TOPT
- **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Lines Added

- **Mnemonic** LADD
- **Description** Number of lines added since the previous version.

Lines Modified

- **Mnemonic** LMOD
- **Description** Number of lines modified since the previous version.

Lines Removed

- **Mnemonic** LREM
- **Description** Number of lines removed since the previous version.

While Statements

- **Mnemonic** WHIL
- **Description** Number of 'while' statements in the function

2.14.2. PL/SQL Ruleset

Backward Goto shall not be used

- **Mnemonic** BWGOTO
- **Description** Backward gotos shall not be used.

Commented-out Source Code is not allowed

- **Mnemonic** R_CSTAT
- **Description** Commented-out Source Code is not allowed

Missing Default

- **Mnemonic** DEFAULT
- **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Missing final else

- **Mnemonic** ELSEFINAL
- **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Factorizable Classes

- **Mnemonic** CAC_CL
- **Description** Consider classes refactorization

Factorizable Files

- **Mnemonic** CAC_FI
- **Description** Consider files refactorization

Factorizable Functions

- **Mnemonic** CAC_FN
- **Description** Consider functions refactorization

Factorizable Packages

- **Mnemonic** CAC_PKG
- **Description** Consider packages refactorization

Cloned Classes

- **Mnemonic** CC_CL
- **Description** There shall be no duplicated classes

Cloned Files

- **Mnemonic** CC_FI
- **Description** There shall be no duplicated files

Cloned Functions

- **Mnemonic** CC_FN
- **Description** There shall be no duplicated functions

Cloned Algorithmic

- **Mnemonic** CFTC_FN
- **Description** There shall be no algorithmic cloning

Commit Used

- **Mnemonic** NOCOMMIT
- **Description** Commit instruction used in code

FIXME shall not be committed in sources code

- **Mnemonic** R_NOFIXME
- **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

- **Mnemonic** NOGOTO
- **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Rollback Used

- **Mnemonic** R_NOROLLBACK
- **Description** Rollback instruction used in code

Avoid Duplicated Blocks in Function

- **Mnemonic** RS_FN
- **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

- **Mnemonic** R_NOTODO
- **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

→ **Mnemonic** ONECASE

→ **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

2.15. TSQL

2.15.1. TSQL Metrics

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Header Blocks Of Comment

→ **Mnemonic** BHCO

→ **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

→ **Mnemonic** BLAN

→ **Description** Number of blank lines of code in the source file(s).

Brace Lines

→ **Mnemonic** BRAC

→ **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

→ **Mnemonic** BRKL

→ **Description** Number of 'break' statements in loop in the function

Catch Statements

→ **Mnemonic** CATC

→ **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

→ **Mnemonic** CCN

→ **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

- **Mnemonic** CFT
- **Description** Number of tokens in the control flow of functions

Call Graph Depth

- **Mnemonic** CGDM
- **Description** Maximum depth of the call graph.

Comment Lines

- **Mnemonic** CLOC
- **Description** Number of lines of comments in the source file(s).

Continue Statements

- **Mnemonic** CONT
- **Description** Number of 'continue' statements in the function

Commented Statements

- **Mnemonic** CSTAT
- **Description** Number of Commented Statements.

Minimum Number of Cycles

- **Mnemonic** CYCL
- **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Delete Statements

- **Mnemonic** DELETE
- **Description** Number of Delete statements

Distinct Operands

- **Mnemonic** DOPD
- **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

- **Mnemonic** DOPT
- **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Else Statements

- **Mnemonic** ELSE
- **Description** Number of 'else' statements

Structures Added

- **Mnemonic** SADD
- **Description** Number of control structures added since the previous version.

Structures Modified

- **Mnemonic** SMOD
- **Description** Number of control structures modified since the previous version.

Structures Removed

- **Mnemonic** SREM
- **Description** Number of control structures removed since the previous version.

Number of Structures

- **Mnemonic** SSIZ
- **Description** Number of control structures: iterations, selections, sequences

Goto Statements

- **Mnemonic** GOTO
- **Description** Number of 'goto' statements

Header Lines Of Comment

- **Mnemonic** HCOM
- **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

- **Mnemonic** HLOC
- **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

- **Mnemonic** ICC
- **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

- **Mnemonic** ICFTC
- **Description** Number of duplicated tokens in control flow of functions

If Statements

- **Mnemonic** IF
- **Description** Number of 'if' statements

Insert Statements

- **Mnemonic** INSERT
- **Description** Number of Insert statements

Label Statements

- **Mnemonic** LABEL
- **Description** Number of Label statements

Line Count

- **Mnemonic** LC
- **Description** Number of lines.

Mixed Lines

- **Mnemonic** MLOC
- **Description** Number of lines containing both code and comment in the source files.

Maximum Nested Structures

- **Mnemonic** NEST
- **Description** Maximum number of nested structures

Non-Cyclic Paths

- **Mnemonic** PATH
- **Description** Number of non-cyclic paths in the function.

Return Statements

- **Mnemonic** RETURN
- **Description** Number of 'return' statements in the function

Repeated Code Blocks

- **Mnemonic** RS
- **Description** Duplicated blocks in the function

Select Statements

- **Mnemonic** SELECT
- **Description** Number of Select statements

Skipped Lines of Comment code

- **Mnemonic** SKLC
- **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

- **Mnemonic** SLOC
- **Description** Number of lines of source code in the source file(s).

Executable Statements

- **Mnemonic** STAT
- **Description** Total number of executable statements.

Throw Statements

- **Mnemonic** THRO

→ **Description** Number of 'throw' statements in the function

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

→ **Mnemonic** TRY

→ **Description** Number of 'try' statements in the function

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

Update Statements

→ **Mnemonic** UPDATE

→ **Description** Number of Update statements

While Statements

→ **Mnemonic** WHIL

→ **Description** Number of 'while' statements in the function

2.15.2. TSQL Ruleset

Backward Goto shall not be used

→ **Mnemonic** BWGOTO

→ **Description** Backward gotos shall not be used.

Commented-out Source Code is not allowed

→ **Mnemonic** R_CSTAT

→ **Description** Commented-out Source Code is not allowed

Missing final else

→ **Mnemonic** ELSEFINAL

→ **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Factorizable Classes

→ **Mnemonic** CAC_CL

→ **Description** Consider classes refactorization

Factorizable Files

→ **Mnemonic** CAC_FI

→ **Description** Consider files refactorization

Factorizable Functions

→ **Mnemonic** CAC_FN

→ **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

Continue shall not be used

→ **Mnemonic** NOCONT

→ **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

→ **Mnemonic** NOGOTO

→ **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Avoid Duplicated Blocks in Function

→ **Mnemonic** RS_FN

→ **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

→ **Mnemonic** R_NOTODO

→ **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Multiple break in loop are not allowed

→ **Mnemonic** SGLBRK

→ **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

2.16. VB.net

2.16.1. VB.net Metrics

Constant Data

→ **Mnemonic** ACST

→ **Description** Number of constant data

Fiend Attributes

→ **Mnemonic** AFRI

→ **Description** Number of Friend Attributes

Number of Attributes

→ **Mnemonic** ANBR

→ **Description** Number of attributes

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Public Data

- **Mnemonic** APBL
- **Description** Number of public data

Protected Data

- **Mnemonic** APRT
- **Description** Number of protected data

Private data

- **Mnemonic** APRV
- **Description** Number of private data

Shadowed Attributes

- **Mnemonic** ASHD
- **Description** Number of Shadowed Attributes

Shared Attributes

- **Mnemonic** ASHR
- **Description** Number of Shared Attributes

Number of comment blocks

- **Mnemonic** BCOM
- **Description** Number of comment blocks.

Header Blocks Of Comment

- **Mnemonic** BHCO
- **Description** Number block of comment placed before the beginning of the artefact.

Blank Lines

- **Mnemonic** BLAN
- **Description** Number of blank lines of code in the source file(s).

Brace Lines

- **Mnemonic** BRAC
- **Description** Number of lines of code containing only a brace in the source file(s).

Break in Loop

- **Mnemonic** BRKL
- **Description** Number of 'break' statements in loop in the function

Stop Statements

- **Mnemonic** BRKP
- **Description** Number of Stop Statements (Breakpoints)

Break in Switch

- **Mnemonic** BRKS
- **Description** Number of 'break' statements in 'switch' in the function

Case Blocks

- **Mnemonic** CABL
- **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

- **Mnemonic** CASE
- **Description** Number of 'case' labels in the function

Catch Statements

- **Mnemonic** CATC
- **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

- **Mnemonic** CCN
- **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

- **Mnemonic** CFT
- **Description** Number of tokens in the control flow of functions

Call Graph Depth

- **Mnemonic** CGDM
- **Description** Maximum depth of the call graph.

Comment Lines

- **Mnemonic** CLOC
- **Description** Number of lines of comments in the source file(s).

Continue Statements

- **Mnemonic** CONT
- **Description** Number of 'continue' statements in the function

Commented Statements

- **Mnemonic** CSTAT
- **Description** Number of Commented Statements.

Minimum Number of Cycles

- **Mnemonic** CYCL
- **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Depth of Descendant Tree

→ **Mnemonic** DDT

→ **Description** Maximun depth of the inheritance tree from the class

Default Statement

→ **Mnemonic** DEFT

→ **Description** Number of 'default' blocks in 'switch' in the function

Depth of Inheritance Tree

→ **Mnemonic** DIT

→ **Description** Maximun depth of the class inheritance tree

Distinct Operands

→ **Mnemonic** DOPD

→ **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

→ **Mnemonic** DOPT

→ **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

→ **Mnemonic** DOWH

→ **Description** Number of 'do...while' statements in the function

Friend Events

→ **Mnemonic** EFRI

→ **Description** Number of Friend Events

Else Statements

→ **Mnemonic** ELSE

→ **Description** Number of 'else' statements

Events

→ **Mnemonic** ENBR

→ **Description** Number of Events

Public Events

→ **Mnemonic** EPBL

→ **Description** Number of Public Events

Protected Events

→ **Mnemonic** EPRT

→ **Description** Number of Protected Events

Private Events

- **Mnemonic** EPRV
- **Description** Number of Private Events

Shadowed Events

- **Mnemonic** ESHD
- **Description** Number of Shadowed Events

Shared Events

- **Mnemonic** ESHR
- **Description** Number of Shared Events

Call to exit

- **Mnemonic** EXIT
- **Description** Number of calls to the exit function

For Statements

- **Mnemonic** FOR
- **Description** Number of 'for' statements in the function

Structures Added

- **Mnemonic** SADD
- **Description** Number of control structures added since the previous version.

Structures Modified

- **Mnemonic** SMOD
- **Description** Number of control structures modified since the previous version.

Structures Removed

- **Mnemonic** SREM
- **Description** Number of control structures removed since the previous version.

Number of Structures

- **Mnemonic** SSIZ
- **Description** Number of control structures: iterations, selections, sequences

Goto Statements

- **Mnemonic** GOTO
- **Description** Number of 'goto' statements

Header Lines Of Comment

- **Mnemonic** HCOM
- **Description** Number of comment lines placed before the beginning of the artefact.

Header Lines Of Code

- **Mnemonic** HLOC
- **Description** Number of lines between the function or class definition and the first opening brace.

Cloned Code

- **Mnemonic** ICC
- **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

- **Mnemonic** ICFTC
- **Description** Number of duplicated tokens in control flow of functions

If Statements

- **Mnemonic** IF
- **Description** Number of 'if' statements

End Statements

- **Mnemonic** KILL
- **Description** Number of End Statements

Line Count

- **Mnemonic** LC
- **Description** Number of lines.

Loop Statements

- **Mnemonic** LOOP
- **Description** Number of loop statements in the function

Declare Members

- **Mnemonic** MDEC
- **Description** Number of Declare Members

Delegate Members

- **Mnemonic** MDEL
- **Description** Number of Delegate Members

Friend Members

- **Mnemonic** MFRI
- **Description** Number of Friend Members

Multiple Inheritance Indicator

- **Mnemonic** MII
- **Description** Number of classes from which the class inherits directly

Mixed Lines

→ **Mnemonic** MLOC

→ **Description** Number of lines containing both code and comment in the source files.

Must Members

→ **Mnemonic** MMST

→ **Description** Number of Must Members

Methods without Accessibility

→ **Mnemonic** MNON

→ **Description** Number of methods without any accessibility specifier

Partial Members

→ **Mnemonic** MPAR

→ **Description** Number of Partial Members

Public Methods

→ **Mnemonic** MPBL

→ **Description** Number of public methods

Protected Methods

→ **Mnemonic** MPRT

→ **Description** Number of protected methods

Private Methods

→ **Mnemonic** MPRV

→ **Description** Number of private methods

Shadowed Members

→ **Mnemonic** MSHD

→ **Description** Number of Shadowed Members

Shared Members

→ **Mnemonic** MSHR

→ **Description** Number of Shared Members

Number of Ancestors

→ **Mnemonic** NAC

→ **Description** Number of classes from which the class inherits directly or indirectly

Number of Descendants

→ **Mnemonic** NDC

→ **Description** Number of classes which inherit from the class directly or indirectly

Maximum Nested Structures

- **Mnemonic** NEST
- **Description** Maximum number of nested structures

Number Of Children

- **Mnemonic** NOC
- **Description** Number of classes which inherit directly from the class

Number of Methods

- **Mnemonic** NOM
- **Description** Number of methods defined in the class

Number of Parameters

- **Mnemonic** NOP
- **Description** Number of formal parameters in the function

Non-Cyclic Paths

- **Mnemonic** PATH
- **Description** Number of non-cyclic paths in the function.

Orelse operators

- **Mnemonic** OREL
- **Description** Number of 'orelse' operators

% of parsed tokens

- **Mnemonic** PARSE
- **Description** Percent of parsed tokens

Friend Properties

- **Mnemonic** PFRI
- **Description** Number of Friend Properties

Must Properties

- **Mnemonic** PMST
- **Description** Number of Must Properties

Properties

- **Mnemonic** PNBR
- **Description** Total number of properties

Public Properties

- **Mnemonic** PPBL
- **Description** Number of public properties

Protected Properties

- **Mnemonic** PPRT
- **Description** Number of protected properties

Private Properties

- **Mnemonic** PPRV
- **Description** Number of private properties

Shadowed Properties

- **Mnemonic** PSHD
- **Description** Number of Shadowed Properties

Shared Properties

- **Mnemonic** PSHR
- **Description** Number of Shared Properties

Return Statements

- **Mnemonic** RETURN
- **Description** Number of 'return' statements in the function

Repeated Code Blocks

- **Mnemonic** RS
- **Description** Duplicated blocks in the function

Skipped Lines of Comment code

- **Mnemonic** SKLC
- **Description** Skipped Lines of Comment code i.e. lines that match a user defined regular expression to skip lines of comments.

Source Lines Of Code

- **Mnemonic** SLOC
- **Description** Number of lines of source code in the source file(s).

Executable Statements

- **Mnemonic** STAT
- **Description** Total number of executable statements.

Switch Statements

- **Mnemonic** SWIT
- **Description** Number of 'switch' statements in the function

Ternary operators

- **Mnemonic** TERN

→ **Description** Number of ternary operators i.e. ?:

Throw Statements

→ **Mnemonic** THRO

→ **Description** Number of 'throw' statements in the function

Operand Occurrences

→ **Mnemonic** TOPD

→ **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

→ **Mnemonic** TOPT

→ **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

→ **Mnemonic** TRY

→ **Description** Number of 'try' statements in the function

Lines Added

→ **Mnemonic** LADD

→ **Description** Number of lines added since the previous version.

Lines Modified

→ **Mnemonic** LMOD

→ **Description** Number of lines modified since the previous version.

Lines Removed

→ **Mnemonic** LREM

→ **Description** Number of lines removed since the previous version.

While Statements

→ **Mnemonic** WHIL

→ **Description** Number of 'while' statements in the function

2.16.2. VB.net Ruleset

Backward Goto shall not be used

→ **Mnemonic** BWGOTO

→ **Description** Backward gotos shall not be used.

Missing Case Else clause

→ **Mnemonic** CASEELSE

→ **Description** The final clause of a Select statement shall be the Case Else clause.

Missing final else

→ **Mnemonic** ELSEFINAL

→ **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

Use of Exit Do statement

→ **Mnemonic** EXITDO

→ **Description** Do not use Exit Do statement to break a Do loop.

Use of Exit Function statement

→ **Mnemonic** EXITFCT

→ **Description** Do not use Exit Function statement, use Return instead.

Use of Exit For statement

→ **Mnemonic** EXITFOR

→ **Description** Do not use Exit For statement to break a For loop.

Use of Exit Property statement

→ **Mnemonic** EXITPROP

→ **Description** Do not use Exit Property statement, use Return instead.

Use of Exit Select statement

→ **Mnemonic** EXITSELECT

→ **Description** Do not use Exit Select statement to exit a Select statement.

Use of Exit Sub statement

→ **Mnemonic** EXITSUB

→ **Description** Do not use Exit Sub statement.

Use of Exit Try statement

→ **Mnemonic** EXITTRY

→ **Description** Do not use Exit Try statement to exit a Try statement.

Use of Exit While statement

→ **Mnemonic** EXITWHILE

→ **Description** Do not use Exit While statement to break a While loop.

Factorizable Classes

→ **Mnemonic** CAC_CL

→ **Description** Consider classes refactorization

Factorizable Files

→ **Mnemonic** CAC_FI

→ **Description** Consider files refactorization

Factorizable Functions

- **Mnemonic** CAC_FN
- **Description** Consider functions refactorization

Factorizable Packages

- **Mnemonic** CAC_PKG
- **Description** Consider packages refactorization

Cloned Classes

- **Mnemonic** CC_CL
- **Description** There shall be no duplicated classes

Cloned Files

- **Mnemonic** CC_FI
- **Description** There shall be no duplicated files

Cloned Functions

- **Mnemonic** CC_FN
- **Description** There shall be no duplicated functions

Cloned Algorithmic

- **Mnemonic** CFTC_FN
- **Description** There shall be no algorithmic cloning

Continue shall not be used

- **Mnemonic** NOCONT
- **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

FIXME shall not be committed in sources code

- **Mnemonic** R_NOFIXME
- **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

- **Mnemonic** NOGOTO
- **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Avoid Duplicated Blocks in Function

- **Mnemonic** RS_FN
- **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

- **Mnemonic** R_NOTODO
- **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

No case in Select

→ **Mnemonic** ONECASE

→ **Description** Every Select statement shall have at least one case clause.

Relaxed violation

→ **Mnemonic** RELAX

→ **Description** A rule violation is relaxed and justified.

Multiple exits are not allowed

→ **Mnemonic** RETURN

→ **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Multiple Exit Do statement

→ **Mnemonic** SGLEXITDO

→ **Description** For any iteration statement there shall be at most one Exit statement used for loop termination.

Multiple Exit (Function, Sub or Property) statement

→ **Mnemonic** SGLEXITFCT

→ **Description** A Function, Sub or Property must have only one Exit statement.

Multiple Exit For statement

→ **Mnemonic** SGLEXITFOR

→ **Description** For any iteration statement there shall be at most one Exit statement used for loop termination.

Multiple Exit While statement

→ **Mnemonic** SGLEXITWHILE

→ **Description** For any iteration statement there shall be at most one Exit statement used for loop termination.

2.17. Xaml

2.17.1. Xaml Metrics

Andthen Operators

→ **Mnemonic** ANTH

→ **Description** Number of 'andthen' operators

Number of attributes

→ **Mnemonic** ATTR

→ **Description** Number of attributes.

Number of comment blocks

→ **Mnemonic** BCOM

→ **Description** Number of comment blocks.

Blank Lines

- **Mnemonic** BLAN
- **Description** Number of blank lines of code in the source file(s).

Break in Loop

- **Mnemonic** BRKL
- **Description** Number of 'break' statements in loop in the function

Break in Switch

- **Mnemonic** BRKS
- **Description** Number of 'break' statements in 'switch' in the function

Case Blocks

- **Mnemonic** CABL
- **Description** Number of 'case' blocks in 'switch' in the function

Case Labels

- **Mnemonic** CASE
- **Description** Number of 'case' labels in the function

Catch Statements

- **Mnemonic** CATC
- **Description** Number of 'catch' statements in the function

Cyclomatic Complexity

- **Mnemonic** CCN
- **Description** Number of linearly independent paths in the function control graph.

Control Flow Token

- **Mnemonic** CFT
- **Description** Number of tokens in the control flow of functions

Call Graph Depth

- **Mnemonic** CGDM
- **Description** Maximum depth of the call graph.

Comment Lines

- **Mnemonic** CLOC
- **Description** Number of lines of comments in the source file(s).

Continue Statements

- **Mnemonic** CONT
- **Description** Number of 'continue' statements in the function

Commented Statements

- **Mnemonic** CSTAT
- **Description** Number of Commented Statements.

Minimum Number of Cycles

- **Mnemonic** CYCL
- **Description** Minimum number of call graph cycles in which the function is involved (including recursivity).

Default Statement

- **Mnemonic** DEFT
- **Description** Number of 'default' blocks in 'switch' in the function

Distinct Operands

- **Mnemonic** DOPD
- **Description** Number of distinct operands: variables and constants ([Halstead,76]: n2)

Distinct Operators

- **Mnemonic** DOPT
- **Description** Number of distinct operators: language keywords ([Halstead,76]: n1)

Do While Statements

- **Mnemonic** DOWH
- **Description** Number of 'do...while' statements in the function

Else Statements

- **Mnemonic** ELSE
- **Description** Number of 'else' statements

Number of XML elements

- **Mnemonic** ELT
- **Description** Number of XML elements.

For Statements

- **Mnemonic** FOR
- **Description** Number of 'for' statements in the function

Structures Added

- **Mnemonic** SADD
- **Description** Number of control structures added since the previous version.

Structures Modified

- **Mnemonic** SMOD
- **Description** Number of control structures modified since the previous version.

Structures Removed

- **Mnemonic** SREM
- **Description** Number of control structures removed since the previous version.

Number of Structures

- **Mnemonic** SSIZ
- **Description** Number of control structures: iterations, selections, sequences

Goto Statements

- **Mnemonic** GOTO
- **Description** Number of 'goto' statements

Cloned Code

- **Mnemonic** ICC
- **Description** Duplicated code in this artefact

Cloned Control Flow Tokens

- **Mnemonic** ICFTC
- **Description** Number of duplicated tokens in control flow of functions

If Statements

- **Mnemonic** IF
- **Description** Number of 'if' statements

Line Count

- **Mnemonic** LC
- **Description** Number of lines.

Loop Statements

- **Mnemonic** LOOP
- **Description** Number of loop statements in the function

Maximum Nested Structures

- **Mnemonic** NEST
- **Description** Maximum number of nested structures

Number of Parameters

- **Mnemonic** NOP
- **Description** Number of formal parameters in the function

Non-Cyclic Paths

- **Mnemonic** PATH
- **Description** Number of non-cyclic paths in the function.

Orelse operators

- **Mnemonic** OREL
- **Description** Number of 'orelse' operators

Return Statements

- **Mnemonic** RETURN
- **Description** Number of 'return' statements in the function

Repeated Code Blocks

- **Mnemonic** RS
- **Description** Duplicated blocks in the function

Source Lines Of Code

- **Mnemonic** SLOC
- **Description** Number of lines of source code in the source file(s).

Executable Statements

- **Mnemonic** STAT
- **Description** Total number of executable statements.

Switch Statements

- **Mnemonic** SWIT
- **Description** Number of 'switch' statements in the function

Ternary operators

- **Mnemonic** TERN
- **Description** Number of ternary operators i.e. ?:

Number of text blocks

- **Mnemonic** TEXT
- **Description** Number of text blocks.

Throw Statements

- **Mnemonic** THRO
- **Description** Number of 'throw' statements in the function

Operand Occurrences

- **Mnemonic** TOPD
- **Description** Number of occurrences of operands: variables and constants ([Halstead,76]: N2)

Operator Occurrences

- **Mnemonic** TOPT
- **Description** Number of occurrences of operators: language keywords ([Halstead,76]: N1)

Try Statements

- **Mnemonic** TRY
- **Description** Number of 'try' statements in the function

Lines Added

- **Mnemonic** LADD
- **Description** Number of lines added since the previous version.

Lines Modified

- **Mnemonic** LMOD
- **Description** Number of lines modified since the previous version.

Lines Removed

- **Mnemonic** LREM
- **Description** Number of lines removed since the previous version.

While Statements

- **Mnemonic** WHIL
- **Description** Number of 'while' statements in the function

2.17.2. Xaml Ruleset

Missing Break

- **Mnemonic** BRKFINAL
- **Description** An unconditional break statement shall terminate every non-empty switch clause (see [MISRA-C:2004]: RULE 15.2).

Backward Goto shall not be used

- **Mnemonic** BWGOTO
- **Description** Backward gotos shall not be used.

Comment Before Paragraph

- **Mnemonic** COMMENT
- **Description** A comment shall introduce a section or a paragraph.

Missing compound statement

- **Mnemonic** COMPOUND
- **Description** The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement (see [MISRA-C:2004]: RULE 14.8).

Missing compound if

- **Mnemonic** COMPOUNDIF
- **Description** An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement (see [MISRA-C:2004]: RULE 14.9).

Commented-out Source Code is not allowed

- **Mnemonic** R_CSTAT
- **Description** Commented-out Source Code is not allowed

Missing Default

- **Mnemonic** DEFAULT
- **Description** The final clause of a switch statement shall be the default clause (see [MISRA-C:2004]: RULE 15.3).

Missing final else

- **Mnemonic** ELSEFINAL
- **Description** All if ... else if constructs shall be terminated with an else clause (see [MISRA-C:2004]: RULE 14.10).

No Resources

- **Mnemonic** FORBIDDEN_ELEMENT
- **Description** Elements 'ResourceDictionary' are forbidden.

Resources Folder

- **Mnemonic** IN_FOLDER
- **Description** ResourceDictionary shall be in a 'Resources' directory

Assignment in Boolean

- **Mnemonic** NOASGCOND
- **Description** Assignment operators shall not be used in expressions that yield a boolean value

Assignment without Comparison

- **Mnemonic** NOASGINBOOL
- **Description** Assignment operators shall not be used in expressions that do not contain comparison operators.

Factorizable Classes

- **Mnemonic** CAC_CL
- **Description** Consider classes refactorization

Factorizable Files

- **Mnemonic** CAC_FI
- **Description** Consider files refactorization

Factorizable Functions

- **Mnemonic** CAC_FN
- **Description** Consider functions refactorization

Factorizable Packages

→ **Mnemonic** CAC_PKG

→ **Description** Consider packages refactorization

Cloned Classes

→ **Mnemonic** CC_CL

→ **Description** There shall be no duplicated classes

Cloned Files

→ **Mnemonic** CC_FI

→ **Description** There shall be no duplicated files

Cloned Functions

→ **Mnemonic** CC_FN

→ **Description** There shall be no duplicated functions

Cloned Algorithmic

→ **Mnemonic** CFTC_FN

→ **Description** There shall be no algorithmic cloning

There shall be a no code before first case

→ **Mnemonic** NOCODEBEFORECASE

→ **Description** There shall be a no code before the first case of a switch statement.

Continue shall not be used

→ **Mnemonic** NOCONT

→ **Description** The 'continue' statement shall not be used (see [MISRA-C:2004]: RULE 14.5).

Fallthrough shall be avoided

→ **Mnemonic** NOFALLTHROUGH

→ **Description** There shall be no fallthrough the next case in a switch statement.

FIXME shall not be committed in sources code

→ **Mnemonic** R_NOFIXME

→ **Description** FIXME shall not be committed in sources code as it brings confusion regarding code reliability.

GOTO shall not be used

→ **Mnemonic** NOGOTO

→ **Description** A unconditional GOTO shall not be used to jump outside the paragraph.

Label out a switch

→ **Mnemonic** NOLABEL

→ **Description** A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement (see [MISRA-C:2004]: RULE 15.1).

Avoid Duplicated Blocks in Function

- **Mnemonic** RS_FN
- **Description** There shall be no duplicated parts in functions

TODO shall not be committed in sources code

- **Mnemonic** R_NOTODO
- **Description** TODO shall not be committed in sources code as it brings confusion regarding code reliability.

Missing case in switch

- **Mnemonic** ONECASE
- **Description** Every switch statement shall have at least one case clause (see [MISRA-C:2004]: RULE 15.5).

Relaxed violation

- **Mnemonic** RELAX
- **Description** A rule violation is relaxed and justified.

Resources Filename

- **Mnemonic** RESOURCES_FILENAME
- **Description** All XAML resources files shall be suffixed with 'Resources.xaml'

Multiple exits are not allowed

- **Mnemonic** RETURN
- **Description** A function shall have a single point of exit at the end (see [MISRA-C:2004]: RULE 14.7).

Risky Empty Statement

- **Mnemonic** RISKYEMPTY
- **Description** Risky Empty Statement

Multiple break in loop are not allowed

- **Mnemonic** SGLBRK
- **Description** For any iteration statement there shall be at most one 'break' statement used for loop termination (see [MISRA-C:2004]: RULE 14.6).

3. Repository Connectors

3.1. Folder Path

3.1.1. Description

The simplest method to analyse source code in Squore is to provide a path to a folder containing your code.

Note

Remember that the path supplied for the analysis is a path local to the machine running the analysis, which may be different from your local machine. If you analyse source code on your local machine and then send results to the server, you will not be able to view the source code directly in Squore, since it will not have access to the source code on the other machine. A common workaround to this problem is to use UNC paths (`\\Server\Share`, `smb://server/share...`) or a mapped server drive in Windows.

3.1.2. Usage

Folder Path has the following options:

- **Datapath (path, mandatory)** Specify the absolute path to the folder containing the files you want to include in the analysis. The path specified must be accessible from the server.

The full command line syntax for Folder Path is:

```
-r "type=FROMPATH,path=[text]"
```

3.2. Zip Upload

3.2.1. Description

This Repository Connector allows you to upload a zip file containing your sources to analyse. Select a file to upload in the project wizard and it will be extracted and analysed on the server.

Note

The contents of the zip file are extracted into Squore Server's temp folder. If you want to uploaded files to persist, contact your Squore administrator so that the uploaded zip files and extracted sources are moved to a location that is not deleted at each server restart.

3.2.2. Usage

This Repository Connector is only available from the web UI, not from the command line interface.

3.3. CVS

3.3.1. Description

The Concurrent Versions System (CVS), is a client-server free software revision control system in the field of software development.

For more details, refer to <http://savannah.nongnu.org/projects/cvs>.

Note

The following is a list of commands used by the CVS Repository Connector to retrieve sources:

```
→ cvs -d $repository export [-r $branch] $project
→ cvs -d $repository co -r $artefactPath -d $tmpFolder
```

3.3.2. Usage

CVS has the following options:

- **Repository (repository , mandatory)** Specify the location of the CVS Repository.
- **Project (project , mandatory)** Specify the name of the project to get files from.
- **Tag or Branch (branch)** Specify the tag or branch to get the files from.

The full command line syntax for CVS is:

```
-r "type=CVS,repository=[text],project=[text],branch=[text]"
```

3.4. ClearCase

3.4.1. Description

IBM Rational ClearCase is a software configuration management solution that provides version control, workspace management, parallel development support, and build auditing. The command executed on the server to check out source code is: \$cleartool \$view_root_path \$view \$vob_root_path.

For more details, refer to <http://www-03.ibm.com/software/products/en/clearcase>.

Note

The ClearCase tool is configured for Linux by default. It is possible to make it work for Windows by editing the configuration file

3.4.2. Usage

ClearCase has the following options:

- **View root path (view_root_path , mandatory, default: /view)** Specify the absolute path of the ClearCase view.
- **Vob Root Path (vob_root_path , mandatory, default: /projets)** Specify the absolute path of the ClearCase vob.
- **View (view)** Specify the label of the view to analyse sources from. If no view is specified, the current ClearCase view will be used automatically, as retrieved by the command cleartool pwv -s.
- **Server Display View (server_display_view)** When viewing source code from the Explorer after building the project, this parameter is used instead of the view parameter specified earlier. Leave this field empty to use the same value as for view.
- **Sources Path (sub_path)** Specify a path in the view to restrict the scope of the source code to analyse. The value of this field must not contain the vob nor the view. Leave this field empty to analyse the code in the entire view. This parameter is only necessary if you want to restrict to a directory lower than root.

The full command line syntax for ClearCase is:

```
-r "type=ClearCase,view_root_path=[text],vob_root_path=[text],view=[text],server_display_view=[text]"
```


3.5. Perforce

3.5.1. Description

The Perforce server manages a central database and a master repository of file versions. Perforce supports both Git clients and clients that use Perforce's own protocol.

For more details, refer to <http://www.perforce.com/>.

Note

The Perforce repository connector assumes that the specified depot exists on the specified Perforce server, that Squore can access this depot and that the Perforce user defined has the right to access it. The host where the analysis takes place must have a Perforce command-line client (p4) installed and fully functional. The P4PORT environment variable is not read by Squore. You have to set it in the form. The path to the p4 command can be configured in the `perforce_conf.tcl` file located in the `configuration/repositoryConnectors/Perforce` folder. The following is a list of commands used by the Perforce Repository Connector to retrieve sources:

```
→ p4 -p $p4port [-u username] [-P password] client -i <$tmpFolder/  
p4conf.txt  
→ p4 -p $p4port [-u username] [-P password] -c $clientName sync  
"$depot/...@$label"  
→ p4 -p $p4port [-u username] [-P password] client -d $clientName  
→ p4 -p $p4port [-u username] [-P password] print -q -o $outputFile  
$artefactPath
```

The format of the `p4conf.txt` file is:

```
Client: $clientName  
Root: $tmpFolder  
Options: noallwrite noclobber nocompress unlocked nomodtime normdir  
SubmitOptions: submitunchanged  
view:  
$depot/... //$clientName/...
```

3.5.2. Usage

Perforce has the following options:

- **P4PORT (p4port , mandatory)** Specify the value of P4PORT using the format [protocol:]host:port (the protocol is optional). This parameter is necessary even if you have specified an environment variable on the machine where the analysis is running.
- **Depot (depot , mandatory)** Specify the name of the depot (and optionally subfolders) containing the sources to be analysed.
- **Revision (label)** Specify a label, changelist or date to retrieve the corresponding revision of the sources. Leave this field empty to analyse the most recent revision for the sources.
- **Authentication (useAccountCredentials , default: NO_CREDENTIALS)**
- **Username (username)**
- **Password (password)**

The full command line syntax for Perforce is:

```
-r  
"type=Perforce,p4port=[text],depot=[text],label=[text],useAccountCredentials=[multipleChoice]
```

3.6. Git

3.6.1. Description

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

For more details, refer to <http://git-scm.com/>.

Note

The following is a list of commands used by the Git Repository Connector to retrieve sources:

```
→ git clone [$username:$password@$url] $tmpFolder
→ git checkout $commit
→ git log -1 "--format=%H"
→ git config --get remote.origin.url
→ git clone [$username:$password@$url] $tmpFolder
→ git checkout $commit
→ git fetch
→ git --git-dir=$gitRoot show $artefactPath
```

Note

Git 1.7.1 is known to fail with a fatal: HTTP request failed error on CentOS 6.9. For this OS, it is recommended to upgrade to git 2.9 as provided by software collections on <https://www.softwarecollections.org/en/scls/rhsc/rh-git29/> and point to the new binary in `git_config.tcl` or make the change permanent as described on <https://access.redhat.com/solutions/527703>.

3.6.2. Usage

Git has the following options:

- **URL (url , mandatory)** URL of the git repository to get files from. The local, HTTP(s), SSH and Git protocols are supported.
- **Branch or commit (commit)** This field allows specifying the SHA1 of a commit or a branch name. If a SHA1 is specified, it will be retrieved from the default branch. If a branch label is specified, then its latest commit is analysed. Leave this field empty to analyse the latest commit of the default branch.
- **Sub-directory (subDir)** Specify a subfolder name if you want to restrict the analysis to a subpath of the repository root.
- **Authentication (useAccountCredentials , default: NO_CREDENTIALS)**
- **Username (username)**
- **Password (password)**

The full command line syntax for Git is:

```
-r
" type=Git , url=[text] , commit=[text] , subDir=[text] , useAccountCredentials=[multipleChoice] , usernam
```

3.7. PTC Integrity

3.7.1. Description

This Repository Connector allows analysing sources hosted in PTC Integrity, a software system lifecycle management and application lifecycle management platform developed by PTC.

For more details, refer to <http://www.ptc.com/products/integrity/>.

Note

You can modify some of the settings of this repository connector if the `si.exe` and `mksAPIViewer.exe` binaries are not in your path. For versions that do not support the `--xmlapi` option, you can also turn off this method of retrieving file information. These settings are available by editing `mks_conf.tcl` in the repository connector's configuration folder.

3.7.2. Usage

PTC Integrity has the following options:

- **Server Hostname (`hostname` , mandatory)** Specify the name of the Integrity server. This value is passed to the command line using the parameter `--hostname`.
- **Port (`port`)** Specify the port used to connect to the Integrity server. This value is passed to the command line using the parameter `--port`.
- **Project (`project`)** Specify the name of the project containing the sources to be analysed. This value is passed to the command line using the `--project` parameter.
- **Revision (`revision`)** Specify the revision number for the sources to be analysed. This value is passed to the command line using the `--projectRevision` parameter.
- **Scope (`scope` , default: `name:*.c,name:*.h`)** Specifies the scope (filter) for the Integrity sandbox extraction. This value is passed to the command line using the `--scope` parameter.
- **Authentication (`useAccountCredentials` , default: `NO_CREDENTIALS`)**
- **Username (`username`)**
- **Password (`password`)**

The full command line syntax for PTC Integrity is:

```
-r  
"type=MKS,hostname=[text],port=[text],project=[text],revision=[text],scope=[text],useAccountC
```

3.8. TFS

3.8.1. Description

Team Foundation Server (TFS) is a Microsoft product which provides source code management, reporting, requirements management, project management, automated builds, lab management, testing and release management capabilities. This Repository Connector provides access to the sources hosted in TFS's revision control system.

For more details, refer to <https://www.visualstudio.com/products/tfs-overview-vs>.

Note

The TFS repository connector (Team Foundation Server - Team Foundation Version Control) assumes that a TFS command-line client (Visual Studio Client or Team Explorer Everywhere) is installed on the Squore server and fully functional. The configuration of this client must be set up in the `tfs_conf.tcl`

file. The repository connector form must be filled according to the TFS standard (eg. the Project Path must begin with the '\$' character...). Note that this repository connector works with a temporary workspace that is deleted at the end of the analysis. The following is a list of commands used by the TFS Repository Connector to retrieve sources:

```
→ tf.exe workspace [/login:$username,$password] /server:$url /noprompt /
new $workspace
→ tf.exe workfold [/login:$username,$password] /map $path $tempFolder /
workspace:$workspace
→ tf.exe get [/login:$username,$password] /version:$version /recursive /
force $path
→ tf.exe workspace [/login:$username,$password] /delete $workspace
→ tf.exe view [/login:$username,$password] /server:$artefactPath
```

Note

When using the Java Team Explorer Everywhere client, / is replaced by - and the view command is replaced by print.

3.8.2. Usage

TFS has the following options:

- **URL (URL , mandatory)** Specify the URL of the TFS server.
- **Path (path , mandatory)** Path the project to be analysed. This path usually starts with \$.
- **Version (version)** Specify the version of the sources to analyse. This field accepts a changeset number, date, or label. Leave the field empty to analyse the most recent revision of the sources.
- **Authentication (useAccountCredentials , default: NO_CREDENTIALS)**
- **Username: (username)**
- **Password (password)**

The full command line syntax for TFS is:

```
-r
"type=TFS,URL=[text],path=[text],version=[text],useAccountCredentials=[multipleChoice],username=[text],password=[text]
```

3.9. Synergy

3.9.1. Description

Rational Synergy is a software tool that provides software configuration management (SCM) capabilities for all artifacts related to software development including source code, documents and images as well as the final built software executable and libraries.

For more details, refer to <http://www-03.ibm.com/software/products/en/ratisyne>.

Note

The Synergy repository connector assumes that a project already exists and that the Synergy user defined has the right to access it. The host where the analysis takes place must have Synergy installed and fully functional. Note that, as stated in IBM's documentation on http://pic.dhe.ibm.com/infocenter/synhelp/v7m2r0/index.jsp?topic=%2Fcom.ibm.rational.synergy.manage.doc%2Ftopics%2Fsc_t_h_start_cli_session.html, using credentials is only supported on Windows, so use the NO_CREDENTIALS option when Synergy runs

on a Linux host. The following is a list of commands used by the Synergy Repository Connector to retrieve sources:

```
→ ccm start -d $db -nogui -m -q [-s $server] [-pw $password] [-n $user -pw
password]
→ ccm prop "$path@$projectSpec"
→ ccm copy_to_file_system -path $tempFolder -recurse $projectSpec
→ ccm cat "$artefactPath@$projectSpec"
→ ccm stop
```

3.9.2. Usage

Synergy has the following options:

- **Server URL (server)** Specify the Synergy server URL, if using a distant server. If specified, the value is used by the Synergy client via the -s parameter.
- **Database (db , mandatory)** Specify the database path to analyse the sources it contains.
- **Project Specification (projectSpec , mandatory)** Specify the project specification for the analysis. Source code contained in this project specification will be analysed recursively.
- **Subfolder (subFolder)** Specify a subfolder name if you want to restrict the scope of the analysis to a particular folder.
- **Authentication: (useAccountCredentials , default: NO_CREDENTIALS)** Note that, as stated in IBM's documentation, using credentials is only supported on Windows. The "No Credentials" must be used option when Synergy runs on a Linux host. For more information, consult http://pic.dhe.ibm.com/infocenter/synhelp/v7m2r0/index.jsp?topic=%2Fcom.ibm.rational.synergy.manage.doc%2Ftopics%2Fsc_t_h_start_cli_session.html.
- **(name)**
- **Password (password)**

The full command line syntax for Synergy is:

```
-r
"type=Synergy,server=[text],db=[text],projectSpec=[text],subFolder=[text],useAccountCredentials=[text]
```

3.10. SVN

3.10.1. Description

Connecting to an SVN server is supported using svn over ssh, or by using a username and password.

For more details, refer to <https://subversion.apache.org/>.

Note

The following is a list of commands used by the SVN Repository Connector to retrieve sources (you can edit the common command base or the path to the executable in <SQUARE_HOME>/configuration/repositoryConnectors/SVN/svn_conf.tcl if needed):

```
→ svn info --xml --non-interactive --trust-server-cert --no-auth-cache [--
username $username] [--password $password] [-r $revision] $url
→ svn export --force --non-interactive --trust-server-cert --no-auth-cache
[--username $username] [--password $password] [-r $revision] $url
```

3.10.2. Usage

SVN has the following options:

- **URL (`url` , mandatory)** Specify the URL of the SVN repository to export and analyse. The following protocols are supported: `svn://`, `svn+ssh://`, `http://`, `https://`.
- **Revision (`rev`)** Specify a revision number in this field, or leave it blank to analyse files at the HEAD revision.
- **External references (`externals` , default: `exclude`)** Specify if when extracting sources from SVN the system should also extract external references.
- **Authentication (`useAccountCredentials` , default: `NO_CREDENTIALS`)**
- **Username (`username`)**
- **Password (`password`)**

The full command line syntax for SVN is:

```
-r "type=SVN,url=[text],rev=[text],externals=[multipleChoice],useAccountCredentials=[multipleCho
```

3.11. Using Multiple Nodes

Squore allows using multiple repositories in the same analysis. If your project consists of some code that is spread over two distinct servers or SVN repositories, you can set up your project so that it includes both locations in the project analysis. This is done by labelling each source code node before specifying parameters, as shown below

```
-r "type=FROMPATH,alias=Node1,path=/home/projects/client-code"  
-r "type=FROMPATH,alias=Node2,path=/home/projects/common/lib"
```

Note that only alpha-numeric characters are allowed to be used as labels. In the artefact tree, each node will appear as a separate top-level folder with the label provided at project creation.

Using multiple nodes, you can also analyse sources using different Repository Connectors in the same analysis:

```
-r "type=FROMPATH,alias=Node1,path=/home/projects/common-config"  
-r "type=SVN,alias=Node2,url=svn+ssh://10.10.0.1/var/svn/project/src,rev=HEAD"
```

4. Data Providers

This chapter describes the available Data Providers and the default parameters that they accept via the Command Line Interface.

4.1. AntiC

4.1.1. Description

AntiC is a part of the jlint static analysis suite and is launched to analyse C and C++ source code and produce findings.

For more details, refer to <http://jlint.sourceforge.net/>.

Note

On Linux, the antiC executable must be compiled manually before you run it for the first time by running the command:

```
# cd <SQUORE_HOME>/addons/tools/Antic_auto/bin/ && gcc antic.c -o antic
```

4.1.2. Usage

AntiC has the following options:

→ **Source code directory to analyse (dir)** Leave this parameter empty if you want to analyse all sources specified above.

The full command line syntax for AntiC is:

```
-d "type=Antic_auto,dir=[text]"
```

4.2. Automotive Coverage Import

4.2.1. Description

Automotive Coverage Import: generic import mechanism for coverage results at FUNCTION level

4.2.2. Usage

Automotive Coverage Import has the following options:

→ **Enter the CSV file for coverage measures (csv)** CSV File shall contain the following (PATH;NAME;TESTED_C1;OBJECT_C1;TESTED_MCC;OBJECT_MCC;TESTED_MCDC;OBJECT_MCDC)

The full command line syntax for Automotive Coverage Import is:

```
-d "type=Automotive_Coverage, csv=[text]"
```

4.3. Automotive Tag Import

4.3.1. Description

4.3.2. Usage

Automotive Tag Import has the following options:

- **Enter the CSV file for measures (csv)**

The full command line syntax for Automotive Tag Import is:

```
-d "type=Automotive_Tag_Import, csv=[text]"
```

4.4. BullseyeCoverage Code Coverage Analyzer

4.4.1. Description

BullseyeCoverage is a code coverage analyzer for C++ and C. The coverage report file is used to generate metrics.

For more details, refer to <http://www.bullseye.com/>.

4.4.2. Usage

BullseyeCoverage Code Coverage Analyzer has the following options:

- **HTML report (html)** Specify the path to the HTML report file generated by BullseyeCoverage.

The full command line syntax for BullseyeCoverage Code Coverage Analyzer is:

```
-d "type=BullseyeCoverage, html=[text]"
```

4.5. CPD

4.5.1. Description

CPD is an open source tool which generates Copy/Paste metrics. The detection of duplicated blocks is set to 100 tokens. CPD provides an XML file which can be imported to generate metrics as well as findings.

For more details, refer to <http://pmd.sourceforge.net/pmd-5.3.0/usage/cpd-usage.html>.

4.5.2. Usage

CPD has the following options:

- **CPD XML results (xml)** Specify the path to the XML results file generated by CPD. The minimum supported version is PMD/CPD 4.2.5.

The full command line syntax for CPD is:

```
-d "type=CPD, xml=[text]"
```

4.6. Cppcheck

4.6.1. Description

Cppcheck is a static analysis tool for C/C++ applications. The tool provides an XML output which can be imported to generate findings.

For more details, refer to <http://cppcheck.sourceforge.net/>.

4.6.2. Usage

Cppcheck has the following options:

- **Cppcheck XML results (`xml`)** Specify the path to the XML results file from Cppcheck. Note that the minimum required version of Cppcheck for this data provider is 1.61.

The full command line syntax for Cppcheck is:

```
-d "type=CPPCheck,xml=[text]"
```

4.7. Cppcheck (plugin)

4.7.1. Description

Cppcheck is a static analysis tool for C/C++ applications. The tool provides an XML output which can be imported to generate findings.

For more details, refer to <http://cppcheck.sourceforge.net/>.

Note

On Windows, this data provider requires an extra download to extract the Cppcheck binary in `<SQUORE_HOME>/addons/tools/PPCheck_auto/` and the MS Visual C++ 2010 Redistributable Package available from Microsoft. On Linux, you can install the cppcheck application anywhere you want. The path to the Cppcheck binary for Linux can be configured in `config.tcl`. For more information, refer to the Installation and Administration Guide's Third-Party Plugins and Applications [`./install_admin_manual/index.html#sect_thirdparty_plugins`] section.

4.7.2. Usage

Cppcheck (plugin) has the following options:

- **Source code folder (`dir`)** Specify the folder containing the source files to analyse. If you want to analyse all of source repositories specified for the project, leave this field empty.

The full command line syntax for Cppcheck (plugin) is:

```
-d "type=PPCheck_auto,dir=[text]"
```

4.8. CPPTest

4.8.1. Description

Parasoft C/C++test is an integrated solution for automating a broad range of best practices proven to improve software development team productivity and software quality for C and C++. The tool provides an XML output file which can be imported to generate findings and metrics.

For more details, refer to <http://www.parasoft.com/product/cpptest/>.

4.8.2. Usage

CPPTest has the following options:

- **XML results file (`xml`)** Specify the path to the CPPTest results file. This data provider is compatible with files exported from CPPTest version 7.2.10.34 and up.

The full command line syntax for CPPTest is:

```
-d "type=CPPTest,xml=[text]"
```

4.9. Cantata

4.9.1. Description

Cantata is Test Coverage tools. It provides an XML output which can be imported to generate coverage metrics at function level.

For more details, refer to <http://www.qa-systems.com/cantata.html>.

4.9.2. Usage

Cantata has the following options:

- **Cantata XML results (`xml`)** Specify the path to the XML results file from Cantata 6.2

The full command line syntax for Cantata is:

```
-d "type=Cantata,xml=[text]"
```

4.10. CheckStyle

4.10.1. Description

CheckStyle is an open source tool that verifies that Java applications adhere to certain coding standards. It produces an XML file which can be imported to generate findings.

For more details, refer to <http://checkstyle.sourceforge.net/>.

4.10.2. Usage

CheckStyle has the following options:

- **CheckStyle results file (`xml`)** Point to the XML file that contains Checkstyle results. Note that the minimum supported version is Checkstyle 5.3.

The full command line syntax for CheckStyle is:

```
-d "type=CheckStyle,xml=[text]"
```

4.11. CheckStyle (plugin)

4.11.1. Description

CheckStyle is an open source tool that verifies that Java applications adhere to certain coding standards. It produces an XML file which can be imported to generate findings.

For more details, refer to <http://checkstyle.sourceforge.net/>.

Note

This data provider requires an extra download to extract the CheckStyle binary in `<SQUORE_HOME>/addons/tools/CheckStyle_auto/`. For more information, refer to the Installation and Administration Guide's Third-Party Plugins and Applications [`../install_admin_manual/index.html#sect_thirdparty_plugins`] section.

4.11.2. Usage

CheckStyle (plugin) has the following options:

- **Configuration file (`configFile`)** A Checkstyle configuration specifies which modules to plug in and apply to Java source files. Modules are structured in a tree whose root is the Checker module. Specify the name of the configuration file only, and the data provider will try to find it in the `CheckStyle_auto` folder of your custom configuration. If no custom configuration file is found, a default configuration will be used.
- **Xmx (`xmx` , default: `1024m`)** Maximum amount of memory allocated to the java process launching Checkstyle.
- **Excluded directory pattern (`excludedDirectoryPattern`)** Java regular expression of directories to exclude from CheckStyle, for example: `^test|generated-sources|.*-report$` or `ou ^lib$`

The full command line syntax for CheckStyle (plugin) is:

```
-d  
"type=CheckStyle_auto,configFile=[text],xmx=[text],excludedDirectoryPattern=[text]"
```

4.12. CheckStyle for SQALE (plugin)

4.12.1. Description

CheckStyle is an open source tool that verifies that Java applications adhere to certain coding standards. It produces an XML file which can be imported to generate findings.

For more details, refer to <http://checkstyle.sourceforge.net/>.

Note

This data provider requires an extra download to extract the CheckStyle binary in `<SQUORE_HOME>/addons/tools/CheckStyle_auto_for_SQALE/`. For more information, refer to the Installation and Administration Guide's Third-Party Plugins and Applications [`../install_admin_manual/index.html#sect_thirdparty_plugins`] section.

4.12.2. Usage

CheckStyle for SQALE (plugin) has the following options:

- **Configuration file (`configFile` , default: `config_checkstyle_for_sqale.xml`)** A Checkstyle configuration specifies which modules to plug in and apply to Java source files. Modules are structured in a tree whose root is the Checker module. Specify the name of the configuration file only, and the data provider will try to

find it in the CheckStyle_auto folder of your custom configuration. If no custom configuration file is found, a default configuration will be used.

- **Xmx (`xmx` , default: **1024m**)** Maximum amount of memory allocated to the java process launching Checkstyle.

The full command line syntax for CheckStyle for SQALE (plugin) is:

```
-d "type=CheckStyle_auto_for_SQALE,configFile=[text],xmx=[text]"
```

4.13. Cobertura

4.13.1. Description

Cobertura is a free code coverage library for Java. Its XML report file can be imported to generate code coverage metrics for your Java project.

For more details, refer to <http://cobertura.github.io/cobertura/>.

4.13.2. Usage

Cobertura has the following options:

- **XML report (`xml`)** Specify the path to the XML report generated by Cobertura.

The full command line syntax for Cobertura is:

```
-d "type=Cobertura,xml=[text]"
```

4.14. CodeSonar

4.14.1. Description

Codesonar is a static analysis tool for C and C++ code designed for zero tolerance defect environments. It provides an XML output file which is imported to generate findings.

For more details, refer to <http://www.grammatech.com/codesonar>.

4.14.2. Usage

CodeSonar has the following options:

- **XML results file (`xml`)** Specify the path to the XML results file generated by Codesonar. The minimum version of Codesonar compatible with this data provider is 3.3.

The full command line syntax for CodeSonar is:

```
-d "type=CodeSonar,xml=[text]"
```

4.15. Compiler

4.15.1. Description

Compiler Warning impor allows to import information from compiler

For more details, refer to Compiler.

4.15.2. Usage

Compiler has the following options:

→ **Compiler output csv file (Path;Line;Rule;Descr - with: Rule = COMP_ERR|COMPILER_WARN|COMPILER_INFO) (txt , mandatory)**

The full command line syntax for Compiler is:

```
-d "type=Compiler,txt=[text]"
```

4.16. Coverity

4.16.1. Description

Coverity is a static analysis tool for C, C++, Java and C#. It provides an XML output which can be imported to generate findings.

For more details, refer to <http://www.coverity.com/>.

4.16.2. Usage

Coverity has the following options:

→ **XML results file (xml)** Specify the path to the XML file containing Coverity results.

The full command line syntax for Coverity is:

```
-d "type=Coverity,xml=[text]"
```

4.17. FindBugs

4.17.1. Description

Findbugs is an open source tool that looks for bugs in Java code. It produces an XML result file which can be imported to generate findings.

For more details, refer to <http://findbugs.sourceforge.net/>.

4.17.2. Usage

FindBugs has the following options:

→ **XML results file (xml)** Specify the location of the XML file containing Findbugs results. Note that the minimum supported version of FindBugs is 1.3.9.

The full command line syntax for FindBugs is:

```
-d "type=Findbugs,xml=[text]"
```

4.18. FindBugs (plugin)

4.18.1. Description

Findbugs is an open source tool that looks for bugs in Java code. It produces an XML result file which can be imported to generate findings. Note that the data provider requires an extra download to extract the Findbugs binary in [INSTALLDIR]/addons/tools/Findbugs_auto/. You are free to use FindBugs 3.0 or FindBugs 2.0 depending on what your standard is. For more information, refer to the Installation and Administration Manual's "Third-Party Plugins and Applications" section.

For more details, refer to <http://findbugs.sourceforge.net/>.

Note

This data provider requires an extra download to extract the Findbugs binary in <SQUORE_HOME>/addons/tools/Findbugs_auto/. For more information, refer to the Installation and Administration Guide's Third-Party Plugins and Applications [../install_admin_manual/index.html#sect_thirdparty_plugins] section.

4.18.2. Usage

FindBugs (plugin) has the following options:

- **Classes (class_dir , mandatory)** Specify the folders and/or jar files for your project in classpath format, or point to a text file that contains one folder or jar file per line.
- **Auxiliary Class path (auxiliarypath)** Specify a list of folders and/or jars in classpath format, or specify the path to a text file that contains one folder or jar per line. This information will be passed to FindBugs via the -auxclasspath parameter.
- **Memory Allocation (xmx , default: 1024m)** Maximum amount of memory allocated to the java process launching FindBugs.

The full command line syntax for FindBugs (plugin) is:

```
-d "type=Findbugs_auto,class_dir=[text],auxiliarypath=[text],xmx=[text]"
```

4.19. Function Relaxer

4.19.1. Description

4.19.2. Usage

Function Relaxer has the following options:

- **Enter the CSV file for measures (csv)**

The full command line syntax for Function Relaxer is:

```
-d "type=Function_Relaxer, csv=[text]"
```

4.20. FxCop

4.20.1. Description

FxCop is an application that analyzes managed code assemblies (code that targets the .NET Framework common language runtime) and reports information about the assemblies, such as possible design,

localization, performance, and security improvements. FxCop generates an XML results file which can be imported to generate findings.

For more details, refer to [https://msdn.microsoft.com/en-us/library/bb429476\(v=vs.80\).aspx](https://msdn.microsoft.com/en-us/library/bb429476(v=vs.80).aspx).

4.20.2. Usage

FxCop has the following options:

- **XML results file (`xml`)** Specify the XML file containing FxCop's analysis results. Note that the minimum supported version of FxCop is 1.35.

The full command line syntax for FxCop is:

```
-d "type=FxCop,xml=[text]"
```

4.21. GCov

4.21.1. Description

GCov is a Code coverage program for C application. GCov generates raw text files which can be imported to generate metrics.

For more details, refer to <http://gcc.gnu.org/onlinedocs/gcc/Gcov.html>.

4.21.2. Usage

GCov has the following options:

- **Directory containing results files (`dir`)** Specify the path of the root directory containing the GCov results files.
- **Results files extension (`ext` , default: `*.c.gcov`)** Specify the file extension of GCov results files.

The full command line syntax for GCov is:

```
-d "type=GCov,dir=[text],ext=[text]"
```

4.22. GNATcheck

4.22.1. Description

GNATcheck is an extensible rule-based tool that allows developers to completely define a coding standard. The results are output to a log file that can be imported to generate findings.

For more details, refer to <http://www.adacore.com/gnatpro/toolsuite/gnatcheck/>.

4.22.2. Usage

GNATcheck has the following options:

- **Log file (`txt`)** Specify the path to the log file generated by the GNATcheck run.

The full command line syntax for GNATcheck is:

```
-d "type=GnatCheck , txt=[ text ] "
```

4.23. GNATCompiler

4.23.1. Description

GNATCompiler is a free-software compiler for the Ada programming language which forms part of the GNU Compiler Collection. It supports all versions of the language, i.e. Ada 2012, Ada 2005, Ada 95 and Ada 83. It creates a log file that can be imported to generate findings.

For more details, refer to <http://www.adacore.com/gnatpro/toolsuite/compilation/>.

4.23.2. Usage

GNATCompiler has the following options:

→ **Log file (log)** Specify the path to the log file containing the compiler warnings.

The full command line syntax for GNATCompiler is:

```
-d "type=GnatCompiler , log=[ text ] "
```

4.24. JUnit

4.24.1. Description

JUnit is a simple framework to write repeatable tests. It is an instance of the xUnit architecture for unit testing frameworks. JUnit XML result files are imported to generate findings and the total number of tests is made available as a measure.

For more details, refer to <http://junit.org/>.

4.24.2. Usage

JUnit has the following options:

→ **Results folder (resultDir , mandatory)** Specify the path to the folder containing the JUnit results. The data provider will parse all available XML files. Note that the minimum support version of JUnit is 4.10.

The full command line syntax for JUnit is:

```
-d "type=JUnit , resultDir=[ text ] "
```

4.25. JaCoCo

4.25.1. Description

JaCoCo is a free code coverage library for Java. Its XML report file can be imported to generate code coverage metrics for your Java project.

For more details, refer to <http://www.eclemma.org/jacoco/>.

4.25.2. Usage

JaCoCo has the following options:

- **XML report (`xml` , mandatory)** Specify the path to the XML report generated by JaCoCo. Note that the folder containing the XML file must also contain JaCoCo's report DTD file, available from <http://www.eclemma.org/jacoco/trunk/coverage/report.dtd>. XML report files are supported from version 0.6.5.

The full command line syntax for JaCoCo is:

```
-d "type=Jacoco,xml=[text]"
```

4.26. Klocwork

4.26.1. Description

Klocwork is a static analysis tool. Its XML result file can be imported to generate findings.

For more details, refer to <http://www.klocwork.com>.

4.26.2. Usage

Klocwork has the following options:

- **XML results file (`xml`)** Specify the path to the XML results file exported from Klocwork. Note that Klocwork version 9.6.1 is the minimum required version.

The full command line syntax for Klocwork is:

```
-d "type=Klocwork,xml=[text]"
```

4.27. Rational Logiscope

4.27.1. Description

The Logiscope suite allows the evaluation of source code quality in order to reduce maintenance cost, error correction or test effort. It can be applied to verify C, C++, Java and Ada languages and produces a CSV results file that can be imported to generate findings.

For more details, refer to <http://www.kalimetrix.com/en/logiscope>.

4.27.2. Usage

Rational Logiscope has the following options:

- **RuleChecker results file (`csv`)** Specify the path to the CSV results file from Logiscope.

The full command line syntax for Rational Logiscope is:

```
-d "type=Logiscope,csv=[text]"
```

4.28. MemUsage

4.28.1. Description

4.28.2. Usage

MemUsage has the following options:

→ **Memory Usage excel file (excel)**

The full command line syntax for MemUsage is:

```
-d "type=MemUsage,excel=[text]"
```

4.29. NCover

4.29.1. Description

NCover is a Code coverage program for C# application. NCover generates an XML results file which can be imported to generate metrics.

For more details, refer to <http://www.ncover.com/>.

4.29.2. Usage

NCover has the following options:

→ **XML results file (xml)** Specify the location of the XML results file generated by NCover. Note that the minimum supported version is NCover 3.0.

The full command line syntax for NCover is:

```
-d "type=NCover,xml=[text]"
```

4.30. Oracle PLSQL compiler Warning checker

4.30.1. Description

This data provider reads an Oracle compiler log file and imports the warnings as findings. Findings extracted from the log file are filtered using a prefix parameter.

For more details, refer to <http://www.oracle.com/>.

4.30.2. Usage

Oracle PLSQL compiler Warning checker has the following options:

→ **Compiler log file (log)**

→ **Prefixes (prefix)** Prefixes and their replacements are specified as pairs using the syntax [prefix1|node1;prefix2|node2]. Leave this field empty to disable filtering. The parsing algorithm looks for lines fitting this pattern: [PATH;SCHEMA;ARTE_ID;ARTE_TYPE;LINE;COL;SEVERITY_TYPE;WARNING_ID;SEVERITY_ID;DESCR] and keeps lines where [PATH] begins with one of the input prefixes. In each kept [PATH], [prefix] is replaced by [node]. If [node] is empty, [prefix] is removed from [PATH], but not replaced. Some valid syntaxes

for prefix: One prefix to remove: svn://aaaa:12345/valid/path/from/svn One prefix to replace: svn://aaaa:12345/valid/path/from/svn|node1 Two prefixes to remove: svn://aaaa:12345/valid/path/from/svn|svn://bbbb:12345/valid/path/from/other_svn| Two prefixes to replace: svn://aaaa:12345/valid/path/from/svn;svn://bbbb:12345/valid/path/from/other_svn Two prefixes to replace: svn://aaaa:12345/valid/path/from/svn|node1;svn://bbbb:12345/valid/path/from/other_svn|node2

The full command line syntax for Oracle PLSQL compiler Warning checker is:

```
-d "type=Oracle_PLSQLCompiler,log=[text],prefix=[text]"
```

4.31. MISRA Rule Checking using PC-lint

4.31.1. Description

PC-lint is a static code analyser. The PC-lint data provider reads an PC-lint log file and imports MISRA violations as findings.

For more details, refer to <http://www.gimpel.com/html/pcl.htm>.

4.31.2. Usage

MISRA Rule Checking using PC-lint has the following options:

- **Log file folder (logDir)** Specify the path to the folder containing the PC-lint log files.
- **Extensions to exclude (excludedExtensions , default: .h;.H)** Specify the file extensions to exclude from the reported violations.

The full command line syntax for MISRA Rule Checking using PC-lint is:

```
-d "type=PC_Lint_MISRA,logDir=[text],excludedExtensions=[text]"
```

4.32. PMD

4.32.1. Description

PMD scans Java source code and looks for potential problems like possible bugs, dead code, sub-optimal code, overcomplicated expressions, duplicate code... The XML results file it generates is read to create findings.

For more details, refer to <http://pmd.sourceforge.net>.

4.32.2. Usage

PMD has the following options:

- **XML results file (xml)** Specify the path to the PMD XML results file. Note that the minimum supported version of PMD for this data provider is 4.2.5.

The full command line syntax for PMD is:

```
-d "type=PMD,xml=[text]"
```

4.33. PMD (plugin)

4.33.1. Description

PMD scans Java source code and looks for potential problems like possible bugs, dead code, sub-optimal code, overcomplicated expressions, duplicate code ... The XML results file it generates is read to create findings.

For more details, refer to <http://pmd.sourceforge.net>.

Note

This data provider requires an extra download to extract the PMD binary in `<SQUORE_HOME>/addons/tools/PMD_auto/`. For more information, refer to the Installation and Administration Guide's Third-Party Plugins and Applications [[../install_admin_manual/index.html#sect_thirdparty_plugins](#)] section.

4.33.2. Usage

PMD (plugin) has the following options:

- **Ruleset file (configFile)** Specify the path to the PMD XML ruleset you want to use for this analysis. If you do not specify a ruleset, the default one from `INSTALLDIR/addons/tools/PMD_auto` will be used.

The full command line syntax for PMD (plugin) is:

```
-d "type=PMD_auto,configFile=[text]"
```

4.34. Polyspace

4.34.1. Description

Polyspace is a static analysis tool which includes a MISRA checker. It produces an XML output which can be imported to generate findings. Polyspace Verifier detects RTE (RunTime Error) such as Division by zero, Illegal Dereferencing Pointer, Out of bound array index... Such information is turned into statistical measures at function level. Number of Red (justified/non-justified), Number of Grey (justified/non-justified), Number of Orange (justified/non-justified), Number of Green.

For more details, refer to <http://www.mathworks.com/products/polyspace/index.html>.

4.34.2. Usage

Polyspace has the following options:

- **XML results file (xml)** Specify the path to the XML results file generated by Polyspace.

The full command line syntax for Polyspace is:

```
-d "type=Polyspace,xml=[text]"
```

4.35. Polyspace MISRA

4.35.1. Description

Polyspace is a static analysis tool which includes a MISRA checker. It produces an XML output which can be imported to generate findings. Polyspace Verifier detects RTE (RunTime Error) such as Division by zero, Illegal Dereferencing Pointer, Out of bound array index... Such information is turned into statistical measures at

function level. Number of Red (justified/non-justified), Number of Grey (justified/non-justified), Number of Orange (justified/non-justified), Number of Green.

For more details, refer to <http://www.mathworks.com/products/polyspace/index.html>.

4.35.2. Usage

Polyspace MISRA has the following options:

- **Results folder (resultDir)** Specify the folder containing the Polyspace results. The data provider will parse all sub-folders searching for XML result files called "MISRA-CPP-report.xml" or "MISRA-C-report.xml" located in a "Polyspace-Doc" folder and aggregate results.
- **Unit by Unit (unitByUnit , default: true)** Check this box if the Polyspace verification was run unit by unit.

The full command line syntax for Polyspace MISRA is:

```
-d "type=Polyspace_MISRA,resultDir=[text],unitByUnit=[booleanChoice]"
```

4.36. Polyspace (plugin)

4.36.1. Description

Polyspace is a static analysis tool which includes a MISRA checker. It produces an binary output format which can be imported to generate findings. Polyspace Verifier detects RTE (RunTime Error) such as Division by zero, Illegal Dereferencing Pointer, Out of bound array index... Such information is turned into statistical measures at function level. Number of Red (justified/non-justified), Number of Grey (justified/non-justified), Number of Orange (justified/non-justified), Number of Green. Note that this data provider requires an extra download to extract the Polyspace Export binary in [INSTALLDIR]/addons/tools/Polyspace_RTE/. For more information, refer to the Installation and Administration Manual's "Third-Party Plugins and Applications" section.

For more details, refer to <http://www.mathworks.com/products/polyspace/index.html>.

Note

This data provider requires an extra download to extract the Polyspace Export binary in <SQUORE_HOME>/addons/tools/Polyspace_RTE. For more information, refer to the Installation and Administration Guide's Third-Party Plugins and Applications [../install_admin_manual/index.html#sect_thirdparty_plugins] section.

4.36.2. Usage

Polyspace (plugin) has the following options:

- **Results folder (resultDir)** Specify the folder containing the Polyspace results. The data provider will run the polyspace-export binary on all sub-folders to export results to XML and aggregate them.
- **Unit by Unit (unitByUnit , default: true)** Check this box if the Polyspace verification was run unit by unit.

The full command line syntax for Polyspace (plugin) is:

```
-d "type=Polyspace_RTE,resultDir=[text],unitByUnit=[booleanChoice]"
```

4.37. MISRA Rule Checking with QAC

4.37.1. Description

QAC identifies problems in C source code caused by language usage that is dangerous, overly complex, non-portable, difficult to maintain, or simply diverges from coding standards. Its CSV results file can be imported to generate findings.

For more details, refer to <http://www.phaedsys.com/principals/programmingresearch/pr-qac.html>.

4.37.2. Usage

MISRA Rule Checking with QAC has the following options:

- **Code Folder (logDir)** Specify the path to the folder that contains the annotated files to process. For the findings to be successfully linked to their corresponding artefact, several requirements have to be met: - The annotated file name should be [Original source file name].txt e.g. The annotation of file "controller.c" should be called "controller.c.txt" - The annotated file location in the annotated directory should match the associated source file location in the source directory. e.g. The annotation for source file "[SOURCE_DIR]/subDir1/subDir2/controller.c" should be located in "[ANNOTATIONS_DIR]/subDir1/subDir2/controller.c.txt" The previous comment suggests that the source and annotated directory are different. However, these directories can of course be identical, which ensures that locations of source and annotated files are the same.
- **Extension (ext , default: html)** Specify the extension used by QAC to create annotated files.

The full command line syntax for MISRA Rule Checking with QAC is:

```
-d "type=QAC_MISRA,logDir=[text],ext=[text]"
```

4.38. Unit Test Code Coverage from Rational Test RealTime

4.38.1. Description

Rational Test RealTime is a cross-platform solution for component testing and runtime analysis of embedded software. Metrics are generated from its CSV results file.

For more details, refer to <http://www-01.ibm.com/software/awdtools/test/realtime/>.

4.38.2. Usage

Unit Test Code Coverage from Rational Test RealTime has the following options:

- **.xrd folder (logDir)** Specify the path to the folder containing the .xrd files generated by RTRT.
- **Excluded file extensions (excludedExtensions , default: .h;.H)**

The full command line syntax for Unit Test Code Coverage from Rational Test RealTime is:

```
-d "type=RTRT,logDir=[text],excludedExtensions=[text]"
```

4.39. ReqIF

4.39.1. Description

RIF/ReqIF (Requirements Interchange Format) is an XML file format that can be used to exchange requirements, along with its associated metadata, between software tools from different vendors.

For more details, refer to <http://www.omg.org/spec/ReqIF/>.

4.39.2. Usage

ReqIF has the following options:

→ (**dir**)

→ **Spec Object Type (objType , default: _AUTO_)** Specify the SPEC_OBJECT_TYPE property LONG-NAME to be used to process the ReqIf file. Using the _AUTO_ value will let the Data Provider extract the value from the ReqIf file, and assumes that there is only one such definition.

The full command line syntax for ReqIF is:

```
-d "type=ReqIf,dir=[text],objType=[text]"
```

4.40. SQL Code Guard

4.40.1. Description

SQL Code Guard is a free solution for SQL Server that provides fast and comprehensive static analysis for T-Sql code, shows code complexity and objects dependencies.

For more details, refer to <http://www.sqlcodeguard.com>.

4.40.2. Usage

SQL Code Guard has the following options:

→ **XML results (xml)** Specify the path to the XML files containing SQL Code Guard results.

The full command line syntax for SQL Code Guard is:

```
-d "type=SQLCodeGuard,xml=[text]"
```

4.41. Squan Sources

4.41.1. Description

Squan Sources provides basic-level analysis of your source code.

For more details, refer to <http://www.squoring.com>.

Note

The analyser can output info and warning messages in the build logs. Recent additions to those logs include better handling of structures in C code, which will produce these messages:

- [Analyzer] Unknown syntax declaration for function XXXXX at line yyy to indicate that we would have found a function but, probably due to preprocessing directives, we are not able to parse it.
- [Analyzer] Unbalanced () blocks found in the file. Probably due to preprocessing directives, parenthesis in the file are not well balanced.
- [Analyzer] Unbalanced { } blocks found in the file. Probably due to preprocessing directives, curly brackets in the file are not well balanced.

Tip

You can specify the languages for your source code by passing pairs of language and extensions to the **languages** parameter. Extensions are case-sensitive and cannot be used for two different languages. For example, a project mixing php and javascript files can be analysed with:

```
--dp "type=SQuORE, languages=php:.php;javascript:.js,.JS"
```

In order to launch an analysis using all the available languages by default, do not specify the **languages** parameter in your command line.

4.41.2. Usage

Squan Sources has the following options:

- **Languages** (**languages** , **default:** **abap;ada;c;cpp;mindc;csharp;cobol;java;javascript;fortran77;fortran90;php;s...**) Check the boxes for the languages used in the specified source repositories. Adjust the list of file extensions as necessary. Note that two languages cannot use the same file extension, and that the list of extensions is case-sensitive. Tip: Leave all the boxes unchecked and Squan Sources will auto-detect the language parser to use.
- **Force full analysis (rebuild_all , default: false)** Analyses are incremental by default. Check this box if you want to force the source code parser to analyse all files instead of only the ones that have changed since the previous analysis. This is useful if you added new rule files or text parsing rules and you want to re-evaluate all files based on your modifications.
- **Generate control graphs (genCG , default: true)** This option allows generating a control graph for every function in your code. The control graph is visible in the dashboard of the function when the analysis completes.
- **Use qualified names (qualified , default: false)** Note: This option cannot be modified in subsequent runs after you create the first version of your project.
- **Limit analysis depth (depth , default: false)** Use this option to limit the depth of the analysis to file-level only. This means that Squan Sources will not create any class or function artefacts for your project.
- **Add a 'Source Code' node (scnode , default: false)** Using this options groups all source nodes under a common source code node instead of directly under the APPLICATION node. This is useful if other data providers group non-code artefacts like tests or requirements together under their own top-level node. This option can only be set when you create a new project and cannot be modified when creating a new version of your project.
- **'Source Code' node label (scnode_name , default: Source Code)** Specify a custom label for your main source code node. Note: this option is not modifiable. It only applies to projects where you use the "Add a 'Source Code' node" option. When left blank, it defaults to "Source Code".
- **Compact folders (compact_folder , default: true)** When using this option, folders with only one son are aggregates together. This avoids creating many unnecessary levels in the artefact tree to get to the first level of files in your project. This option cannot be changed after you have created the first version of your project.
- **Content exclusion via regexp (pattern)** Specify a PERL regular expression to automatically exclude files from the analysis if their contents match the regular expression. Leave this field empty to disable content-based file exclusion.
- **File Filtering (files_choice , default: Exclude)** Specify a pattern and an action to take for matching file names. Leave the pattern empty to disable file filtering.
- **pattern (pattern_files)** Use a shell-like wildcard e.g. '*-test.c'. * Matches any sequence of characters in string, including a null string. ? Matches any single character in string. [chars] Matches any character in the set given by chars. If a sequence of the form x-y appears in chars, then any character between x and y, inclusive, will match. On Windows, this is used with the -nocase option, meaning that the end points of

the range are converted to lower case first. Whereas `[[A-z]]` matches `'_'` when matching case-sensitively (`'_'` falls between the `'Z'` and `'a'`), with `-nocase` this is considered like `[[A-Za-z]]`. `\x` Matches the single character `x`. This provides a way of avoiding the special interpretation of the characters `*?[]` in pattern. Tip: Use `;` to separate multiple patterns.

- **Folder Filtering (`dir_choice` , default: **Exclude**)** Specify a pattern and an action to take for matching folder names. Leave the pattern empty to disable folder filtering.
- **pattern (`pattern_dir`)** Use a shell-like wildcard e.g. `'Test_*'`. `*` Matches any sequence of characters in string, including a null string. `?` Matches any single character in string. `[chars]` Matches any character in the set given by `chars`. If a sequence of the form `x-y` appears in `chars`, then any character between `x` and `y`, inclusive, will match. On Windows, this is used with the `-nocase` option, meaning that the end points of the range are converted to lower case first. Whereas `[[A-z]]` matches `'_'` when matching case-sensitively (`'_'` falls between the `'Z'` and `'a'`), with `-nocase` this is considered like `[[A-Za-z]]`. `\x` Matches the single character `x`. This provides a way of avoiding the special interpretation of the characters `*?[]` in pattern. Tip: Use `;` to separate multiple patterns.
- **Detect algorithmic cloning (`clAlg` , default: **true**)** When checking this box, Squan Sources launches a cloning detection tool capable of finding algorithmic cloning in your code.
- **Detect text cloning (`clTxt` , default: **true**)** When checking this box, Squan Sources launches a cloning detection tool capable of finding text duplication in your code.
- **Backwards-compatible cloning (`clBw` , default: **false**)** When checking this box, the cloning detection tool is run in a way that produces metrics that are backwards-compatible with earlier versions of this product (2014-A): exact matching is used for algorithmic cloning and a 5% margin is used for text duplication. This legacy behaviour should only be used if you are using an old configuration that was developed before 2014-B.
- **Cloning fault ratio (`clFR` , default: **0.1**)** This threshold defines how much cloning between two artefacts is necessary for them to be considered as clones by the cloning detection tool. For example, a fault ratio of 0.1 means that two artefacts are considered clones if less than 10% of their contents differ. Note that this option is ignored if you are using backwards-compatible cloning.
- **Detect Open Source cloning (deprecated) (`clOS` , default: **false**)** This option is no longer supported and should not be used anymore.
- **Compute Textual stability (`genTs` , default: **true**)** This option allows keeping track of the stability of the code analysed for each version. The computed stability is available on the dashboard as a metric called and can be interpreted as 0% meaning completely changed and 100% meaning not changed at all.
- **Compute Algorithmic stability (`genAs` , default: **true**)** This option allows keeping track of the stability of the code analysed for each version. The computed stability is available on the dashboard as a metric called Stability Index (SI) and can be interpreted as 0% meaning completely changed and 100% meaning not changed at all.
- **Detect artefact renaming (`clRen` , default: **true**)** This option allows Squan Sources to detect artefacts that have been moved since the previous version, ensuring that the stability metrics of the previous artefact are passed to the new one. This is typically useful if you have moved a file to a different folder in your source tree and do not want to lose the previous metrics generated for this file. If you do not use this option, moved artefacts will be considered as new artefacts.
- **Additional parameters (`additional_param`)** These additional parameters can be used to pass instructions to external processes started by this data provider. This value is generally left empty in most cases.

The full command line syntax for Squan Sources is:

```
-d
" type=SquORE , languages=[multipleChoice] , rebuild_all=[booleanChoice] , genCG=[booleanChoice] , qua
```

4.42. Squore Import

4.42.1. Description

Squore Import is a data provider used to import the results of another data provider analysis. It is generally only used for debugging purposes.

For more details, refer to <http://www.squoring.com>.

4.42.2. Usage

Squore Import has the following options:

- **XML folder (`inputDir`)** Specify the folder that contains the `squore_data_*.xml` files that you want to import.

The full command line syntax for Squore Import is:

```
-d "type=SQuOREImport,inputDir=[text]"
```

4.43. Squore Virtual Project

4.43.1. Description

Squore Virtual Project is a data provider that can use the output of several projects to compile metrics in a meta-project composed of the import sub-projects.

For more details, refer to <http://www.squoring.com>.

4.43.2. Usage

Squore Virtual Project has the following options:

- **Paths to output.xml files (`output`)** Specify the paths to all the `output.xml` files you want to include in the virtual project. Separate paths using ';'.

The full command line syntax for Squore Virtual Project is:

```
-d "type=SQuOREVirtualProject,output=[text]"
```

4.44. StyleCop

4.44.1. Description

StyleCop is a C# code analysis tool. Its XML output is imported to generate findings.

For more details, refer to <https://stylecop.codeplex.com/>.

4.44.2. Usage

StyleCop has the following options:

- **XML results file (`xml`)** Specify the path to the StyleCop XML results file. The minimum version compatible with this data provider is 4.7.

The full command line syntax for StyleCop is:

```
-d "type=StyleCop,xml=[text]"
```

4.45. StyleCop (plugin)

4.45.1. Description

StyleCop is a C# code analysis tool. Its XML output is imported to generate findings.

For more details, refer to <https://stylecop.codeplex.com/>.

Note

Note that this data provider is not supported on Linux. On windows, this data provider requires an extra download to extract the StyleCop binary in <SQUORE_HOME>/addons/tools/StyleCop_auto/ and .NET framework 3.5 to be installed on your machine (run `Net.SF.StyleCopCmd.Console.exe` manually once to install .NET automatically). For more information, refer to the Installation and Administration Guide's Third-Party Plugins and Applications [[../install_admin_manual/index.html#sect_thirdparty_plugins](#)] section.

4.45.2. Usage

StyleCop (plugin) has the following options:

- **Solution (`sln`)** Specify the path to the .sln file to analyse. Leave empty to analyse all .sln found in the source repository.

The full command line syntax for StyleCop (plugin) is:

```
-d "type=StyleCop_auto,sln=[text]"
```

4.46. Tessy

4.46.1. Description

Tessy is a tool automating module/unit testing of embedded software written in dialects of C/C++. Tessy generates an XML results file which can be imported to generate metrics. This data provider supports importing files that have a `xml_version="1.0"` attribute in their header.

For more details, refer to <https://www.hitex.com/en/tools/tessy/>.

4.46.2. Usage

Tessy has the following options:

- **Results folder (`resultDir`)** Specify the top folder containing XML result files from Tessy. Note that this data provider will recursively scan sub-folders looking for `index.xml` files to aggregate results.

The full command line syntax for Tessy is:

```
-d "type=Tessy,resultDir=[text]"
```

4.47. VectorCAST

4.47.1. Description

VectorCAST

For more details, refer to VectorCAST.

4.47.2. Usage

VectorCAST has the following options:

→ **HTML Report (`html_report`)** Enter the path to the HTML report which contains the Coverage results

The full command line syntax for VectorCAST is:

```
-d "type=VectorCAST,html_report=[text]"
```

4.48. CodeSniffer

4.48.1. Description

CodeSniffer is a rulecker for PHP and Javascript

For more details, refer to <http://www.squizlabs.com/php-codesniffer>.

4.48.2. Usage

CodeSniffer has the following options:

→ **CodeSniffer results file (checkstyle formmated xml) (`xml`)** Point to the XML file that contains CodeSniffer results.

The full command line syntax for CodeSniffer is:

```
-d "type=codesniffer,xml=[text]"
```

4.49. Configuration Checker

4.49.1. Description

Use this tool to check for duplicated files or XML Elements between a custom configuration and the standard configuration.

4.49.2. Usage

Configuration Checker has the following options:

→ **Standard Configuration Path (`s`)**

→ **Custom Configurations Path (`p`)**

The full command line syntax for Configuration Checker is:

```
-d "type=conf-checker,s=[text],p=[text]"
```

4.50. Csv Coverage Import

4.50.1. Description

Csv Coverage Import: generic import mechanism for coverage results at FUNCTION level

4.50.2. Usage

Csv Coverage Import has the following options:

- **Enter the CSV file for coverage measures (csv)** CSV File shall contain the following (PATH;NAME;TESTED_C1;OBJECT_C1;TESTED_MCC;OBJECT_MCC;TESTED_MCDC;OBJECT_MCDC;TCOV_MCC;TCOV_MCDC;T

The full command line syntax for Csv Coverage Import is:

```
-d "type=csv_coverage , csv=[text]"
```

4.51. CSV Findings

4.51.1. Description

CSV Findings (Generic Import of findings)

4.51.2. Usage

CSV Findings has the following options:

- **CSV File (FILE;FUNCTION;RULE_ID;MESSAGE;LINE;COL;STATUS;STATUS_MESSAGE;TOOL) (csv)** Your CSV file should use include the following as a header: FILE;FUNCTION;RULE_ID;MESSAGE;LINE;COL;STATUS;STATUS_MESSAGE;TOOL. CSV files in other formats are not supported.

The full command line syntax for CSV Findings is:

```
-d "type=csv_findings , csv=[text]"
```

4.52. Csv Tag Import

4.52.1. Description

Csv Tag Import

4.52.2. Usage

Csv Tag Import has the following options:

- **Enter the CSV file for measures (csv)**

The full command line syntax for Csv Tag Import is:

```
-d "type=csv_tag_import , csv=[text]"
```

4.53. Csv Test Results Import

4.53.1. Description

Csv Test Results Import: generic import mechanism for Test results at FILES level

4.53.2. Usage

Csv Test Results Import has the following options:

- **Enter the CSV file for Test Results measures at FILES level (csv)** CSV File shall contain the following (PATH;NB_TEST;NB_ERROR;NB_FAILURE;NB_PASS)

The full command line syntax for Csv Test Results Import is:

```
-d "type=csv_test , csv=[text]"
```

4.54. OSLC

4.54.1. Description

OSLC-CM allows retrieving information from Change Management systems following the OSLC standard. Metrics and artefacts are created by connecting to the OSLC system and retrieving issues with the specified query.

For more details, refer to <http://open-services.net/>.

4.54.2. Usage

OSLC has the following options:

- **Change Server (server)** Specify the URL of the project you want to query on the OSLC server. Typically the URL will look like this: <http://myserver:8600/change/oslc/db/3454a67f-656ddd4348e5/role/User/>
- **Query (query)** Specify the query to send to the OSLC server (e.g.: release="9TDE/TDE_00_01_00_00"). It is passed to the request URL via the ?oslc_cm.query= parameter.
- **Query Properties (properties , default: request_type,problem_number,crstatus,severity,submission_area,functionality...)** Specify the properties to add to the query. They are passed to the OSLC query URL using the ?oslc_cm.properties= parameter.
- **Login (login)**
- **Password (password)**

The full command line syntax for OSLC is:

```
-d "type=oslc_cm , server=[text] , query=[text] , properties=[text] , login=[text] , password=[password]"
```

4.55. pep8

4.55.1. Description

pep8 is a tool to check your Python code against some of the style conventions in PEP 88. Its CSV report file is imported to generate findings.

For more details, refer to <https://pypi.python.org/pypi/pep8>.

4.55.2. Usage

pep8 has the following options:

- **CSV results file (csv)** Specify the path to the CSV report file created by pep8.

The full command line syntax for pep8 is:

```
-d "type=pep8, csv=[text]"
```

4.56. pycodestyle / pep8 (plugin)

4.56.1. Description

Style Guide for Python Code. Pep8 results are imported to produce findings on Python code. This data provider requires having pycodestyle or pep8 installed on the machine running the analysis and the pycodestyle or pep8 command to be available in the path. It is compatible with pycodestyle 2.4 or pep8 1.7 and may also work with older versions.

For more details, refer to <https://pypi.org/project/pycodestyle>.

4.56.2. Usage

pycodestyle / pep8 (plugin) has the following options:

- **Source code directory to analyse (dir)** Leave this field empty to analyse all sources.

The full command line syntax for pycodestyle / pep8 (plugin) is:

```
-d "type=pep8_auto, dir=[text]"
```

4.57. PHP Code Coverage

4.57.1. Description

PHP Code Coverage

For more details, refer to <https://github.com/sebastianbergmann/php-code-coverage>.

4.57.2. Usage

PHP Code Coverage has the following options:

- **HTML Report Folder (html_report)** Enter the path to the HTML report folder which contains the Coverage results

The full command line syntax for PHP Code Coverage is:

```
-d "type=phpcodecoverage,html_report=[text]"
```

4.58. pylint

4.58.1. Description

Pylint is a Python source code analyzer which looks for programming errors, helps enforcing a coding standard and sniffs for some code smells (as defined in Martin Fowler's Refactoring book). Pylint results are imported to generate findings for Python code.

For more details, refer to <http://www.pylint.org/>.

4.58.2. Usage

pylint has the following options:

- **CSV results file (csv)** Specify the path to the CSV file containing pylint results. Note that the minimum version supported is 1.1.0.

The full command line syntax for pylint is:

```
-d "type=pylint,csv=[text]"
```

4.59. pylint (plugin)

4.59.1. Description

Coding Guide for Python Code. Pylint results are imported to produce findings on Python code. This data provider requires having pylint installed on the machine running the analysis and the pylint command to be available in the path. It is known to work with pylint 1.7.0 and may also work with older versions.

4.59.2. Usage

pylint (plugin) has the following options:

- **Source code directory to analyse (dir)** Leave this field empty to analyse all sources.

The full command line syntax for pylint (plugin) is:

```
-d "type=pylint_auto,dir=[text]"
```

4.60. Qac_8_2

4.60.1. Description

QA-C is a static analysis tool for MISRA checking.

For more details, refer to <http://www.programmingresearch.com/static-analysis-software/qac-qacpp-static-analyzers/>.

4.60.2. Usage

Qac_8_2 has the following options:

→ **QAC output file (.tab file) (txt , mandatory)**

The full command line syntax for Qac_8_2 is:

```
-d "type=qac , txt=[text]"
```

4.61. Advanced COBOL Parsing

By default, Squan Sources generates artefacts for all PROGRAMs in COBOL source files. It is possible to configure the parser to also generate artefacts for all SECTIONs and PARAGRAPHs in your source code. This feature can be enabled with the following steps:

1. Open <SQUORE_HOME>/configuration/tools/SQuORE/Analyzer/artifacts/cobol/ArtifactsList.txt
2. Edit the list of artefacts to generate and add the section and paragraph types:

```
program  
section  
paragraph
```

3. Save your changes

If you create a new project, you will see the new artefacts straight away. For already-existing projects, make sure to launch a new analysis and check Squan Sources's **Force full analysis** option to parse the entire code again and generate the new artefacts.

4.62. Using Data Provider Input Files From Version Control

Input files for Squore's Data Providers, like source code, can be located in your version control system. When this is the case, you need to specify a variable in the input field for the Data Provider instead of an absolute path to the input file.

Specify Repository Locations


Folder
 Zip Upload
 ClearCase
 Git
 PTC Integrity
 Perforce
 SVN
 Synergy
 TFS
 ?

Datapath *
?

Select Data Providers

<input type="checkbox"/> AntiC	<input type="checkbox"/> GCov	<input checked="" type="checkbox"/> Squan Sources
<input type="checkbox"/> Automotive Coverage Import	<input type="checkbox"/> GNATcheck	<input type="checkbox"/> Squore Import
<input type="checkbox"/> Automotive Tag Import	<input type="checkbox"/> GNATCompiler	<input type="checkbox"/> Squore Virtual Project
<input type="checkbox"/> BullseyeCoverage Code Coverage Analyzer	<input type="checkbox"/> JUnit	<input type="checkbox"/> StyleCop
<input type="checkbox"/> CPD	<input type="checkbox"/> JaCoCo	<input type="checkbox"/> StyleCop (plugin)
<input checked="" type="checkbox"/> Cppcheck	<input type="checkbox"/> Klocwork	<input type="checkbox"/> Tessy

Cppcheck



Cppcheck
XML results

?

A Data Provider using an input file extracted from a remote repository

The variable to use varies depending on your scenario:

→ **You have only one node of source code in your project**

In this case, the variable to use is **\$src**.

→ **You have more than one node of source code in your project**

In this case, you need to tell Squore in which node the input file is located. This is done using a variable that has the same name as the alias you defined for the source code node in the previous step of the wizard. For example, if your nodes are labelled **Node1** and **Node2** (the default names), then you can refer to them using the **\$Node1** and **\$Node2** variables.

Tip

When using these variables from the command line on a linux system, the **\$** symbol must be escaped:

```
-d "type=PMD,configFile=\\$src/pmd_data.xml"
```

4.63. Providing a catalog file to a Data Provider for Offline XSL Transformations

When transforming an XML results file with an XSL stylesheet, the XML parser used by Squore will try to validate the XML file against the DTD declared in the XML header. In cases where the XSL transformation is running on a machine with no internet access, this can result in the execution of the Data Provider failing with a `No route to host` error message.

You can fix this issue by modifying the data provider to use a catalog file that will provide an alternate location for the DTD used to validate the XML. This feature can be used by all Data Providers that include an XSL transformation ¹.

The following example adds this functionality to the Cobertura Data Provider:

1. Add a `catalog.xml` file in the Data Provider's configuration folder:

```
<configuration>/tools/cobertura/catalog.xml:
<?xml version="1.0"?>
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog">
  <rewriteSystem systemIdStartString="http://cobertura.sourceforge.net/xml"
    rewritePrefix="./DTD"/>
</catalog>
```

2. Copy the `dtd` that the XML needs to validate again inside a `DTD` folder in `<configuration>/tools/cobertura/`.

The catalog file will be used the next time the Data Provider is executed and the DTD declaration will dynamically be changed from:

```
<!DOCTYPE coverage SYSTEM "http://cobertura.sourceforge.net/xml/coverage-04.dtd">
```

to:

```
<!DOCTYPE coverage SYSTEM "<configuration>/tools/cobertura/DTD/
coverage-04.dtd">>
```

For more information about how to write your catalog file, refer to <https://xerces.apache.org/xerces2-j/faq-xcatalogs.html>.

4.64. Creating your own Data Providers

¹The list includes:

- Cantata
- Cobertura
- CodeSonar
- Coverity
- CPD
- CPPCheck
- CPPTest
- FindBugs
- JaCoCo
- Klocwork
- NCover
- Polyspace
- sqlcodeguard

All Data Providers are utilities that run during an analysis. They usually take an input file to parse or parameters specified by the user to generate output files containing data other metrics to add to your project. Here is a non-exhaustive list of what some of them do:

- Use XSLT files to transform XML files
- Read information from microsoft Word Files
- Parse HTML test results
- Query web services
- Export data from OSLC systems
- Launch external processes

This section describes two ways to add your own Data Providers to Squore:

1. Using one of the generic, built-in Data Provider frameworks. Each solution uses a different approach, but the overall goal is to produce one or more CSV files that your Data Provider will send to Squore to associate metrics, findings, textual information or links to artefacts in your project.
2. Writing your own utility to generate the XML files expected by Squore so they can be imported as part of the analysis result.

Tip

If you are interested in developing Data Providers that go beyond the scope of what is described in this manual, consult Squoring Technologies to learn more about the available training courses in writing Data Providers.

4.64.1. Choosing the Right Data Provider Framework

The following is a list of the available Data Provider frameworks:

	Import Metrics	Import Textual Information	Import Findings	Import Links	Create Artefacts	Parse Subfolders
CSV	✓	✓	✗	✗	✓	✓
csv_findings	✗	✗	✓	✗	✗	✗
CSVPerl	✓	✓	✗	✗	✓	✓
Generic	✓	✓	✓	✓	✓	✗
GenericPerl	✓	✓	✓	✓	✓	✓
FindingsPerl	✗	✗	✓	✗	✗	✓
ExcelMetrics	✓	✓	✓	✗	✓	✓

✓ Supported

✓ Your Perl script needs to handle subfolder parsing

✗ Not Supported

Data Provider frameworks and their capabilities

1. Csv

The Csv framework is used to import metrics or textual information and attach them to artefacts of type Application or File. While parsing one or more input CSV files, if it finds the same metric for the same artefact several times, it will only use the last occurrence of the metric and ignore the previous ones. Note that the type of artefacts you can attach metrics to is limited to Application and File artefacts. If you

are working with File artefacts, you can let the Data Provider create the artefacts by itself if they do not exist already. Refer to the full Csv Reference for more information.

2. **csv_findings**

The csv_findings framework is used to import findings in a project and attach them to artefacts of type Application, File or Function. It takes a single CSV file as input and is the only framework that allows you to import relaxed findings directly. Refer to the full csv_findings Reference for more information.

3. **CsvPerl**

The CsvPerl framework offers the same functionality as Csv, but instead of dealing with the raw input files directly, it allows you to run a perl script to modify them and produce a CSV file with the expected input format for the Csv framework. Refer to the full CsvPerl Reference for more information.

4. **FindingsPerl**

The FindingsPerl framework is used to import findings and attach them to existing artefacts. Optionally, if an artefact cannot be found in your project, the finding can be attached to the root node of the project instead. When launching a Data Provider based on the FindingsPerl framework, a perl script is run first. This perl script is used to generate a CSV file with the expected format which will then be parsed by the framework. Refer to the full FindingsPerl Reference for more information.

5. **Generic**

The Generic framework is the most flexible Data Provider framework, since it allows attaching metrics, findings, textual information and links to artefacts. If the artefacts do not exist in your project, they will be created automatically. It takes one or more CSV files as input (one per type of information you want to import) and works with any type of artefact. Refer to the full Generic Reference for more information.

6. **GenericPerl**

The GenericPerl framework is an extension of the Generic framework that starts by running a perl script in order to generate the metrics, findings, information and links files. It is useful if you have an input file whose format needs to be converted to match the one expected by the Generic framework, or if you need to retrieve and modify information exported from a web service on your network. Refer to the full GenericPerl Reference for more information.

7. **ExcelMetrics**

The ExcelMetrics framework is used to extract information from one or more Microsoft Excel files (.xls or .xlsx). A detailed configuration file allows defining how the Excel document should be read and what information should be extracted. This framework allows importing metrics, findings and textual information to existing artefacts or artefacts that will be created by the Data Provider. Refer to the full ExcelMetrics Reference for more information.

4.64.2. Extending a Framework

After you choose the framework to extend, you should follow these steps to make your custom Data Provider known to Squore:

1. Create a new configuration `tools` folder to save your work in your custom configuration folder: `MyConfiguration/configuration/tools`.
2. Create a new folder for your data provider inside the new `tools` folder: **CustomDP**. This folder needs to contain the following files:
 - **form.xml** defines the input parameters for the Data Provider, and the base framework to use, as described in Section 4.64.4, “Data Provider Parameters”
 - **form_en.properties** contains the strings displayed in the web interface for this Data Provider, as described in Section 4.64.5, “Localising your Data Provider”
 - **config.tcl** contains the parameters for your custom Data Provider that are specific to the selected framework
 - **CustomDP.pl** is the perl script that is executed automatically if your custom Data Provider uses one of the *Perl frameworks.

3. Edit Squore Server's configuration file to register your new configuration path, as described in the Installation and Administration Guide.
4. Log into the web interface as a Squore administrator and reload the configuration.

Your new Data Provider is now known to Squore and can be triggered in analyses. Note that you may have to modify your Squore configuration to make your wizard aware of the new Data Provider and your model aware of the new metrics it provides. Refer to the relevant sections of the Configuration Guide for more information.

4.64.3. Creating a Freestyle Data Provider

Instead of using one of the Data Provider frameworks, you can directly produce your results in an XML format that Squore can read and import. The syntax of the XML file to generate is as follows:

```
<!-- input-data.xml: -->
<bundle version="2.0">
  <artifact [local-key="" ] [local-parent="" |parent="" ]>
    <artifact [id="<guid-stable-in-time-also-a-key>" ] name="Component "
    type="REQ" [location="" ]>
      <info name|n="DESCR" value="The description of the object"/>
      <key value="3452-e89b-ff82"/>
      <metric name="TEST_KO" value="2"/>
      <finding name="AR120" loc="xxx" p0="The message" />
      <link name="TEST" local-dst="" |dst="" />
      <artifact id="" name="SubComponent" type="REQ">
        ...
      </artifact>
    </artifact>
  </artifact>

  <artifact id="" local-key="" name="" type="" local-parent="" |
parent="" [location="" ] />
  ...

  <link name="" local-src="" |src="" local-dst="" |dst="" />
  ...

  <info local-ref="" |ref="" name="" value="" />
  ...

  <metric local-ref="" |ref="" name="" value="" />
  ...

  <finding local-ref="" |ref="" [location="" ] p0="" />
  <finding local-ref="" |ref="" [location="" ] p0="">
  <location local-ref="" |ref="" [location="" ] />
  ...
  <relax status="RELAXED_DEROGATION|RELAXED_LEGACY|RELAXED_FALSE_POSITIVE">< !
[CDATA[My Comment]]></relax>
</finding>
  ...
</bundle>
```

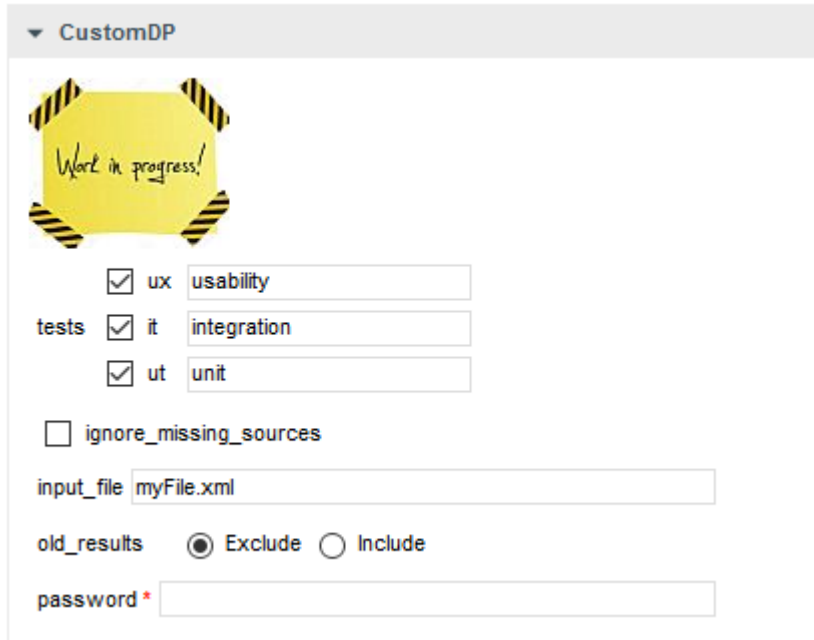
Tip

You can find the XML schema for `input-data.xml` in `input-data-2.xsd`.

`input-data.xml` must be written in a specific location by an executable or a script declared in the Data Provider's `form.xml` in an `exec-phase` element, as described in Section 4.64.4, "Data Provider Parameters".

4.64.4. Data Provider Parameters

A Data Provider's parameters are defined in a file called `form.xml`. The following is an example of `form.xml` for a Data Provider extending the GenericPerl framework:



CustomDP parameters

```
<?xml version="1.0" encoding="UTF-8"?>
<tags baseName="GenericPerl" needSources="true" image="CustomDP.png"
projectStatusOnFailure="ERROR">
  <tag type="multipleChoice" displayType="checkbox" key="tests">
    <value key="ux" option="usability" />
    <value key="it" option="integration" />
    <value key="ut" option="unit" />
  </tag>
  <tag type="booleanChoice" key="ignore_missing_sources" defaultValue="false" />
  <tag type="text" key="input_file" defaultValue="myFile.xml" changeable="false" />
  >
  <tag type="multipleChoice" key="old_results" style="margin-left:10px"
displayType="radioButton" defaultValue="Exclude">
    <value key="Exclude" />
    <value key="Include" />
  </tag>
  <tag type="text" key="java_path" defaultValue="/usr/bin/java" hide="true" />
  <tag type="password" required="true" key="password" />
</tags>
```

The `tags` element accepts the following attributes:

- **baseName (mandatory)** indicates which framework you are basing this Data Provider on
- **needSources (optional, default: false)** allows specifying whether the Data Provider requires sources or not. When set to true, an error will be displayed if you try to select this Data Provider without adding any Repository Connector to your project.

- **image (optional, default: none)** allows displaying a logo in the web UI for the Data Provider
- **projectStatusOnFailure (optional, default: ERROR)** defines what status the project ends in when this Data Provider produces an error. The following values are allowed:
 - **IGNORE**
 - **WARNING**
 - **ERROR**
- **projectStatusOnWarning (optional, default: WARNING)** defines what status the project ends in when this Data Provider produces a warning. The following values are allowed:
 - **IGNORE**
 - **WARNING**
 - **ERROR**

Each **tag** element is a Data Provider option and allows the following attributes:

- **key (mandatory)** is the option's key that will be passed to the perl script, or can be used to specify the parameter's value from the command line
- **type (mandatory)** defines the type of the parameter. The following values are accepted:
 - **text** for free text entry
 - **password** for password fields
 - **booleanChoice** for a boolean
 - **multipleChoice** for offering a selection of predefined values

Note

Predefined values are specified with a **value** element with a mandatory **key** attribute and an optional **option** attribute that allows modifying the value of the option from the UI.

- **displayType (optional)** allows specifying how to display a **multipleChoice** parameter by using one of:
 - **comboBox**
 - **radioButton**
 - **checkbox**
- **defaultValue (optional, default: empty)** is the value used for the parameter when not specified
- **hide (optional, default: false)** allows hiding a parameter from the web UI, which is useful when combining it with a default value
- **changeable (optional, default: true)** allows making a parameter configurable only when creating the project but read-only for following analyses when set to true
- **style (optional, default: empty)** allows setting basic css for the attribute in the web UI
- **required (optional, default: false)** allows showing a red asterisk next to the field in the web UI to make it visibly required. Note that this is only a visual aid at the moment and cannot be used to force users to enter a value for the parameter.

In order to create a freestyle Data Provider, your **form.xml** must contain an **exec-phase** element with a mandatory **id="add-data"** attribute:

```
<exec-phase id="add-data">
  <exec name="tcl|perl|java|javascript or nashorn" | executable="/path/to/bin"
  failOnError="true|false" failOnStdErr="true|false" warn="[WARN]" error="[ERROR|
ERR]" fatal="[FATAL]">
    <arg value="\${<function>(<args>)}"/>
  </exec>
</exec-phase>
```



```

<arg value="-freeText" />
<arg value="{<predefinedVars>}" />
<arg value="versions" />
<arg value="-myTag" />
<arg tag="myTag" />
<env key="MY_VAR" value="SOME_VALUE" />
</exec>
<exec ... />
</exec-phase>
  
```

The `exec-phase` element accepts one or more launches of scripts or executables specified in an `exec` child element, that can receive arguments and environment variables specified via `arg` and `env` elements.

There are four built-in languages for executables:

- **tcl**
- **perl**
- **java**
- **javascript or nashorn**

The scripts are launched using the tcl, perl, or java runtimes defined in your Squore installation. This is also the case for javascript, which is handled by Java's Nashorn engine. Alternatively, you can call an executable directly by specifying its absolute path using the `executable` attribute.

The following attributes of the `exec` element allow you to control error handling:

- **failOnError (optional, default: true)** marks the Data Provider execution as failed if the executable returns an error code
- **failOnStdErr (optional, default: true)** marks the Data Provider execution as failed if the executable prints something to `stderr` during the execution
- **warn**, **error** and **fatal (optional, default: see code block above)** allow you to define patterns to look for in the executable's standard output to fine-tune the result of the execution.

Argument values can be:

1. Free text, useful to specify a parameter for your script
2. A tag `key` declared in `form.xml` to retrieve the input specified by the user
3. One of the predefined functions
 - **getToolConfigDir(<relative/path>)** to get the absolute path of the Data Provider's configuration folder
 - **getToolAddonsDir(<relative/path>)** to get the absolute path of the Data Provider's addons folder
4. One of the predefined variables
 - **{tmpDirectory}** to get an absolute path to a temp folder to create files
 - **{sourcesList}** to get a list of the aliases and locations containing the data extracted by the repository connectors used in the analysis
 - **{outputDirectory}** to get the absolute path of folder where the Data Provider needs to write the final `input-data.xml`

4.64.5. Localising your Data Provider

In order to display your Data Provider parameters in different languages in the web UI, your Data Provider's `form.xml` does not contain any hard-coded strings. Instead, Squore uses each parameter's `key` attribute to dynamically retrieve a translation from a `form_xx.properties` file located next to `form.xml`.

When you create a Data Provider, it is mandatory to include at least an English version of the strings in a file called `form_en.properties`. You are free to add other languages as needed. Here is a sample `.properties` for for the CustomDP you created in the previous section:

```
FORM.GENERAL.NAME = CustomDP
FORM.DASHBOARD.NAME = Test Status
FORM.GENERAL.DESCR = CustomDP imports test results for my project
FORM.GENERAL.URL = http://example.com/CustomDP

TAG.tests.NAME = Test Types
TAG.tests.DESCR = Check the boxes next to the types of test results contained in
the results

TAG.ignore_missing_sources.NAME = Ignore Missing Sources

TAG.input_file.NAME = Test Results
TAG.input_file.DESCR = Specify the absolute path to the file containing the test
results

TAG.old_results.NAME = Old Test Results
TAG.old_results.DESCR = If the previous analysis contained results that are not
in this results file, what do you want to do with the old results?
OPT.Exclude.NAME = discard
OPT.Include.NAME = keep


TAG.password.NAME = File Password
TAG.password.DESCR = Specify the password to decrypt the test results file
```

The syntax for the `.properties` file is as follows:

- **FORM.GENERAL.NAME** is the display name of the Data Provider in the project wizard
- **FORM.DASHBOARD.NAME** is the display name of the Data Provider in the Explorer
- **FORM.GENERAL.DESCR** is the description displayed in the Data Provider's tooltip in the web UI
- **FORM.GENERAL.URL** is a reference URL for the Data Provider. Note that it is not displayed in ther web UI yet.
- **TAG.tag_name.NAME** allows setting the display name of a parameter
- **TAG.tag_name.DESCR** is a help text displayed in a tooltip next to the Data Provider option in the web UI
- **OPT.option_name.NAME** allows setting the display name of an option

Using the `form_en.properties` above for CustomDP results in the following being displayed in the web UI when launching an analysis:

▼ CustomDP



usability

Test Types integration ?

unit

Ignore Missing Sources

Test Results If the previous analysis contained results that are not in this results file, what do you want to do with the old results? ?

Old Test Results discard keep ?

File Password * ?

CustomDP pulling translations from a `.properties` file

5. Cloning Detection

This chapter lists the various metrics collected in Squore when running the cloning detection tool, as well as the violations presented in the Findings tab of the web interface.

Note that the concepts used for cloning detection in Squore are based on the notions of `longest common subsequence problem` (http://en.wikipedia.org/wiki/Longest_common_subsequence_problem) and `longest repeated substring problem` (http://en.wikipedia.org/wiki/Longest_repeated_substring_problem).

5.1. Cloning Metrics

None of the metrics below are set by the cloning detection tool if thresholds are not met. That is, if an artefact has no CC measure in the output file, that does NOT mean that it has no line in common with other artefacts. In models, metrics default to 0 though.

The two main thresholds are:

- A minimum size, to skip small artefacts
- A minimum cloning ratio, to keep only similar artefacts

5.1.1. CC - Code Cloned

Length of the highest Longest Common Substring (LCS) among all cloned artefacts.

Clones are looked in the whole application, in artefacts with the same language and the same type.

- Textual detection, using lines, with trailing spaces removed
- Two artefacts are cloned if they have 90% of lines in common, for $LC \geq 10$

Scope: all artefacts but the root node.

5.1.2. CFTC - Control Flow Token (CFT) Cloned

Length of the highest LCS among all cloned CFT.

Clones are looked in the whole application, in artefacts with the same language and the same type.

- Algorithmic detection, using CFT characters
- Two artefacts are cloned if they have 90% of characters in common, for $CFT \geq 50$

Scope: all artefacts but the root node.

5.1.3. CAC - Children Artefact Cloned

Number of clones in direct children of an artefact.

Parent clones are looked in the whole application, in artefacts with the same language and the same type.

Two classes may have two methods in common, for example, without being cloned. The CAC metric for these two classes will be two (assuming that they only have these two methods in common). Such artefacts should be re-factored (using inheritance for example).

- Use both textual ($CC > 0$) and algorithmic ($CFTC > 0$) cloning when counting
- Two parent artefacts are cloned if 25% of their direct children are cloned

→ Small children artefacts ($LC < 10$) are taken in account, using exact comparison

Scope: all artefacts but the root node.

5.1.4. CN - Clones Number

Number of cloned artefacts.

Clones are looked in the whole application, in artefacts with the same language and the same type.

→ Use both textual ($CC > 0$) and algorithmic ($CFTC > 0$) cloning when counting

Scope: all artefacts but the root node.

5.1.5. RS - Repeated Substrings (Repeated Code Blocks)

Length of all Repeated Substrings in the artefact definition.

That is, duplicated blocks in a function for example.

→ Textual detection, using lines, with trailing spaces removed

→ The metric is triggered if blocks longer than 10 are found, for $LC \geq 10$

Scope: files and all children artefacts.

5.1.6. CFTRS - Repeated Substrings in Control Flow Token

Length of all Repeated Substrings in the artefact CFT.

That is, duplicated algorithmic blocks in a function for example.

→ Algorithmic detection, using CFT characters

→ The metric is triggered if blocks longer than 20 are found, for $CFT \geq 50$

Scope: artefacts with a CFT, like functions.

5.1.7. ICC - Inner Code Cloned

Number of duplicated lines in an artefact.

Clones are looked in all descendants of the artefact. This basically sums all duplicated lines in descendants.

→ Use textual cloning ($CC > 0$) when counting

Scope: all artefacts.

5.1.8. ICFTC - Inner Control Flow Token Cloned

Number of duplicated tokens in an artefact.

Clones are looked in all descendants of the artefact. This basically sums all cloned tokens in descendants.

→ Use algorithmic cloning ($CFTC > 0$) when counting

Scope: all artefacts.

5.2. Cloning Violations

This section lists all the findings that are reported by Squore cloning detection tool.

5.2.1. CC (R_NOCC)

Avoid code duplication.

- Similar artefacts (transitive closure) are part of the same violation
- Use artefacts with textual cloning (CC > 0) when grouping

Scope: files and all children artefacts.

5.2.2. CFTC (R_NOCFTC)

Avoid algorithmic cloning.

- Similar artefacts (transitive closure) are part of the same violation
- Use artefacts with algorithmic cloning (CFTC > 0) when grouping

Scope: artefacts with a CFT, like functions.

5.2.3. CAC (R_NOCAC)

Consider refactorisation.

- Similar artefacts (transitive closure) are part of the same violation
- Use "refactorable" artefacts (CAC > 0) when grouping

Scope: files and all children artefacts.

5.2.4. RS (R_NORS)

Consider refactorisation.

- One violation per "refactorable" artefact (RS > 0)

Scope: files and all children artefacts.

5.2.5. CFTRS (R_NOCFTRS)

Consider algorithmic refactorisation.

- One violation per "refactorable" artefact (CFTRS > 0)

Scope: artefacts with a CFT, like functions.

6. Glossary

6.1. Acceptance Testing

Formal testing conducted to enable a user, customer, or other authorised entity to determine whether to accept a system or component. [SIGIST]

Other Definitions

Acceptance Testing [IEEE 610.12]: Formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system.

See also

Standards:

- SIGIST
- IEEE 610.12

External Links:

- Wikipedia article: Acceptance_testing [http://en.wikipedia.org/wiki/Acceptance_testing]

6.2. Accessibility

Usability of a product, service, environment or facility by people with the widest range of capabilities. [ISO/IEC/IEEE 24765, ISO/IEC 25062]

Notes

- Although "accessibility" typically addresses users who have disabilities, the concept is not limited to disability issues. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC 25062
- ISO/IEC/IEEE 24765

6.3. Accuracy

The capability of the software product to provide the right or agreed results or effects with the needed degree of precision. [ISO/IEC 9126-1]

Other Definitions

Accuracy [ISO/IEC/IEEE 24765]:

1. A qualitative assessment of correctness, or freedom from error.
2. A quantitative measure of the magnitude of error

See also

Standards:

- ISO/IEC 9126-1
- ISO/IEC/IEEE 24765

6.4. Accuracy of Measurement

The closeness of the agreement between the result of a measurement and the true value of the measurand. [ISO/IEC 14143-3, ISO/IEC/IEEE 24765]

Notes

- Accuracy is a qualitative concept. The term precision should not be used for "accuracy"¹. A true value is a value consistent with the definition of a given particular quantity and this is a value that would be obtained by a perfect measurement. In contexts where perfect measurement is not practically feasible, a conventional true value is a value attributed to a particular quantity and accepted, sometimes by convention, as having an uncertainty appropriate for a given purpose. 'Conventional true value', in the same reference, is sometimes called assigned value, best estimate of the value, conventional value or reference value. The accuracy should be expressed in terms of the Mean magnitude of relative error. [ISO/IEC 14143-3]

See also

Standards:

- ISO/IEC 99
- ISO/IEC 14143
- ISO/IEC/IEEE 24765

6.5. Acquirer

Individual or organisation that procures a system, software product, or software service from a supplier. [ISO/IEC 9126-1, ISO/IEC 15939]

Other Definitions

Acquirer [ISO/IEC/IEEE 24765, ISO/IEC 12207]: Stakeholder that acquires or procures a product or service from a supplier.

Acquirer [IEEE 1058, ISO/IEC 15288]: The individual or organization that specifies requirements for and accepts delivery of a new or modified software product and its documentation.

Notes

- The acquirer may be internal or external to the supplier organization. Acquisition of a software product may involve, but does not necessarily require, a legal contract or a financial transaction between the acquirer and supplier. [ISO/IEC/IEEE 24765]
- "buyer", "customer", "owner", "purchaser" are synonyms for acquirer. [ISO/IEC/IEEE 24765]

¹ ISO/IEC 99:2007 International vocabulary of metrology - Basic and general concepts and associated terms

See also

Glossary:

→ Supplier

Standards:

- IEEE 1058
- ISO/IEC 9126-1
- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC 15939
- ISO/IEC/IEEE 24765

6.6. Action

Element of a step that a user performs during a procedure. [ISO/IEC 26514]

See also

Glossary:

- Procedure
- Step
- Task

Standards:

- ISO 5806
- ISO/IEC 26514

6.7. Activity

Any step taken or function performed, both mental and physical, toward achieving some objective. Activities include all the work the managers and technical staff do to perform the tasks of the project and organization. [CMMi]

Other Definitions

Activity [ISO/IEC 12207, ISO/IEC 15288]: Set of cohesive tasks of a process.

Activity [IEEE 1490]: A component of work performed during the course of a project.

Activity [ISO/IEC 14756]: An order submitted to the system under test (SUT) by a user or an emulated user demanding the execution of a data processing operation according to a defined algorithm to produce specific output data from specific input data and (if requested) stored data.

Activity [IEEE 1074]: A defined body of work to be performed, including its required input information and output information

Activity [ISO/IEC 90003]: Collection of related tasks.

Activity [IEEE 829]: Element of work performed during the implementation of a process.

Notes

- An activity normally has an expected duration, cost, and resource requirements. Activities are often subdivided into tasks. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Task

Standards:

- CMMi
- IEEE 829
- IEEE 1074
- IEEE 1490
- ISO/IEC 12207
- ISO/IEC 14756
- ISO/IEC 15288
- ISO/IEC 90003
- ISO/IEC/IEEE 24765

6.8. Actor

A role (with respect to that action) in which the enterprise object fulfilling the role participates in the action. [ISO/IEC 15414]

See also

Standards:

- ISO/IEC 15414

6.9. Adaptability

The capability of the software product to be adapted for different specified environments without applying actions or means other than those provided for this purpose for the software considered. [ISO/IEC 9126-1]

Notes

- Adaptability includes the scalability of internal capacity (e.g. screen fields, tables, transaction volumes, report formats, etc.). [ISO/IEC 9126-1]
- If the software is to be adapted by the end user, adaptability corresponds to suitability for individualisation as defined in ISO 9241-10, and may affect operability. [ISO/IEC 9126-1]

See also

Glossary:

→ Changeability

→ Flexibility

Standards:

→ ISO/IEC 9126-1

→ IEEE 610.12

6.10. Agreement

Mutual acknowledgement of terms and conditions under which a working relationship is conducted. [ISO/IEC 12207, ISO/IEC 15288]

See also

Standards:

→ ISO/IEC 12207

→ ISO/IEC 15288

6.11. Analysability

The capability of the software product to be diagnosed for deficiencies or causes of failures in the software, or for the parts to be modified to be identified. [ISO/IEC 9126-1]

See also

→ ISO/IEC 9126-1

6.12. Analysis Model

Algorithm or calculation combining one or more base and/or derived measures with associated decision criteria. [ISO/IEC 25000]

See also

Standards:

→ ISO/IEC 25000

6.13. Architecture

Fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution. [ISO/IEC 15288]

Notes

→ Architecture sometimes refers to the design of a system's hardware and software components. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC 15288
- ISO/IEC/IEEE 24765

6.14. Attractiveness

The capability of the software product to be attractive to the user. [ISO/IEC 9126-1]

Other Definitions

Other definitions of this word are..

See also

- ISO/IEC 9126-1

6.15. Attribute

A measurable physical or abstract property of an entity. [ISO/IEC 12207, ISO/IEC 14598]

Other Definitions

Attribute [IEEE 610.12]: A characteristic of an item; for example, the item's color, size, or type.

Attribute [ISO/IEC 15939, ISO/IEC 25000]: Inherent property or characteristic of an entity that can be distinguished quantitatively or qualitatively by human or automated means.

Attribute for Quality Measure [ISO/IEC 25000]: Attribute that relates to software product itself, to the use of the software product or to its development process.

Notes

- Can refer either to general characteristics such as reliability, maintainability, and usability or to specific features of a software product. ISO 9000 distinguishes two types of attributes: a permanent characteristic existing inherently in something; and an assigned characteristic of a product, process or system (e.g. the price of a product, the owner of a product). The assigned characteristic is not an inherent quality characteristic of that product, process or system. An attribute expresses some characteristic that is generally common to the instances of a class. The name of the attribute is the name of the role that the value class plays in describing the class, which may simply be the name of the value class (as long as using the value class name does not cause ambiguity). [ISO/IEC/IEEE 24765]
- Attributes for quality measure are used in order to obtain quality measure elements. [ISO/IEC/IEEE 24765]

See also

Glossary:

- External Attribute
- Internal Attribute
- Optional Attribute

Standards:

- IEEE 610.12
- ISO/IEC 12207
- ISO/IEC 14598
- ISO/IEC 15939
- ISO/IEC/IEEE 24765

6.16. Availability

The degree to which a system or component is operational and accessible when required for use. [ISO/IEC 20000]

Other Definitions

Availability [ISO/IEC 20000]: Ability of a component or service to perform its required function at a stated instant or over a stated period of time.

Notes

- Often expressed as a probability. Availability is usually expressed as a ratio of the time that the service is actually available for use by the business to the agreed service hours. [ISO/IEC 25000]

See also

Glossary:

- Fault Tolerance

Standards:

- ISO/IEC 25000

6.17. Base Measure

Measure defined in terms of an attribute and the method for quantifying it. [ISO/IEC 99, ISO/IEC 15939, ISO/IEC 25000]

Notes

- A base measure is functionally independent of other measures. [ISO/IEC 15939]

See also

Glossary:

- Measure
- Derived Measure
- Direct Measure
- External Measure
- Indirect Measure
- Internal Measure

Standards:

- ISO/IEC 99
- ISO/IEC 15939
- ISO/IEC 25000

6.18. Baseline

Formally approved version of a configuration item, regardless of media, formally designated and fixed at a specific time during the configuration item's life cycle. [ISO/IEC 19770-1]

Other Definitions

Baseline [ISO/IEC 12207, ISO/IEC 15288]: Specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures.

Baseline [ISO/IEC 20000]: Snapshot of the state of a service or individual configuration items at a point in time.

Baseline [IEEE 1490]: An approved plan (for a project), plus or minus approved changes. It is compared to actual performance to determine if performance is within acceptable variance thresholds. Generally refers to the current baseline, but may refer to the original or some other baseline. Usually used with a modifier (e.g., cost performance baseline, schedule baseline, performance measurement baseline, technical baseline).

Notes

- A baseline should be changed only through formal configuration management procedures. Some baselines may be project deliverables while others provide the basis for further work. Baselines, plus approved changes from those baselines, constitute the current configuration identification. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Configuration Management

Standards:

- IEEE 1490
- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC 20000
- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.19. Branch

A Branch is either:

- a conditional transfer of control from any statement to any other statement in a component, or
- an unconditional transfer of control from any statement to any other statement in the component except the next statement, or

- when a component has more than one entry point, a transfer of control to an entry point of the component. [SIGIST]

Other Definitions

Branch [ISO/IEC/IEEE 24765]:

1. a computer program construct in which one of two or more alternative sets of program statements is selected for execution.
2. a point in a computer program at which one of two or more alternative sets of program statements is selected for execution.
3. to perform the selection in (1).
4. any of the alternative sets of program statements in (1).
5. a set of evolving source file versions.

Notes

- Every branch is identified by a tag. Often, a branch identifies the file versions that have been or will be released as a product release. May denote unbundling of arrow meaning, i.e., the separation of object types from an object type set. Also refers to an arrow segment into which a root arrow segment has been divided. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Branch Coverage

Standards:

- SIGIST
- ISO/IEC/IEEE 24765

6.20. Branch Coverage

The percentage of branches that have been exercised by a test case suite. [SIGIST]

See also

Glossary:

- Branch
- Coverage

Standards:

- SIGIST

6.21. Branch Testing

Testing designed to execute each outcome of each decision point in a computer program. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Path Testing
- Statement Testing
- Testing

Standards:

- ISO/IEC/IEEE 24765

6.22. Budget

The approved estimate for the project or any work breakdown structure component or any schedule activity. [IEEE 1490]

Notes

- Often used also to refer to work effort as well as, or instead of, money. [IEEE 1490]

See also

Standards:

- IEEE 1490

6.23. Build

An operational version of a system or component that incorporates a specified subset of the capabilities that the final product will provide. [IEEE 610.12, ISO/IEC/IEEE 24765]

See also

Standards:

- IEEE 610.12
- ISO/IEC/IEEE 24765

6.24. Call Graph

A diagram that identifies the modules in a system or computer program and shows which modules call one another. [ISO/IEC/IEEE 24765]

Notes

- The result is not necessarily the same as that shown in a structure chart. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.25. Capability Maturity Model

Model that contains the essential elements of effective processes for one or more disciplines and describes an evolutionary improvement path from ad hoc, immature processes to disciplined, mature processes with improved quality and effectiveness. [ISO/IEC/IEEE 24765]

Other Definitions

Capability Maturity Model [CMMi]: A description of the stages through which software organizations evolve as they define, implement, measure, control, and improve their software processes. This model provides a guide for selecting process improvement strategies by facilitating the determination of current process capabilities and the identification of the issues most critical to software quality and process improvement.

See also

Standards:

- CMMi
- ISO/IEC/IEEE 24765

6.26. Certification

A formal demonstration that a system or component complies with its specified requirements and is acceptable for operational use. [ISO/IEC/IEEE 24765]

Other Definitions

Certification [ISO/IEC/IEEE 24765]:

1. A written guarantee that a system or component complies with its specified requirements and is acceptable for operational use.
2. The process of confirming that a system or component complies with its specified requirements and is acceptable for operational use.

Example

- A written authorization that a computer system is secure and is permitted to operate in a defined environment. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Certification Criteria

Standards:

- ISO/IEC/IEEE 24765

6.27. Certification Criteria

A set of standards, rules, or properties to which an asset must conform in order to be certified to a certain level. [ISO/IEC/IEEE 24765]

Notes

- Certification criteria are defined by a certification policy. Certification criteria may be specified as a set of certification properties that must be met. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Certification

Standards:

- ISO/IEC/IEEE 24765

6.28. Change Control Board

A formally constituted group of stakeholders responsible for reviewing, evaluating, approving, delaying, or rejecting changes to a project, with all decisions and recommendations being recorded. [IEEE 1490]

See also

Standards:

- IEEE 1490

6.29. Change Control System

A collection of formal documented procedures that define how project deliverables and documentation will be controlled, changed, and approved. [IEEE 1490, ISO/IEC/IEEE 24765]

Notes

- In most application areas, the change control system is a subset of the configuration management system. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Change Management
- Configuration Management

Standards:

- IEEE 1490
- ISO/IEC/IEEE 24765

6.30. Change Management

Judicious use of means to effect a change, or a proposed change, to a product or service. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Change Control System
- Configuration Management

Standards:

- ISO/IEC/IEEE 24765

6.31. Changeability

The capability of the software product to enable a specified modification to be implemented. [ISO/IEC 9126-1]

Notes

- Implementation includes coding, designing and documenting changes. [ISO/IEC 9126-1]
- If the software is to be modified by the end user, changeability may affect operability. [ISO/IEC 9126-1]

See also

Glossary:

- Flexibility

Standards:

- ISO/IEC 9126-1

6.32. Co-existence

The capability of the software product to co-exist with other independent software in a common environment sharing common resources. [ISO/IEC 9126-1]

See also

Standards:

- ISO/IEC 9126-1

6.33. Code

In software engineering, computer instructions and data definitions expressed in a programming language or in a form output by an assembler, compiler, or other translator. [ISO/IEC/IEEE 24765]

Other Definitions

Code (verb) [ISO/IEC/IEEE 24765]: To express a computer program in a programming language.

See also

Glossary:

→ Coding

Standards:

→ ISO/IEC/IEEE 24765

6.34. Code Coverage

An analysis method that determines which parts of the software have been executed (covered) by the test case suite and which parts have not been executed and therefore may require additional attention. [SIGIST]

See also

Glossary:

→ Coverage

Standards:

→ SIGIST

6.35. Code Freeze

A period during which non-critical changes to the code are not allowed. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Code

→ Feature Freeze

Standards:

→ ISO/IEC/IEEE 24765

6.36. Code Review

A meeting at which software code is presented to project personnel, managers, users, customers, or other interested parties for comment or approval. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Code

→ Coding

Standards:

→ ISO/IEC/IEEE 24765

6.37. Code Verification

Ensures by static verification methods the conformance of source code to the specified design of the software module, the required coding standards, and the safety planning requirements. [IEC 61508-3]

See also

→ IEC 61508-3

6.38. Coding

In software engineering, the process of expressing a computer program in a programming language. [ISO/IEC/IEEE 24765]

Other Definitions

Coding [ISO/IEC/IEEE 24765]: The transforming of logic and data from design specifications (design descriptions) into a programming language.

See also

Glossary:

→ Code

Standards:

→ ISO/IEC/IEEE 24765

6.39. Cohesion

In software design, a measure of the strength of association of the elements within a module. [ISO/IEC/IEEE 24765]

Other Definitions

Cohesion [ISO/IEC/IEEE 24765]: The manner and degree to which the tasks performed by a single software module are related to one another.

Notes

→ Types include coincidental, communicational, functional, logical, procedural, sequential, and temporal. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Coupling

Standards:

→ ISO/IEC/IEEE 24765

6.40. Commercial-Off-The-Shelf (COTS)

Software defined by a market-driven need, commercially available, and whose fitness for use has been demonstrated by a broad spectrum of commercial users. [ISO/IEC 25051]

Notes

- COTS software product includes the product description (including all cover information, data sheet, web site information, etc.), the user documentation (necessary to install and use the software), the software contained on a computer sensible media (disk, CD-ROM, internet downloadable, etc.). Software is mainly composed of programs and data. This definition applies also to product descriptions, user documentation and software which are produced and supported as separate manufactured goods, but for which typical commercial fees and licensing considerations may not apply. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Software Product

Standards:

- ISO/IEC 15289
- ISO/IEC 25051
- ISO/IEC 90003
- ISO/IEC/IEEE 24765

6.41. Commit

To integrate the changes made to a developer's private view of the source code into a branch accessible through the version control system's repository. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Configuration Management
- Software Repository

Standards:

- ISO/IEC/IEEE 24765

6.42. Commitment

An action resulting in an obligation by one or more of the participants in the act to comply with a rule or perform a contract. [ISO/IEC 15414]

Other Definitions

Commitment [CMMi]: A pact that is freely assumed, visible, and expected to be kept by all parties.

Notes

- The enterprise object(s) participating in an action of commitment may be parties or agents acting on behalf of a party or parties. In the case of an action of commitment by an agent, the principal becomes obligated. [ISO/IEC/IEEE 24765]

See also

Standards:

- CMMi
- ISO/IEC 15414
- ISO/IEC/IEEE 24765

6.43. Compatibility

The ability of two or more systems or components to perform their required functions while sharing the same hardware or software environment. [ISO/IEC/IEEE 24765]

Other Definitions

Compatibility [ISO/IEC/IEEE 24765]: The ability of two or more systems or components to exchange information.

Compatibility [ISO/IEC 2382-1]: The capability of a functional unit to meet the requirements of a specified interface without appreciable modification.

See also

Standards:

- ISO/IEC 2382-1
- ISO/IEC/IEEE 24765

6.44. Complexity

The degree to which a system's design or code is difficult to understand because of numerous components or relationships among components. [ISO/IEC/IEEE 24765]

Other Definitions

Complexity [ISO/IEC/IEEE 24765]: The degree to which a system or component has a design or implementation that is difficult to understand and verify.

See also

Glossary:

- Maintainability

Standards:

- ISO/IEC/IEEE 24765

6.45. Component

An entity with discrete structure, such as an assembly or software module, within a system considered at a particular level of analysis. [ISO/IEC 15026]

Other Definitions

Component [SIGIST]: A minimal software item for which a separate specification is available.

Component [IEEE 829]: One of the parts that make up a system.

Component [ISO/IEC 29881]: Set of functional services in the software, which, when implemented, represents a well-defined set of functions and is distinguishable by a unique name.

Software Component [IEEE 1061]: A general term used to refer to a software system or an element, such as module, unit, data, or document.

Software Component [ISO/IEC/IEEE 24765]: A functionally or logically distinct part of a software configuration item, distinguished for the purpose of convenience in designing and specifying a complex SCI as an assembly of subordinate elements.

Notes

- A component may be hardware or software and may be subdivided into other components. The terms "module," "component," and "unit" are often used interchangeably or defined to be sub-elements of one another in different ways depending upon the context. The relationship of these terms is not yet standardized. A component may or may not be independently managed from the end-user or administrator's point of view. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Configuration Item

Standards:

- IEEE 829
- IEEE 1061
- ISO/IEC 15026
- ISO/IEC 29881
- ISO/IEC/IEEE 24765
- SIGIST

6.46. Conciseness

Software attributes that provide implementation of a function with a minimum amount of code. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.47. Condition

A boolean expression containing no boolean operators. For instance $A < B$ is a condition but A and B is not. [RTCA/EUROCAE]

Other Definitions

Condition [ISO 5806, ISO/IEC/IEEE 24765]: a description of a contingency to be considered in the representation of a problem, or a reference to other procedures to be considered as part of the condition.

See also

Standards:

- RTCA/EUROCAE
- ISO/IEC/IEEE 24765

6.48. Configuration

The arrangement of a computer system or component as defined by the number, nature, and interconnections of its constituent parts. [ISO/IEC/IEEE 24765]

Other Definitions

Configuration [ISO/IEC/IEEE 24765]: In configuration management, the functional and physical characteristics of hardware or software as set forth in technical documentation or achieved in a product.

Configuration [ISO/IEC/IEEE 24765]: The arrangement of a system or network as defined by the nature, number, and chief characteristics of its functional units.

Configuration [ISO/IEC/IEEE 24765]: The requirements, design, and implementation that define a particular version of a system or system component.

Configuration [ISO/IEC 2382-1]: The manner in which the hardware and software of an information processing system are organized and interconnected.

See also

Glossary:

- Configuration Item
- Configuration Management

Standards:

- ISO/IEC 2382-1
- ISO/IEC/IEEE 24765

6.49. Configuration Control

An element of configuration management, consisting of the evaluation, coordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification. [IEEE 610.12, ISO/IEC/IEEE 24765]

See also

Glossary:

→ Configuration Management

Standards:

→ IEEE 610.12

→ ISO/IEC/IEEE 24765

6.50. Configuration Item

Entity within a configuration that satisfies an end use function and that can be uniquely identified at a given reference point. [ISO/IEC 12207]

Other Definitions

Configuration Item [ISO/IEC 19770]: Item or aggregation of hardware or software or both that is designed to be managed as a single entity.

Configuration Item [ISO/IEC 20000-1]: Component of an infrastructure or an item which is, or will be, under the control of configuration management.

Configuration Item [ISO/IEC/IEEE 24765]: An aggregation of hardware, software, or both, that is designated for configuration management and treated as a single entity in the configuration management process.

Configuration Item [ISO/IEC/IEEE 24765]: Aggregation of work products that is designated for configuration management and treated as a single entity in the configuration management process.

Software Configuration Item [ISO/IEC/IEEE 24765]: A software entity that has been established as a configuration item.

Notes

→ Configuration items may vary widely in complexity, size and type, ranging from an entire system including all hardware, software and documentation, to a single module or a minor hardware component. [ISO/IEC/IEEE 24765]

→ The SCI (Software Configuration Item) exists where functional allocations have been made that clearly distinguish equipment functions from software functions and where the software has been established as a configurable item. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Configuration

→ Configuration Management

Standards:

- ISO/IEC 12207
- ISO/IEC 19770
- ISO/IEC 20000
- ISO/IEC/IEEE 24765

6.51. Configuration Management

A discipline applying technical and administrative direction and surveillance to

- identify and document the functional and physical characteristics of a configuration item,
- control changes to those characteristics, record and report change processing and implementation status, and
- verify compliance with specified requirements. [IEEE 610.12, ISO/IEC/IEEE 24765]

Other Definitions

Configuration Management <ref name="sting">Software Technology Interest Group On-line Glossary, <http://www.apl.jhu.edu/Notes/Hausler/web/glossary.html> .</ref>: The process of identifying, defining, recording and reporting the configuration items in a system and the change requests. Controlling the releases and change of the items throughout the life-cycle.

Configuration Management [ISO/IEC 29881]: Technical and organizational activities comprising configuration identification, control, status accounting, and auditing.

Software Configuration Management [ISO/IEC 15846]: The process of applying configuration management throughout the software life cycle to ensure the completeness and correctness of Software Configuration Items.

See also

Glossary:

- Change Management
- Configuration Control
- Configuration Item

Standards:

- IEEE 610.12
- ISO/IEC 15846
- ISO/IEC 29881
- ISO/IEC/IEEE 24765

External Links:

- Wikipedia article: Software Configuration Management [http://en.wikipedia.org/wiki/Software_configuration_management]

6.52. Configuration Management System

The discipline of identifying the components of a continually evolving system to control changes to those components and maintaining integrity and traceability throughout the life cycle. [ISO/IEC/IEEE 24765]

Notes

- A subsystem of the overall project management system. It is a collection of formal documented procedures used to apply technical and administrative direction and surveillance to:
- * identify and document the functional and physical characteristics of a product, result, service, or component;
- * control any changes to such characteristics;
- * record and report each change and its implementation status; and
- * support the audit of the products, results, or components to verify conformance to requirements.

:It includes the documentation, tracking systems, and defined approval levels necessary for authorizing and controlling changes. [IEEE 1490]

See also

Glossary:

- Configuration Management

Standards:

- IEEE 1490
- ISO/IEC/IEEE 24765

6.53. Conflict

A change in one version of a file that cannot be reconciled with the version of the file to which it is applied. [ISO/IEC/IEEE 24765]

Notes

- Conflicts can occur when versions from different branches are merged or when two committers work concurrently on the same file. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Version

Standards:

- ISO/IEC/IEEE 24765

6.54. Conformance

The fulfillment by a product, process or service of specified requirements. [ISO/IEC 12207]

See also

Glossary:

- Requirement

Standards:

→ ISO/IEC 12207

6.55. Connectivity

The capability of a system or device to be attached to other systems or devices without modification. [ISO/IEC 2382-1]

See also

Standards:

→ ISO/IEC 2382-1

6.56. Consistency

The degree of uniformity, standardization, and freedom from contradiction among the documents or parts of a system or component. [ISO/IEC/IEEE 24765]

Other Definitions

Consistency [ISO/IEC/IEEE 24765]: Software attributes that provide uniform design and implementation techniques and notations.

See also

Standards:

→ ISO/IEC/IEEE 24765

6.57. Constraint

A restriction on the value of an attribute or the existence of any object based on the value or existence of one or more others. [ISO/IEC 15474-1]

Other Definitions

Constraint [IEEE 1362]: An externally imposed limitation on system requirements, design, or implementation or on the process used to develop or modify a system.

Constraint [IEEE 1490]: The state, quality, or sense of being restricted to a given course of action or inaction. An applicable restriction or limitation, either internal or external to a project, which will affect the performance of the project or a process. For example, a schedule constraint is any limitation or restraint placed on the project schedule that affects when a schedule activity can be scheduled and is usually in the form of fixed imposed dates.

Constraint [IEEE 1233]: A statement that expresses measurable bounds for an element or function of the system.

Notes

→ That is, a constraint is a factor that is imposed on the solution by force or compulsion and may limit or modify the design changes. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Requirement

Standards:

→ IEEE 1233

→ IEEE 1362

→ IEEE 1490

→ ISO/IEC 15474-1

→ ISO/IEC/IEEE 24765

6.58. Content Coupling

A type of coupling in which some or all of the contents of one software module are included in the contents of another module. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Control Coupling

→ Coupling

→ Data Coupling

→ Hybrid Coupling

→ Pathological Coupling

Standards:

→ ISO/IEC/IEEE 24765

6.59. Context of Use

Users, tasks, equipment (hardware, software and materials), and the physical and social environments in which a product is used. [ISO/IEC 25000]

See also

Glossary:

→ Environment

Standards:

→ ISO/IEC 25000

6.60. Contract

Binding agreement between two parties, especially enforceable by law, or a similar internal agreement wholly within an organization. [ISO/IEC 12207]

Other Definitions

Contract [IEEE 1490]: A mutually binding agreement that obligates the seller to provide the specified product or service or result and obligates the buyer to pay for it.

See also

Standards:

- IEEE 1490
- ISO/IEC 12207

6.61. Control Coupling

A type of coupling in which one software module communicates information to another module for the explicit purpose of influencing the latter module's execution. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Content Coupling
- Coupling
- Data Coupling
- Hybrid Coupling
- Pathological Coupling

Standards:

- ISO/IEC/IEEE 24765

6.62. Control Flow

The sequence in which operations are performed during the execution of a computer program. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Control Flow Diagram
- Data Flow

Standards:

- ISO/IEC/IEEE 24765

6.63. Control Flow Diagram

A diagram that depicts the set of all possible sequences in which operations may be performed during the execution of a system or program. [ISO/IEC/IEEE 24765]

Notes

→ Types include box diagram, flowchart, input-process-output chart, state diagram. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Control Flow

→ Data Flow Diagram

Standards:

→ ISO/IEC/IEEE 24765

6.64. Convention

Requirement employed to prescribe a disciplined, uniform approach to providing consistency in a software product, that is, a uniform pattern or form for arranging data. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.65. Correctability

The degree of effort required to correct software defects and to cope with user complaints. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Maintainability

Standards:

→ ISO/IEC/IEEE 24765

6.66. Correctness

The degree to which a system or component is free from faults in its specification, design, and implementation. [ISO/IEC/IEEE 24765]

Other Definitions

Correctness [ISO/IEC/IEEE 24765]: The degree to which software, documentation, or other items meet specified requirements.

Correctness [ISO/IEC/IEEE 24765]: The degree to which software, documentation, or other items meet user needs and expectations, whether specified or not.

See also

Glossary:

→ Fault

Standards:

→ ISO/IEC/IEEE 24765

6.67. Coupling

The manner and degree of interdependence between software modules. [ISO/IEC/IEEE 24765]

Other Definitions

Coupling [ISO/IEC 19759]: The strength of the relationships between modules.

Coupling [ISO/IEC/IEEE 24765]: A measure of how closely connected two routines or modules are.

Coupling [ISO/IEC/IEEE 24765]: In software design, a measure of the interdependence among modules in a computer program

Notes

→ Types include common-environment coupling, content coupling, control coupling, data coupling, hybrid coupling, and pathological coupling. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Cohesion

→ Control Coupling

→ Data Coupling

Standards:

→ ISO/IEC 19759

→ ISO/IEC/IEEE 24765

6.68. Coverage

The degree, expressed as a percentage, to which a specified coverage item has been exercised by a test case suite. [SIGIST]

Other Definitions

Test Coverage [ISO/IEC 12207]: Extent to which the test cases test the requirements for the system or software product.

test Coverage [ISO/IEC/IEEE 24765]: The degree to which a given test or set of tests addresses all specified requirements for a given system or

component.

See also

Glossary:

- Code Coverage
- Test Case
- Test Case Suite

Standards:

- ISO/IEC 12207
- ISO/IEC/IEEE 24765
- SIGIST

6.69. Criteria

Specific data items identified as contents of information items for appraising a factor in an evaluation, audit, test or review. [ISO/IEC 15289]

Other Definitions

Criteria [ISO/IEC 15289]: standards, rules, or tests on which a judgment or decision can be based, or by which a product, service, result, or process can be evaluated.

See also

Glossary:

- Certification Criteria
- Decision Criteria

Standards:

- ISO/IEC 15289

6.70. Criticality

The degree to which a system or component is operational and accessible when required for use. [IEEE 829]

See also

Standards:

- IEEE 829

6.71. Custom Software

Software product developed for a specific application from a user requirements specification. [ISO/IEC 25000]

See also

Glossary:

- Product
- Requirement

Standards:

- ISO/IEC 25000

6.72. Customer

Organization or person that receives a product or service. [ISO/IEC 12207, ISO/IEC 15288]

Other Definitions

Customer [IEEE 1233]: The entity or entities for whom the requirements are to be satisfied in the system being defined and developed.

Customer [IEEE 1362]: An individual or organization who acts for the ultimate user of a new or modified hardware or software product to acquire the product and its documentation.

Customer [IEEE 830]: The person, or persons, who pay for the product and usually (but not necessarily) decide the requirements.

Notes

- Synonyms of Customer are: acquirer, buyer, beneficiary, purchaser. [ISO/IEC/IEEE 24765]
- A customer can be internal or external to the organization. The customer may be a higher level project. This is the entity to whom the system developer must provide proof that the system developed satisfies the system requirements specified. Customers are a subset of stakeholders. [ISO/IEC/IEEE 24765]
- Example: an end-user of the completed system, an organization within the same company as the developing organization (e.g., System Management). [ISO/IEC/IEEE 24765]

See also

Glossary:

- Stakeholder

Standards:

- IEEE 830
- IEEE 1233
- IEEE 1362
- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC/IEEE 24765

6.73. Data

Collection of values assigned to base measures, derived measures, and/or indicators. [ISO/IEC 15939, ISO/IEC 25000]

Other Definitions

Data [ISO/IEC/IEEE 24765]: A representation of facts, concepts, or instructions in a manner suitable for communication, interpretation, or processing by humans or by automatic means.

Data [ISO/IEC 2382-1]: A reinterpretable representation of information in a formalized manner suitable for communication, interpretation, or communication, or processing.

See also

Glossary:

- Base Measure
- Derived Measure
- Indicator

Standards:

- ISO/IEC 2382-1
- ISO/IEC 15939
- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.74. Data Coupling

A type of coupling in which output from one software module serves as input to another module. [ISO/IEC/IEEE 24765]

Notes

- Synonym for Data Coupling: input-output coupling. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Content Coupling
- Control Coupling
- Coupling
- Hybrid Coupling
- Pathological Coupling

Standards:

- ISO/IEC/IEEE 24765

6.75. Data Flow

The sequence in which data transfer, use, and transformation are performed during the execution of a computer program. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Control Flow

Standards:

→ ISO/IEC/IEEE 24765

6.76. Data Flow Diagram

A diagram that depicts data sources, data sinks, data storage, and processes performed on data as nodes, and logical flow of data as links between the nodes. [ISO/IEC/IEEE 24765]

Notes

→ Synonyms for Data Flow Diagram: data flowchart, data flow graph. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Data Flow

→ Control Flow Diagram

Standards:

→ ISO/IEC/IEEE 24765

6.77. Data Management

In a data processing system, the functions that provide access to data, perform or monitor the storage of data, and control input-output operations. [ISO/IEC 2382-1]

Other Definitions

Data Management [ISO/IEC/IEEE 24765]: The disciplined processes and systems that plan for, acquire, and provide stewardship for business and technical data, consistent with data requirements, throughout the data lifecycle.

See also

Glossary:

→ Data

Standards:

→ ISO/IEC 2382-1

→ ISO/IEC/IEEE 24765

6.78. Data Model

A model about data by which an interpretation of the data can be obtained in the modeling tool industry. [ISO/IEC 15474-1]

Notes

- A data model is one that may be encoded and manipulated by a computer. A data model identifies the entities, domains (attributes), and relationships (associations) with other data and provides the conceptual view of the data and the relationships among data. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Data
- Model

Standards:

- ISO/IEC 15474-1
- ISO/IEC/IEEE 24765

6.79. Data Processing

The systematic performance of operations upon data. [ISO/IEC 2382-1]

Notes

- Example: arithmetic or logic operations upon data, merging or sorting of data, assembling or compiling of programs, or operations on text, such as editing, sorting, merging, storing, retrieving, displaying, or printing. [ISO/IEC/IEEE 24765]
- The term data processing should not be used as a synonym for information processing. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Data

Standards:

- ISO/IEC 2382-1
- ISO/IEC/IEEE 24765

6.80. Data Provider

Individual or organisation that is a source of data. [ISO/IEC 15939]

See also

Standards:

- ISO/IEC 15939

6.81. Data Store

Organised and persistent collection of data and information that allows for its retrieval. [ISO/IEC 15939]

See also

Standards:

→ ISO/IEC 15939

6.82. Data Type

A class of data, characterized by the members of the class and the operations that can be applied to them. [ISO/IEC/IEEE 24765]

Notes

→ Example: integers, real numbers, and character strings. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Data

Standards:

→ ISO/IEC/IEEE 24765

6.83. Database

A collection of data organized according to a conceptual structure describing the characteristics of the data and the relationships among their corresponding entities, supporting one or more application areas. [ISO/IEC 2382-1]

Other Definitions

Database [ISO/IEC/IEEE 24765]: A collection of interrelated data stored together in one or more computerized files.

Database [ISO/IEC 29881]: Collection of data describing a specific target area that is used and updated by one or more applications.

See also

Glossary:

→ Data

→ Data Store

Standards:

→ ISO/IEC 2382-1

→ ISO/IEC 29881

→ ISO/IEC/IEEE 24765

6.84. Decision Criteria

Thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result. [ISO/IEC 15939, ISO/IEC 25000]

See also

Glossary:

→ Criteria

Standards:

→ ISO/IEC 15939

→ ISO/IEC 25000

6.85. Decoupling

The process of making software modules more independent of one another to decrease the impact of changes to, and errors in, the individual modules. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Coupling

Standards:

→ ISO/IEC/IEEE 24765

6.86. Defect

A flaw in a system or system component that causes the system or component to fail to perform its required function. A defect, if encountered during execution, may cause a failure of the system. [CMMi]

Other Definitions

Defect [IEEE 1490]: An imperfection or deficiency in a project component where that component does not meet its requirements or specifications and needs to be either repaired or replaced.

Notes

→ (1) Omissions and imperfections found during early life cycle phases and (2) faults contained in software sufficiently mature for test or operation. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Failure

→ Fault

→ Defect Density

Standards:

→ CMMi

→ IEEE 1490

6.87. Degree of Confidence

The degree of confidence that software conforms to its requirements. [ISO/IEC 15026]

See also

Standards:

→ ISO/IEC 15026

6.88. Deliverable

Items whose delivery to the customer is a requirement of the contract. [ISO/IEC 15910]

Other Definitions

Deliverable [IEEE 1490]: Any unique and verifiable product, result, or capability to perform a service that must be produced to complete a process, phase, or project. Often used more narrowly in reference to an external deliverable, which is a deliverable that is subject to approval by the project sponsor or customer.

Deliverable [ISO/IEC/IEEE 24765]: Item² to be provided to an acquirer or other designated recipient as specified in an agreement.

Deliverables [ISO/IEC 15910]: Items whose delivery to the customer is a requirement of the contract.

See also

Glossary:

→ Software Product

Standards:

→ IEEE 1490

→ ISO/IEC 15910

→ ISO/IEC/IEEE 24765

6.89. Delivery

Release of a system or component to its customer or intended user. [ISO/IEC/IEEE 24765]

See also

Glossary:

²This item can be a document, hardware item, software item, service, or any type of work product.

→ Deliverable

Standards:

→ ISO/IEC/IEEE 24765

6.90. Dependability

Measure of the degree to which an item is operable and capable of performing its required function at any (random) time during a specified mission profile, given item availability at the start of the mission. [ISO/IEC/IEEE 24765]

Other Definitions

Dependability [IEEE 982]: Trustworthiness of a computer system such that reliance can be justifiably placed on the service it delivers.

Notes

→ Reliability, availability, and maintainability are aspects of dependability. [ISO/IEC/IEEE 24765]

See also

Standards:

→ IEEE 982

→ ISO/IEC/IEEE 24765

6.91. Deployment

Phase of a project in which a system is put into operation and cutover issues are resolved. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.92. Derived Measure

Measure that is defined as a function of two or more values of base measures. [ISO/IEC 99, ISO/IEC 15939, ISO/IEC 25000]

Notes

→ A derived measure is a measure that is defined as a function of two or more values of base measures. Derived measures capture information about more than one attribute or the same attribute from multiple entities. Simple transformations of base measures (for example, taking the square root of a base measure) do not add information, thus do not produce derived measures. Normalisation of data often involves converting base measures into derived measures that can be used to compare different entities. [ISO/IEC 15939]

→ A transformation of a base measure using a mathematical function can also be considered as a derived. [ISO/IEC 25000]

See also

Glossary:

- Measure
- Base Measure
- Direct Measure
- Indirect Measure

Standards:

- ISO/IEC 99
- ISO/IEC 15939
- ISO/IEC 25000

6.93. Design

The process of defining the architecture, components, interfaces, and other characteristics of a system or component. [ISO/IEC/IEEE 24765]

Other Definitions

Design [ISO/IEC/IEEE 24765]: The result of the process of defining the architecture, components, interfaces, and other characteristics of a system or component.

Design [ISO/IEC/IEEE 24765]: The process of defining the software architecture, components, modules, interfaces, and data for a software system to satisfy specified requirements.

Design [ISO/IEC/IEEE 24765]: The process of conceiving, inventing, or contriving a scheme for turning a computer program specification into an operational program.

Design [ISO/IEC/IEEE 24765]: Activity that links requirements analysis to coding and debugging.

Design [ISO/IEC 26514]: Stage of documentation development that is concerned with determining what documentation will be provided in a product and what the nature of the documentation will be.

See also

Standards:

- ISO/IEC/IEEE 24765

6.94. Design Pattern

A description of the problem and the essence of its solution to enable the solution to be reused in different settings. [ISO/IEC/IEEE 24765]

Notes

- Not a detailed specification, but a description of accumulated wisdom and experience. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.95. Developer

Individual or organisation that performs development activities (including requirements analysis, design, testing through acceptance) during the software lifecycle process. [ISO/IEC 12207, ISO/IEC 9126-1]

Notes

- May include new development, modification, reuse, reengineering, maintenance, or any other activity that results in software products, and includes the testing, quality assurance, configuration management, and other activities applied to these products. Developers apply methodologies via enactment. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC 9126-1
- ISO/IEC 12207
- ISO/IEC/IEEE 24765

6.96. Development

Software life cycle process that contains the activities of requirements analysis, design, coding, integration, testing, installation and support for acceptance of software products. [ISO/IEC 90003, ISO/IEC 12207, ISO/IEC 15939]

Other Definitions

Development [ISO/IEC 26514]: Activity of preparing documentation after it has been designed.

See also

Glossary:

- Developer
- Development Testing
- Process
- Software Life Cycle

Standards:

- ISO/IEC 12207

- ISO/IEC 15939
- ISO/IEC 26514
- ISO/IEC 90003

6.97. Development Testing

Formal or informal testing conducted during the development of a system or component, usually in the development environment by the developer. [ISO/IEC/IEEE 24765]

Other Definitions

Development Testing [IEEE 829]: Testing conducted to establish whether a new software product or software-based system (or components of it) satisfies its criteria.

See also

Glossary:

- Acceptance Testing
- Qualification Testing
- Operational Testing
- Testing

Standards:

- IEEE 829
- ISO/IEC/IEEE 24765

6.98. Direct Measure

A measure of an attribute that does not depend upon a measure of any other attribute. [ISO/IEC 14598, ISO/IEC 9126-1]

See also

Glossary:

- Base Measure
- Derived Measure
- Indirect Measure
- Measure

Papers:

- Software Engineering Metrics: What Do They Measure And How Do We Know [https://maisqual.squoring.com/wiki/index.php/Software_Engineering_Metrics:_What_Do_They_Measure_And_How_Do_We_Know]

Standards:

- ISO/IEC 9126-1

→ ISO/IEC 14598

6.99. Direct Metric

A metric that does not depend upon a measure of any other attribute. [IEEE 1061]

See also

Glossary:

- Direct Measure
- Indirect Metric
- Metric

Standards:

- IEEE 1061

6.100. Document

Uniquely identified unit of information for human use, such as a report, specification, manual or book, in printed or electronic form. [ISO/IEC 9294]

Other Definitions

Document (verb) [ISO/IEC/IEEE 24765]: To add comments to a computer program.

Document [ISO/IEC 15910]: An item of documentation.

Document [ISO/IEC 20000]: Information and its supporting medium.

Document [ISO/IEC 26514]: Separately identified piece of documentation which could be part of a documentation set.

Notes

- Example: in software engineering: project plans, specifications, test plans, user manuals. [ISO/IEC/IEEE 24765]
- Documents include both paper and electronic documents. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Documentation
- Installation Manual
- Maintenance Manual
- Operator Manual
- Support Manual
- User Manual

Standards:

- ISO/IEC 9294
- ISO/IEC 15910
- ISO/IEC 20000
- ISO/IEC 26514
- ISO/IEC/IEEE 24765

6.101. Documentation

Collection of related documents that are designed, written, produced and maintained. [ISO/IEC 9294]

Other Definitions

Documentation [ISO/IEC 26514]: Information that explains how to use a software product.

Documentation [IEEE 829]:

1. A collection of documents on a given subject.
2. Any written or pictorial information describing, defining, specifying, reporting, or certifying activities, requirements, procedures, or results.
3. The process of generating or revising a document.
4. The management of documents, including identification, acquisition, processing, storage, and dissemination.

Examples

- Printed manuals, on-screen information, and stand-alone online help. [ISO/IEC 26514]

Notes

- Documentation can be provided as separate documentation or as embedded documentation or both. [ISO/IEC 26514]

See also

Glossary:

- Document
- Installation Manual
- Maintenance Manual
- Operator Manual
- Programmer Manual
- Support Manual
- Test Documentation
- User Documentation
- User Manual

Standards:

- IEEE 829
- ISO/IEC 9294
- ISO/IEC 26514

6.102. Dynamic Analysis

The process of evaluating a system or component based on its behavior during execution. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Testing

Standards:

- ISO/IEC/IEEE 24765

6.103. Earned Value

The value of work performed expressed in terms of the approved budget assigned to that work for a schedule activity or work breakdown structure component. [IEEE 1490]

See also

Standards:

- IEEE 1490

6.104. Effectiveness

The capability of the software product to enable users to achieve specified goals with accuracy and completeness in a specified context of use. [ISO/IEC 9126-1]

Other Definitions

Effectiveness [ISO/IEC 25062]: The accuracy and completeness with which users achieve specified goals.

See also

- ISO/IEC 9126-1

6.105. Efficiency

Resources expended in relation to the accuracy and completeness with which users achieve goals. [ISO/IEC 25062]

Other Definitions

Efficiency [IEEE 610.12, ISO/IEC/IEEE 24765]: The degree to which a system or component performs its designated functions with minimum consumption of resources.

Efficiency [ISO/IEC 9126-1]: The capability of the software product to provide appropriate performance, relative to the amount of resources used, under stated conditions.

Notes

- Resources may include other software products, the software and hardware configuration of the system, and materials (e.g. print paper, diskettes). [ISO/IEC 9126-1]
- For a system which is operated by a user, the combination of functionality, reliability, usability and efficiency can be measured externally by quality in use. [ISO/IEC 9126-1]

See also

Glossary:

- Efficiency Compliance

Standards:

- IEEE 610.12
- ISO/IEC 9126-1
- ISO/IEC/IEEE 24765
- ISO/IEC 25062

6.106. Efficiency Compliance

The capability of the software product to adhere to standards or conventions relating to efficiency. [ISO/IEC 9126-1]

See also

Glossary:

- Efficiency

Standards:

- ISO/IEC 9126-1

6.107. Effort

The number of labor units required to complete a schedule activity or work breakdown structure component. Usually expressed as staff hours, staff days, or staff weeks. [IEEE 1490]

See also

Standards:

- IEEE 1490

6.108. Encapsulation

A software development technique that consists of isolating a system function or a set of data and operations on those data within a module and providing precise specifications for the module. [ISO/IEC/IEEE 24765]

Other Definitions

Encapsulation [IEEE 1320]: The concept that access to the names, meanings, and values of the responsibilities of a class is entirely separated from access to their realization.

Encapsulation [ISO/IEC/IEEE 24765]: The idea that a module has an outside that is distinct from its inside, that it has an external interface and an internal implementation.

See also

Standards:

- IEEE 1320
- ISO/IEC/IEEE 24765

6.109. End User

Individual person who ultimately benefits from the outcomes of the system. [ISO/IEC 25000]

Other Definitions

End User [IEEE 1233]: The person or persons who will ultimately be using the system for its intended purpose. [IEEE 1233]

End User [ISO 9127]: The person who uses the software package.

End User [ISO/IEC 29881]: Any person that communicates or interacts with the software at any time.

See also

Standards:

- IEEE 1233
- ISO 9127
- ISO/IEC 25000
- ISO/IEC 29881

6.110. Entity

Object³ that is to be characterised by measuring its attributes. [ISO/IEC 15939, ISO/IEC 25000]

Other Definitions

Entity [IEEE 1320]: The representation of a set of real or abstract things that are recognized as the same type because they share the same characteristics and can participate in the same relationships.

Entity [ISO/IEC 15474]: An object (i.e., thing, event or concept) that occurs in a model (i.e., transfer).

Entity [ISO/IEC/IEEE 24765]: In computer programming, any item that can be named or denoted in a program.

Entity [ISO/IEC 29881]: Logical component of the data store, representing fundamental things of relevance to the user, and about which persistent information is stored.

³An object can be a process, product, project, or resource.

Examples

- A data item, program statement, or subprogram. [ISO/IEC/IEEE 24765]

See also

Standards:

- IEEE 1320
- ISO/IEC 15474
- ISO/IEC 15939
- ISO/IEC/IEEE 24765
- ISO/IEC 25000
- ISO/IEC 29881

6.111. Entry Point

A point in a software module at which execution of the module can begin. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Exit

Standards:

- ISO/IEC/IEEE 24765

6.112. Environment

The configuration(s) of hardware and software in which the software operates. [ISO 9127]

Other Definitions

Environment [IEEE 1362]: The circumstances, objects, and conditions that surround a system to be built.

Environment [IEEE 1233]: The circumstances, objects, and conditions that will influence the completed system.

Environment [IEEE 1320]: A concept space, i.e., an area in which a concept has an agreed-to meaning and one or more agreed-to names that are used for the concept.

See also

Standards:

- IEEE 1233
- IEEE 1320
- IEEE 1362
- ISO 9127

6.113. Error

A human action that produces an incorrect result, such as software containing a fault. [ISO/IEC/IEEE 24765]

Other Definitions

Error [ISO/IEC/IEEE 24765]:

1. An incorrect step, process, or data definition.
2. An incorrect result
3. The difference between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition.

Notes

→ Example: omission or misinterpretation of user requirements in a software specification, incorrect translation, or

omission of a requirement in the design specification. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Defect
- Failure
- Fault

Standards:

- ISO/IEC/IEEE 24765

6.114. Error Tolerance

The ability of a system or component to continue normal operation despite the presence of erroneous inputs. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Fault Tolerance
- Robustness

Standards:

- ISO/IEC/IEEE 24765

6.115. Evaluation

Systematic determination of the extent to which an entity meets its specified criteria. [ISO/IEC 12207]

Other Definitions

Evaluation [ISO/IEC 15414]: An action that assesses the value of something.

Notes

- Example: The action by which an ODP system assigns a relative status to some thing according to estimation by the system. Value can be considered in terms of usefulness, importance, preference, acceptability, etc.; the evaluated target may be a credit rating, a system state, a potential behavior. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Evaluation Activity
- Evaluation Group
- Evaluation Method
- Evaluation Module
- Evaluation Technology

Standards:

- ISO/IEC 12207
- ISO/IEC 15414
- ISO/IEC/IEEE 24765

6.116. Evaluation Activity

Assessment of a software product against identified and applicable quality characteristics performed using applicable techniques or methods. [ISO/IEC 25001]

See also

Glossary:

- Evaluation
- Evaluation Group
- Evaluation Method
- Evaluation Module
- Evaluation Technology

Standards:

- ISO/IEC 25001

6.117. Evaluation Group

Organization responsible for specifying the software quality requirements as well as managing and implementing the software quality evaluation activities through the provision of technology, tools, experiences, and management skills. [ISO/IEC 25001]

See also

Glossary:

- Evaluation
- Evaluation Activity
- Evaluation Method
- Evaluation Module
- Evaluation Technology

Standards:

- ISO/IEC 25001

6.118. Evaluation Method

Procedure describing actions to be performed by the evaluator in order to obtain results for the specified measurement applied to the specified product components or on the product as a whole. [ISO/IEC 25000]

See also

Glossary:

- Evaluation
- Evaluation Activity
- Evaluation Group
- Evaluation Module
- Evaluation Technology

Standards:

- ISO/IEC 25000

6.119. Evaluation Module

A package of evaluation technology for a specific software quality characteristic or sub-characteristic. [ISO/IEC 9126-1, ISO/IEC 14598]

Notes

- The package includes evaluation methods and techniques, inputs to be evaluated, data to be measured and collected and supporting procedures and tools. [ISO/IEC 9126-1]

See also

Glossary:

- Evaluation
- Evaluation Activity
- Evaluation Group
- Evaluation Method
- Evaluation Technology

Standards:

- ISO/IEC 9126-1
- ISO/IEC 14598

6.120. Evaluation Technology

Technique, processes, tools, measures and relevant technical information used for evaluation. [ISO/IEC 25001]

See also

Glossary:

- Evaluation
- Evaluation Activity
- Evaluation Group
- Evaluation Method
- Evaluation Module

Standards:

- ISO/IEC 25001

6.121. Evaluation Tool

An instrument that can be used during evaluation to collect data, to perform interpretation of data or to automate part of the evaluation. [ISO/IEC 14598-5]

See also

Glossary:

- Evaluation

Standards:

- ISO/IEC 14598-5

6.122. Execute

To carry out an instruction, process, or computer program. [IEEE 1490]

Other Definitions

Execute [IEEE 1490]: Directing, managing, performing, and accomplishing the project work, providing the deliverables, and providing work performance information.

See also

Standards:

- IEEE 1490

6.123. Execution Efficiency

The degree to which a system or component performs its designated functions with minimum consumption of time. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Efficiency
- Execution Time

Standards:

- ISO/IEC/IEEE 24765

6.124. Execution Time

The time which elapses between task submission and completion. [ISO/IEC 14756]

Notes

- Processor time is usually less than elapsed time because the processor may be idle (for example, awaiting needed computer resources) or employed on other tasks during the execution of a program. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Execution Efficiency

Standards:

- ISO/IEC 14756
- ISO/IEC/IEEE 24765

6.125. Exit

A point in a software module at which execution of the module can terminate. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Entry Point

Standards:

- ISO/IEC/IEEE 24765

6.126. Expandability

The degree of effort required to improve or modify software functions' efficiency. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.127. Extendability

The ease with which a system or component can be modified to increase its storage or functional capacity.
[ISO/IEC/IEEE 24765]

See also

Glossary:

→ Expandability

→ Flexibility

→ Maintainability

Standards:

→ ISO/IEC/IEEE 24765

6.128. External Attribute

A measurable property of an entity which can only be derived with respect to how it relates to its environment.
[ISO/IEC 14598-3]

Notes

→ External attributes are those that relate to requirements (external properties of the software). External attributes can only be derived from the operational behavior of the system of which it is a part. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Internal Attribute

Standards:

→ ISO/IEC 14598

6.129. External Measure

An indirect measure of a product derived from measures of the behaviour of the system of which it is a part.
[ISO/IEC 9126-1, ISO/IEC 14598]

Notes

→ The system includes any associated hardware, software (either custom software or off-the-shelf software) and users. [ISO/IEC 9126-1]

- The number of failures found during testing is an external measure of the number of faults in the program because the number of failures are counted during the operation of a computer system running the program. [ISO/IEC 9126-1]
- External measures can be used to evaluate quality attributes closer to the ultimate objectives of the design. [ISO/IEC 9126-1]

See also

Glossary:

- Base Measure
- Derived Measure
- Direct Measure
- Indirect Measure
- Internal Measure
- Measure

Standards:

- ISO/IEC 14598
- ISO/IEC 9126-1

6.130. External Quality

The extent to which a product satisfies stated and implied needs when used under specified conditions. [ISO/IEC 9126-1, ISO/IEC 14598]

Notes

- External Quality is the totality of characteristics of the software product from an external view. It is the quality when the software is executed, which is typically measured and evaluated while testing in a simulated environment with simulated data using external metrics. [ISO/IEC 9126-1]

See also

Glossary:

- External Software Quality
- Internal Quality
- Quality
- Quality in Use

Standards:

- ISO/IEC 9126-1
- ISO/IEC 9126-2
- ISO/IEC 14598

6.131. External Software Quality

Capability of a software product to enable the behavior of a system to satisfy stated and implied needs when the system is used under specified conditions. [ISO/IEC 25000]

Notes

- The number of failures found during testing is an external software quality measure related to the number of faults present in the program. The two measures are not necessarily identical since testing may not find all faults, and a fault may give rise to apparently different failures in different circumstances. [ISO/IEC/IEEE 24765]
- Attributes of the behavior can be verified and/or validated by executing the software product during testing and operation. [ISO/IEC/IEEE 24765]

See also

Glossary:

- External Quality
- Internal Software Quality

Standards:

- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.132. Facility

Physical means or equipment for facilitating the performance of an action. [ISO/IEC 12207, ISO/IEC 15288]

Notes

- Buildings, instruments, tools. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC/IEEE 24765

6.133. Failure

The termination of the ability of a product to perform a required function or its inability to perform within previously specified limits. [ISO/IEC 9126-1, ISO/IEC 14598-5, ISO/IEC 25000]

Other Definitions

Failure [SIGIST]: Deviation of the software from its expected delivery or service.

Failure [IEEE 610.12]: The inability of a system or component to perform its required functions within specified performance requirements.

Failure [ISO/IEC/IEEE 24765]: An event in which a system or system component does not perform a required function within specified limits.

Notes

- According to Laprie et al.⁴, "a system failure occurs when the delivered service no longer complies with the specifications, the latter being an agreed description of the system's expected function and/or service". This definition applies to both hardware and software system failures. Faults or bugs in a hardware or a software component cause errors. An error is defined by Laprie et al. as that part of the system which is liable to lead to subsequent failure, and an error affecting the service is an indication that a failure occurs or has occurred. If the system comprises of multiple components, errors can lead to a component failure. As various components in the system interact, failure of one component might introduce one or more faults in another. [University of Duke]
- The fault tolerance discipline distinguishes between a human action (a mistake), its manifestation (a hardware or software fault), the result of the fault (a failure), and the amount by which the result is incorrect (the error). [IEEE 610.12]

See also

Glossary:

- Defect
- Fault
- Fault Tolerance

Standards:

- IEEE 610.12
- ISO/IEC 9126-1
- ISO/IEC 14598-5
- ISO/IEC 25000
- ISO/IEC/IEEE 24765
- SIGIST

6.134. Failure Rate

The ratio of the number of failures of a given category to a given unit of measure. [ISO/IEC/IEEE 24765]

Notes

- Failures per unit of time, failures per number of transactions, failures per number of computer runs. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Defect Density
- Failure

Standards:

⁴J. C. Laprie (Ed.). Dependability: Basic Concepts and Terminology. Springer-Verlag, Wein, New York, 1992.

→ IEEE 829

6.135. Fault

An incorrect step, process or data definition in a computer program. [IEEE 610.12, ISO/IEC 9126-1]

Other Definitions

Fault [RTCA/EUROCAE]: A manifestation of an error in software. A fault, if encountered may cause a failure.

Fault [ISO/IEC/IEEE 24765]:

1. a manifestation of an error in software.
2. an incorrect step, process, or data definition in a computer program.
3. a defect in a hardware device or component.

See also

Glossary:

→ Defect

→ Failure

Standards:

→ IEEE 610.12

→ ISO/IEC 9126-1

→ ISO/IEC/IEEE 24765

→ RTCA/EUROCAE

6.136. Fault Tolerance

The capability of the software product to maintain a specified level of performance in cases of software faults or of infringement of its specified interface. [ISO/IEC 9126-1]

Other Definitions

Fault Tolerance [IEEE 610.12]:

1. The ability of a system or component to continue normal operation despite the presence of hardware or software faults.
2. The number of faults a system or component can withstand before normal operation is impaired.
3. Pertaining to the study of errors, faults, and failures, and of methods for enabling systems to continue normal operation in the presence of faults.

Notes

→ The specified level of performance may include fail safe capability. [ISO/IEC 9126-1]

See also

Glossary:

- Fault
- Reliability

Standards:

- ISO/IEC 9126-1
- IEEE 610.12

6.137. Feasibility

The degree to which the requirements, design, or plans for a system or component can be implemented under existing constraints. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.138. Feature

Distinguishing characteristic of a system item. [IEEE 829]

Notes

- Includes both functional and nonfunctional attributes such as performance and reusability. [ISO/IEC/IEEE 24765]

See also

Standards:

- IEEE 829
- ISO/IEC/IEEE 24765

6.139. Feature Freeze

A period during which no new features are added to a specific branch. [IEEE 829]

Notes

- Allows the branch to stabilize for a release. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Code Freeze
- Feature

Standards:

- IEEE 829

→ ISO/IEC/IEEE 24765

6.140. Finite State Machine

A computational model consisting of a finite number of states and transitions between those states, possibly with accompanying actions. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.141. Flexibility

The ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed. [ISO/IEC/IEEE 24765, IEEE 610.12]

See also

Glossary:

→ Adaptability

→ Changeability

Standards:

→ ISO/IEC/IEEE 24765

→ IEEE 610.12

6.142. Frozen Branch

A branch where no development takes place, either in preparation for a release or because active development has ceased on it. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.143. Function

A software module that performs a specific action, is invoked by the appearance of its name in an expression, may receive input values, and returns a single value. [ISO/IEC/IEEE 24765]

Other Definitions

Function [IEEE 1233]: A task, action, or activity that must be accomplished to achieve a desired outcome.

Function [IEEE 1320]: A transformation of inputs to outputs, by means of some mechanisms, and subject to certain controls, that is identified by a function name and modeled by a box.

Function [ISO/IEC 26514]: Part of an application that provides facilities for users to carry out their tasks.

Function [ISO/IEC/IEEE 24765]: A defined objective or characteristic action of a system or component.

See also

Glossary:

→ Routine

Standards:

→ IEEE 1233

→ IEEE 1320

→ ISO/IEC 26514

→ ISO/IEC/IEEE 24765

6.144. Functional Analysis

A systematic investigation of the functions of a real or planned system. [ISO/IEC 2382-1]

Other Definitions

Functional Analysis [ISO/IEC/IEEE 24765]: Examination of a defined function to identify all the sub-functions necessary to accomplish that function, to identify functional relationships and interfaces (internal and external) and capture these in a functional architecture, to flow down upper-level performance requirements and to assign these requirements to lower-level sub-functions.

See also

Standards:

→ ISO/IEC 2382-1

→ ISO/IEC/IEEE 24765

6.145. Functional Requirement

A statement that identifies what a product or process must accomplish to produce required behavior and/or results. [IEEE 1220]

Other Definitions

Model [ISO/IEC/IEEE 24765]: A requirement that specifies a function that a system or system component must be able to perform.

See also

Glossary:

→ Nonfunctional Requirement

→ Requirement

Standards:

- IEEE 1220
- ISO/IEC/IEEE 24765

6.146. Functional Size

A size of the software derived by quantifying the functional user requirements. [ISO/IEC 14143-1]

See also

Standards:

- ISO/IEC 14143

6.147. Functional Testing

Testing that ignores the internal mechanism of a system or component and focuses solely on the outputs generated in response to selected inputs and execution conditions. [ISO/IEC/IEEE 24765]

Other Definitions

- Testing conducted to evaluate the compliance of a system or component with specified functional requirements. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Performance Testing
- Structural Testing
- Testing

Standards:

- ISO/IEC/IEEE 24765

6.148. Functional Unit

An entity of hardware or software, or both, capable of accomplishing a specified purpose. [ISO/IEC 2382-1]

See also

Standards:

- ISO/IEC 2382-1

6.149. Functionality

The capability of the software product to provide functions which meet stated and implied needs when the software is used under specified conditions. [ISO/IEC 9126-1]

Other Definitions

Functionality [IEEE 1362]: The capabilities of the various computational, user interface, input, output, data management, and other features provided by a product.

Notes

- Functionality is one of the six characteristics of quality in the ISO/IEC 9126 quality model.
- This characteristic is concerned with what the software does to fulfil needs, whereas the other characteristics are mainly concerned with when and how it fulfils needs. [ISO/IEC 9126-1]
- For a system which is operated by a user, the combination of functionality, reliability, usability and efficiency can be measured externally by quality in use. [ISO/IEC 9126-1]

See also

Standards:

- IEEE 1362
- ISO/IEC 9126-1

6.150. Functionality Compliance

The capability of the software product to adhere to standards, conventions or regulations in laws and similar prescriptions relating to functionality. [ISO/IEC 9126-1]

See also

Glossary:

- Functionality

Standards:

- ISO/IEC 9126-1

6.151. Generality

The degree to which a system or component performs a broad range of functions. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.152. Generic Practice

An activity that, when consistently performed, contributes to the achievement of a specific process attribute. [ISO/IEC 15504]

See also

Standards:

→ ISO/IEC 15504

6.153. Glossary

The collection of the names and narrative descriptions of all terms that may be used for defined concepts within an environment. [IEEE 1320]

See also

Standards:

→ IEEE 1320

6.154. Goal

Intended outcome of user interaction with a product. [ISO/IEC 25062]

Other Definitions

Goal [ISO/IEC 9126-4]: An intended outcome.

See also

Standards:

→ ISO/IEC 9126

→ ISO/IEC 25062

6.155. Granularity

The depth or level of detail at which data is collected. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.156. Historical Information

Documents and data on prior projects including project files, records, correspondence, closed contracts, and closed projects. [IEEE 1490]

See also

Standards:

→ IEEE 1490

6.157. Hybrid Coupling

A type of coupling in which different subsets of the range of values that a data item can assume are used for different and unrelated purposes in different software modules. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Coupling

Standards:

→ ISO/IEC/IEEE 24765

6.158. Impact Analysis

Identification of all system and software products that a change request affects and development of an estimate of the resources needed to accomplish the change. [ISO/IEC/IEEE 24765]

Notes

→ This includes determining the scope of the changes to plan and implement work, accurately estimating the resources needed to perform the work, and analyzing the requested changes' cost and benefits. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.159. Implementation

The process of translating a design into hardware components, software components, or both. [ISO/IEC/IEEE 24765]

Other Definitions

Implementation ISO/IEC/IEEE 24765]: The installation and customization of packaged software.

Implementation ISO/IEC/IEEE 24765]: Construction.

Implementation ISO/IEC 2382]: The system development phase at the end of which the hardware, software and procedures of the system considered become operational.

Implementation ISO/IEC 26514]: Phase of development during which user documentation is created according to the design, tested, and revised.

See also

Glossary:

→ Coding

Standards:

→ ISO/IEC/IEEE 24765

6.160. Implied Needs

Needs that may not have been stated but are actual needs. [ISO/IEC 25000]

Other Definitions

Implied Needs [ISO/IEC 9126-1, ISO/IEC 14598-1]: Needs that may not have been stated but are actual needs when the entity is used in particular conditions.

Notes

- Implied needs are real needs which may not have been documented. [ISO/IEC 9126-1]

See also

Glossary:

- Requirement

Standards:

- ISO/IEC 9126-1
- ISO/IEC 14598-1
- ISO/IEC 25000

6.161. Incremental Development

A software development technique in which requirements definition, design, implementation, and testing occur in an overlapping, iterative (rather than sequential) manner, resulting in incremental completion of the overall software product. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.162. Indicator

Measure that provides an estimate or evaluation of specified attributes derived from a model with respect to defined information needs. [ISO/IEC 15939, ISO/IEC 25000]

Other Definitions

Indicator [ISO/IEC 9126-1, ISO/IEC 14598-1]: A measure that can be used to estimate or predict another measure.

Notes

- Example: A flag or semaphore. [ISO/IEC/IEEE 24765]
- The predicted measure may be of the same or a different software quality characteristic. ISO/IEC 9126-1
- Indicators may be used both to estimate software quality attributes and to estimate attributes of the development process. They are imprecise indirect measures of the attributes. ISO/IEC 9126-1

See also

Glossary:

→ Indicator Value

Standards:

→ ISO/IEC 9126-1

→ ISO/IEC 15939

6.163. Indicator Value

Numerical or categorical result assigned to an indicator. [ISO/IEC 15939]

See also

Glossary:

→ Indicator

Standards:

→ ISO/IEC 15939

6.164. Indirect Measure

A measure of an attribute that is derived from measures of one or more other attributes. [ISO/IEC 14598]

Notes

→ An external measure of an attribute of a computing system (such as the response time to user input) is an indirect measure of attributes of the software as the measure will be influenced by attributes of the computing environment as well as attributes of the software. [ISO/IEC 9126-1]

See also

Glossary:

→ Base Measure

→ Derived Measure

→ Direct Measure

→ External Measure

→ Internal Measure

Papers:

→ Software Engineering Metrics: What Do They Measure And How Do We Know [https://maisqual.squoring.com/wiki/index.php/Software_Engineering_Metrics:_What_Do_They_Measure_And_How_Do_We_Know]

Standards:

→ ISO/IEC 14598

→ ISO/IEC 9126-1

6.165. Indirect Metric

An Indirect Metric is a metric that is derived from one or more other metrics. [IEEE 1061]

See also

Glossary:

→ Direct Metric

→ Metric

Standards:

→ IEEE 1061

6.166. Information

An information processing, knowledge concerning objects, such as facts, events, things, processes, or ideas, including concepts, that within a certain context has a particular meaning. [ISO/IEC 2382-1]

Notes

→ Although information will necessarily have a representation form to make it communicable, it is the interpretation of this representation (the meaning) that is relevant in the first place. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Information Analysis

Standards:

→ ISO/IEC 2382-1

→ ISO/IEC/IEEE 24765

6.167. Information Analysis

A systematic investigation of information and its flow in a real or planned system. [ISO/IEC 2382-1]

See also

Glossary:

→ Information

Standards:

→ ISO/IEC 2382-1

6.168. Information Management

In an information processing system, the functions of controlling the acquisition, analysis, retention, retrieval, and distribution of information. [ISO/IEC 2382-1]

See also

Glossary:

→ Information

Standards:

→ ISO/IEC 2382-1

6.169. Information Need

Insight necessary to manage objectives, goals, risks, and problems. [ISO/IEC 15939]

See also

→ ISO/IEC 15939

6.170. Information Product

One or more indicators and their associated interpretations that address an information need. [ISO/IEC 15939, ISO/IEC 25000]

Example

→ A comparison of a measured defect rate to planned defect rate along with an assessment of whether or not the difference indicates a problem. [ISO/IEC 15939]

See also

Glossary:

→ Indicator

→ Information

Standards:

→ ISO/IEC 15939

→ ISO/IEC 25000

6.171. Inspection

A static analysis technique that relies on visual examination of development products to detect errors, violations of development standards, and other problems. [ISO/IEC/IEEE 24765]

Other Definitions

Inspection [IEEE 1490]: Examining or measuring to verify whether an activity, component, product, result, or service conforms to specified requirements.

Notes

- Inspections are peer examinations led by impartial facilitators who are trained in inspection techniques. Determination of remedial or investigative action for an anomaly is a mandatory element of a software inspection, although the solution should not be determined in the inspection meeting. Types include code inspection; design inspection. [ISO/IEC/IEEE 24765]

See also

Standards:

- IEEE 1490
- ISO/IEC/IEEE 24765

6.172. Installability

The capability of the software product to be installed in a specified environment. [ISO/IEC 9126-1]

Notes

- If the software is to be installed by an end user, installability can affect the resulting suitability and operability. [ISO/IEC 9126-1]

See also

- ISO/IEC 9126-1

6.173. Installation Manual

A document that provides the information necessary to install a system or component, set initial parameters, and prepare the system or component for operational use. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Document
- Maintenance Manual
- Operator Manual
- Support Manual
- User Manual

Standards:

- ISO/IEC/IEEE 24765

6.174. Integration

The process of combining software components, hardware components, or both into an overall system. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.175. Integration Test

The progressive linking and testing of programs or modules in order to ensure their proper functioning in the complete system. [ISO/IEC 2382]

See also

Glossary:

→ Integration

→ Testing

Standards:

→ ISO/IEC 2382

6.176. Integrity

The degree to which a system or component prevents unauthorized access to, or modification of, computer programs or data. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.177. Interface Testing

Testing conducted to evaluate whether systems or components pass data and control correctly to one another. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Integration Test

→ System Testing

→ Testing

→ Unit Test

Standards:

→ ISO/IEC/IEEE 24765

6.178. Intermediate Software Product

A product of the software development process that is used as input to another stage of the software development process. [ISO/IEC 14598, ISO/IEC 9126-1, ISO/IEC 25000]

Notes

- Example: static and dynamic models, other documents and source code. [ISO/IEC/IEEE 24765]
- In some cases an intermediate product may also be an end product. [ISO/IEC 9126-1]

See also

Glossary:

- Software Product

Standards:

- ISO/IEC 9126-1
- ISO/IEC 14598
- ISO/IEC/IEEE 24765
- ISO/IEC 25000

6.179. Internal Attribute

A measurable property of an entity which can be derived purely in terms of the entity itself. [ISO/IEC 14598, ISO/IEC/IEEE 24765]

Notes

- Internal attributes are those that relate to the internal organization of the software and its development. [ISO/IEC/IEEE 24765]

See also

Glossary:

- External Attribute

Standards:

- ISO/IEC 14598
- ISO/IEC/IEEE 24765

6.180. Internal Measure

A measure of the product itself, either direct or indirect. [ISO/IEC 14598, ISO/IEC 9126-1, ISO/IEC/IEEE 24765]

Notes

- An external measure of an attribute of a computing system (such as the response time to user input) is an indirect measure of attributes of the software as the measure will be influenced by attributes of the computing environment as well as attributes of the software. [ISO/IEC 9126-1]
- The number of lines of code, complexity measures, the number of faults found in a walk through and the Fog Index are all internal measures made on the product itself. [ISO/IEC 9126-1]

See also

Glossary:

- Direct Measure
- External Measure
- Indirect Measure
- Measure

Standards:

- ISO/IEC 14598
- ISO/IEC 9126-1
- ISO/IEC/IEEE 24765

6.181. Internal Quality

The totality of attributes of a product that determine its ability to satisfy stated and implied needs when used under specified conditions. [ISO/IEC 9126-1, ISO/IEC/IEEE 24765]

Notes

- Internal quality is the totality of characteristics of the software product from an internal view. Internal quality is measured and evaluated against the internal quality requirements. Details of software product quality can be improved during code implementation, reviewing and testing, but the fundamental nature of the software product quality represented by internal quality remains unchanged unless redesigned. [ISO/IEC 9126-1]
- The term "internal quality", used in ISO/IEC 14598 to contrast with "external quality", has essentially the same meaning as "quality" in ISO 8402. [ISO/IEC 9126-1]

See also

Glossary:

- External Quality
- Quality in Use

Standards:

- ISO/IEC 9126
- ISO/IEC 9126-1
- ISO/IEC 9126-3
- ISO/IEC 14598
- ISO/IEC/IEEE 24765

6.182. Internal Software Quality

Capability of a set of static attributes of a software product to satisfy stated and implied needs when the software product is used under specified conditions. [ISO/IEC 25000]

Examples

- The number of lines of code, complexity measures and the number of faults found in a walkthrough are all internal software quality measures made on the product itself. [ISO/IEC/IEEE 24765]

Notes

- Static attributes include those that relate to the software architecture, structure and its components. Static attributes can be verified by review, inspection and/or automated tools. [ISO/IEC/IEEE 24765]

See also

Glossary:

- External Software Quality

Standards:

- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.183. Interoperability

The capability of the software product to interact with one or more specified systems. [ISO/IEC 9126-1]

Notes

- Interoperability is used in place of compatibility in order to avoid possible ambiguity with replaceability. [ISO/IEC 9126-1]

See also

- ISO/IEC 9126-1

6.184. Interoperability Testing

Testing conducted to ensure that a modified system retains the capability of exchanging information with systems of different types, and of using that information. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Testing

Standards:

- ISO/IEC/IEEE 24765

6.185. Interval Scale

Scale in which the measurement values have equal distances corresponding to equal quantities of the attribute. [ISO/IEC/IEEE 24765]

Notes

- Example: Cyclomatic complexity has the minimum value of one, but each increment represents an additional path. The value of zero is not possible. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Scale

Standards:

- ISO/IEC/IEEE 24765

6.186. Item

An entity such as a part, component, subsystem, equipment or system that can be individually considered. An item may consist of hardware, software or both. [ISO/IEC 15026]

See also

Standards:

- ISO/IEC 15026

6.187. Iteration

1. The process of performing a sequence of steps repeatedly.
2. A single execution of the sequence of steps. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.188. Key Practices

The infrastructures and activities that contribute most to the effective implementation and institutionalization of a key process area. [CMMi]

Notes

In the CMMi process, each key process area is described in terms of the key practices that contribute to satisfying its goals. The key practices describe the infrastructure and activities that contribute most to the effective implementation and institutionalization of the key process area.

Each key practice consists of a single sentence, often followed by a more detailed description, which may include examples and elaboration. These

key practices, also referred to as the top-level key practices, state the fundamental policies, procedures, and activities for the key process area.

The components of the detailed description are frequently referred to as sub-practices.

The key practices describe "what" is to be done, but they should not be interpreted as mandating "how" the goals should be achieved. Alternative practices may accomplish the goals of the key process area. The key practices should be interpreted rationally to judge whether the goals of the key

process area are effectively, although perhaps differently, achieved.

See also

- CMMi
- Key Process Area

6.189. Key Process Area

A cluster of related activities that, when performed collectively, achieve a set of goals considered important for establishing process capability. [CMMi]

Notes

The key process areas have been defined to reside at a single maturity level. They are the areas identified by the SEI [<http://www.sei.cmu.edu>] to be the principal building blocks to help determine the software process capability of an organization and understand the improvements needed to advance to higher maturity levels.

- The Level 2 key process areas in the CMMi are Requirements Management, Software Project Planning, Software Project Tracking and Oversight, Software Subcontract Management, Software Quality Assurance, and Software Configuration Management.
- The Level 3 key process areas in the CMMi are Organization Process Focus, Organization Process Definition, Training Program, Integrated Software Management, Software Product Engineering, Intergroup Coordination, and Peer Reviews.
- The Level 4 key process areas are Quantitative Process Management and Software Quality Management.
- The Level 5 key process areas are Defect Prevention, Technology Change Management, and Process Change Management.

See also

- CMMi
- Key Practices

6.190. Knowledge Base

A database that contains inference rules and information about human experience and expertise in a domain. [ISO/IEC 2382-1]

See also

- Standards:
- ISO/IEC 2382-1

6.191. Learnability

The capability of the software product to enable the user to learn its application. [ISO/IEC 9126-1]

Notes

→ The internal attributes correspond to suitability for learning as defined in ISO 9241-10. [ISO/IEC 9126-1]

See also

Standards:

→ ISO/IEC 9126-1

6.192. Lessons Learned

The learning gained from the process of performing the project. Lessons learned may be identified at any point. Also considered a project record, to be included in the lessons learned knowledge base. [IEEE 1490]

See also

Glossary:

→ Knowledge Base

Standards:

→ IEEE 1490

6.193. Level of Performance

The degree to which the needs are satisfied, represented by a specific set of values for the quality characteristics. [ISO/IEC 9126-1]

See also

Glossary:

→ Performance

→ Performance Indicator

Standards:

→ ISO/IEC 9126-1

6.194. Life Cycle

Evolution of a system, product, service, project or other human-made entity from conception through retirement. [ISO/IEC 12207, ISO/IEC 15288]

Other Definitions

Life Cycle [IEEE 1220]: The system or product evolution initiated by a perceived stakeholder need through the disposal of the products.

See also

Glossary:

→ Life Cycle Model

Standards:

→ IEEE 1220

→ ISO/IEC 12207

→ ISO/IEC 15288

6.195. Life Cycle Model

Framework of processes and activities concerned with the life cycle that may be organized into stages, which also acts as a common reference for communication and understanding. [ISO/IEC 12207, ISO/IEC 15288]

See also

Glossary:

→ Life Cycle

Standards:

→ ISO/IEC 12207

→ ISO/IEC 15288

6.196. Maintainability

The capability of the software product to be modified. [ISO/IEC 9126-1, ISO/IEC 14764]

Other Definitions

Maintainability [ISO/IEC/IEEE 24765]: The ease with which a software system or component can be modified to change or add capabilities, correct faults or defects, improve performance or other attributes, or adapt to a changed environment.

Maintainability [ISO/IEC/IEEE 24765]: The average effort required to locate and fix a software failure.

Maintainability [IEEE 982]: Speed and ease with which a program can be corrected or changed.

Notes

→ Modifications may include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications. [ISO/IEC 9126-1

See also

Glossary:

→ Extendability

→ Flexibility

- Maintainer
- Maintenance

Standards:

- IEEE 982
- ISO/IEC 9126-1
- ISO/IEC 14764
- ISO/IEC/IEEE 24765

6.197. Maintainability Compliance

The capability of the software product to adhere to standards or conventions relating to maintainability. [ISO/IEC 9126-1]

See also

Glossary:

- Maintainability

Standards:

- ISO/IEC 9126-1

6.198. Maintainer

Individual or organization that performs maintenance activities. [ISO/IEC 25000]

Other Definitions

Maintainer [ISO/IEC 9126-1, ISO/IEC 12207, ISO/IEC 14598]: An organisation that performs maintenance activities.

See also

Glossary:

- Maintainability
- Maintenance

Standards:

- ISO/IEC 9126-1
- ISO/IEC 12207
- ISO/IEC 14598
- ISO/IEC 25000

6.199. Maintenance

The process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment. [IEEE 610.12, ISO/IEC/IEEE 24765]

Other Definitions

Software Maintenance [ISO/IEC 14764]: The totality of activities required to provide cost-effective support to a software system.

Notes

- Maintenance activities include ⁵:
- * Perfective maintenance - Changes which improve the system in some way without changing its functionality;
- * Adaptive maintenance - Maintenance which is required because of changes in the environment of a program;
- * Corrective maintenance - The correction of previously undiscovered system errors.
- Maintainability is defined as the effort to perform maintenance tasks, the impact domain of the maintenance actions, and the error rate caused by those actions. ⁶
- Pre-delivery activities include planning for post-delivery operations, supportability, and logistics determination. Post-delivery activities include software modification, training, and operating a help desk. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Maintainability
- Maintainer
- Maintenance Manual

Standards:

- IEEE 610.12
- ISO/IEC 14764
- ISO/IEC/IEEE 24765

6.200. Maintenance Manual

A software engineering project-deliverable document that enables a system's maintenance personnel (rather than users) to maintain the system. [ISO/IEC/IEEE 24765]

Notes

- Maintenance Manual is synonym for Support Manual. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Document
- Installation Manual
- Operator Manual

⁵Ian Sommerville, "Software Engineering". Addison-Wesley, 1996.

⁶Harry M. Sneed & Agnes Kaposi. "A study on the effect of reengineering on maintainability". In Proceedings of the International Conference on Software Maintenance 1990, pages 91-99. IEEE, Computer Society Press 1990.

→ Support Manual

→ User Manual

Standards:

→ ISO/IEC/IEEE 24765

6.201. Maturity

The capability of the software product to avoid failure as a result of faults in the software. [ISO/IEC 9126-1]

See also

Standards:

→ ISO/IEC 9126-1

6.202. Measurable Concept

Abstract relationship between attributes of entities and information needs. [ISO/IEC 15939]

See also

→ ISO/IEC 15939

6.203. Measurand

Particular quantity subject to measurement. [ISO/IEC 14143-3, ISO/IEC/IEEE 24765]

Notes

→ Example of operand: vapor pressure of a given sample of water at 20 °C. [ISO/IEC/IEEE 24765]

→ The specification of a measurand may require statements about quantities such as time, temperature and pressure. [ISO/IEC 99]

See also

Standards:

→ ISO/IEC 99

→ ISO/IEC 14143-3

→ ISO/IEC/IEEE 24765

6.204. Measure

Variable to which a value is assigned as the result of measurement. [ISO/IEC 15939, ISO/IEC 25000]

Other Definition

Measure (verb) [ISO/IEC 14598, ISO/IEC 15939]: To make a measurement.

Measure [IEEE 1061]: A way to ascertain or appraise value by comparing it to a norm.

Measure (verb) [IEEE 1061]: To apply a metric.

Measure [ISO/IEC 14598]: The number or category assigned to an attribute of an entity by making a measurement.

Measure [IEEE 982]: The number or symbol assigned to an entity by a mapping from the empirical world to the formal, relational world in order to characterize an attribute.

Measure [IEEE 982]: The act or process of measuring.

Notes

- The term "measures" is used to refer collectively to base measures, derived measures, and indicators. [ISO/IEC 15939]

See also

Glossary:

- Base Measure
- Derived Measure
- Direct Measure
- Indicator
- Indirect Measure
- Measurement
- Metric

Papers:

- Software Engineering Metrics: What Do They Measure And How Do We Know [https://maisqual.squoring.com/wiki/index.php/Software_Engineering_Metrics:_What_Do_They_Measure_And_How_Do_We_Know]

Standards:

- IEEE 982
- IEEE 1061
- ISO/IEC 14598
- ISO/IEC 15939
- ISO/IEC 25000

6.205. Measurement

Set of operations having the object of determining a value of a measure. [ISO/IEC 25000]

Other Definitions

Measurement [ISO/IEC 99, ISO/IEC 15939]: Set of operations having the object of determining a value of a measure.

Measurement [IEEE 1061]: Act or process of assigning a number or category to an entity to describe an attribute of that entity.

Measurement [ISO/IEC 19759]: The assignment of numbers to objects in a systematic way to represent properties of the object.

Measurement [ISO/IEC 9126-1, ISO/IEC 14598]: The use of a metric to assign a value (which may be a number or category) from a scale to an attribute of an entity.

Measurement [ISO/IEC 19759]: the assignment of values and labels to aspects of software engineering (products, processes, and resources) and the models that are derived from them, whether these models are developed using statistical, expert knowledge or other techniques.

Notes

- Measurement can be qualitative when using categories. For example, some important attributes of software products, e.g. the language of a source program (ADA, C, COBOL, etc.) are qualitative categories. [ISO/IEC 9126-1]

See also

Glossary:

- Measure

Standards:

- IEEE 1061
- ISO/IEC 99
- ISO/IEC 9126-1
- ISO/IEC 14598
- ISO/IEC 15939
- ISO/IEC 19759
- ISO/IEC 25000

6.206. Measurement Analyst

Individual or organisation that is responsible for the planning, performance, evaluation, and improvement of measurement. [ISO/IEC 15939]

See also

- ISO/IEC 15939

6.207. Measurement Experience Base

Data store that contains the evaluation of the information products and the measurement process as well as any lessons learned during the measurement process. [ISO/IEC 15939]

See also

- ISO/IEC 15939

6.208. Measurement Function

Algorithm or calculation performed to combine two or more base measures. [ISO/IEC 15939, ISO/IEC 25000]

Notes

- A function is an algorithm or calculation performed to combine two or more base measures. The scale and unit of the derived measure depend on the scales and units of the base measures from which it is composed as well as how they are combined by the function. [ISO/IEC 15939]

See also

Glossary:

- Measure
- Measurement

Standards:

- ISO/IEC 15939
- ISO/IEC 25000

6.209. Measurement Method

Logical sequence of operations, described generically, used in quantifying an attribute with respect to a specified scale. [ISO/IEC 99, ISO/IEC 15939, ISO/IEC 25000]

Notes

- The type of measurement method depends on the nature of the operations used to quantify an attribute. Two types may be distinguished:
 - * subjective — quantification involving human judgement,
 - * objective — quantification based on numerical rules. [ISO/IEC 15939]

See also

Glossary:

- Attribute

Standards:

- ISO/IEC 99
- ISO/IEC 15939
- ISO/IEC 25000

6.210. Measurement Procedure

Set of operations, described specifically, used in the performance of a particular measurement according to a given method. [ISO/IEC 99, ISO/IEC 15939, ISO/IEC 25000]

See also

Glossary:

→ Measurement Method

Standards:

→ ISO/IEC 99

→ ISO/IEC 15939

→ ISO/IEC 25000

6.211. Measurement Process

The process for establishing, planning, performing and evaluating software measurement within an overall project or organisational measurement structure. [ISO/IEC 15939]

See also

→ ISO/IEC 15939

6.212. Measurement Process Owner

Individual or organisation responsible for the measurement process. [ISO/IEC 15939]

See also

Glossary:

→ Measurement Process

Standards:

→ ISO/IEC 15939

6.213. Measurement Sponsor

Individual or organisation that authorises and supports the establishment of the measurement process. [ISO/IEC 15939]

See also

Glossary:

→ Measurement Process

Standards:

→ ISO/IEC 15939

6.214. Measurement User

Individual or organisation that uses the information products. [ISO/IEC 15939]

See also

Glossary:

→ Information Product

Standards:

→ ISO/IEC 15939

6.215. Metric

The defined measurement method and the measurement scale. [ISO/IEC 14598, ISO/IEC 9126-1]

Notes

- Metrics can be internal or external, and direct or indirect. [ISO/IEC 9126-1]
- Metrics include methods for categorising qualitative data. [ISO/IEC 9126-1]

See also

Glossary:

- Direct Measure
- External Measure
- Indirect Measure
- Internal Measure
- Measure
- Measurement Method
- Scale

Standards:

- ISO/IEC 9126-1
- ISO/IEC 14598

6.216. Milestone

A significant point or event in the project. [IEEE 1490]

Other Definitions

Milestone [IEEE 1058]: A scheduled event used to measure progress.

Notes

- Major milestones for software projects may include an acquirer or managerial sign-off, baselining of a specification, completion of system integration, and product delivery. Minor milestones might include baselining of a software module or completion of a chapter of the user manual

See also

Glossary:

- Base Measure
- Decision Criteria

- Derived Measure
- Measurement Function

Standards:

- ISO/IEC 15939

6.217. Mock Object

Temporary dummy objects created to aid testing until the real objects become available. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Testing

Standards:

- ISO/IEC/IEEE 24765

6.218. Model

A semantically closed abstraction of a system or a complete description of a system from a particular perspective. [ISO/IEC/IEEE 24765]

Other Definitions

Model [IEEE 1233]: A representation of a real world process, device, or concept.

Model [ISO/IEC 15474]: A related collection of instances of meta-objects, representing (describing or prescribing) an information system, or parts thereof, such as a software product.

See also

Standards:

- IEEE 1233
- ISO/IEC 15474
- ISO/IEC/IEEE 24765

6.219. Modifiability

The ease with which a system can be changed without introducing defects. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Flexibility
- Maintainability

Standards:

→ ISO/IEC/IEEE 24765

6.220. Modifiable

Structured and has a style such that changes can be made completely, consistently, and correctly while retaining the structure. [ISO/IEC 12207]

See also

Standards:

→ ISO/IEC 12207

6.221. Modularity

The degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components. [ISO/IEC/IEEE 24765]

Other Definitions

Modularity [ISO/IEC/IEEE 24765]: Software attributes that provide a structure of highly independent components.

Modularity [ISO/IEC/IEEE 24765]: The extent to which a routine or module is like a black box

See also

Glossary:

→ Cohesion

→ Coupling

→ Module

Standards:

→ ISO/IEC/IEEE 24765

6.222. Module

A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading. [ISO/IEC/IEEE 24765]

Other Definitions

Module [ISO/IEC/IEEE 24765]:

1. A logically separable part of a program.
2. A set of source code files under version control that can be manipulated together as one.
3. A collection of both data and the routines that act on it.

Notes

- The terms 'module', 'component,' and 'unit' are often used interchangeably or defined to be sub-elements of one another in different ways depending upon the context. The relationship of these terms is not yet standardized.

See also

Glossary:

- Component
- Modularity

Standards:

- ISO/IEC/IEEE 24765

6.223. Moke Object

Temporary dummy objects created to aid testing until the real objects become available. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Testing

Standards:

- ISO/IEC/IEEE 24765

6.224. Multidimensional Analysis

Multidimensional analysis is a measurement function that weights different base measures to give a more relevant insight of the final goal of the measure.

It was primarily developed by Kaner and Bond in Software Engineering Metrics: What Do They Measure And How Do We Know [https://maisqual.squoring.com/wiki/index.php/Software_Engineering_Metrics:_What_Do_They_Measure_And_How_Do_We_Know].

See also

- Software Engineering Metrics: What Do They Measure And How Do We Know [https://maisqual.squoring.com/wiki/index.php/Software_Engineering_Metrics:_What_Do_They_Measure_And_How_Do_We_Know]

6.225. Network

An arrangement of nodes and interconnecting branches. [ISO/IEC 2382-1]

See also

Standards:

→ ISO/IEC 2382

6.226. Nonfunctional Requirement

A software requirement that describes not what the software will do but how the software will do it. [ISO/IEC/IEEE 24765]

Notes

→ Software performance requirements, software external interface requirements, software design constraints, and software quality attributes. Nonfunctional requirements are sometimes difficult to test, so they are usually evaluated subjectively. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Functional Requirement

→ Requirement

Standards:

→ ISO/IEC/IEEE 24765

6.227. Nontechnical Requirement

Requirement affecting product and service acquisition or development that is not a property of the product or service. [ISO/IEC/IEEE 24765]

Notes

→ Example: numbers of products or services to be delivered; data rights for delivered COTS nondevelopmental items; delivery dates; milestones with exit criteria; work constraints associated with training, site provisions, and deployment schedules. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Requirement

→ Technical Requirement

Standards:

→ ISO/IEC/IEEE 24765

6.228. Object

An encapsulation of data and services that manipulate that data. [ISO/IEC/IEEE 24765]

Other Definitions

Object [ISO/IEC/IEEE 24765]: A specific entity that exists in a program at runtime in object-oriented programming.

Object [ISO/IEC/IEEE 24765]: Pertaining to the outcome of an assembly or compilation process.

Object [IEEE 1320]: A member of an object set and an instance of an object type.

See also

Glossary:

→ Object Model

Standards:

→ IEEE 1320

→ ISO/IEC/IEEE 24765

6.229. Object Model

An integrated abstraction that treats all activities as performed by collaborating objects and encompassing both the data and the operations that can be performed against that data. [ISO/IEC 12207]

Notes

→ An object model captures both the meanings of the knowledge and actions of objects behind the abstraction of responsibility. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC 12207

→ ISO/IEC/IEEE 24765

6.230. Object Oriented Design

A software development technique in which a system or component is expressed in terms of objects and connections between those objects. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Object

→ Object Model

Standards:

→ ISO/IEC/IEEE 24765

6.231. Observation

Instance of applying a measurement procedure to produce a value for a base measure. [ISO/IEC 15939, ISO/IEC 25000]

See also

Standards:

- ISO/IEC 15939
- ISO/IEC 25000

6.232. Observation Period

The time interval, where the measurement procedure is observed for collecting (logging) measurement results for rating or validation, consisting of the rating interval and the supplementary run. [ISO/IEC 14756]

See also

Standards:

- ISO/IEC 14756

6.233. Operability

The capability of the software product to enable the user to operate and control it. [ISO/IEC 9126-1]

Notes

- Aspects of suitability, changeability, adaptability and installability may affect operability. [ISO/IEC 9126-1]
- Operability corresponds to controllability, error tolerance and conformity with user expectations as defined in ISO 9241-10. [ISO/IEC 9126-1]
- For a system which is operated by a user, the combination of functionality, reliability, usability and efficiency can be measured externally by quality in use. [ISO/IEC 9126-1]

See also

Standards:

- ISO/IEC 9126-1
- ISO 9241-10

6.234. Operand

A variable, constant, or function upon which an operation is to be performed. [ISO/IEC/IEEE 24765]

Notes

- Example: in the expression $A = B + 3$, B and 3 are the operands. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.235. Operational Testing

Testing conducted to evaluate a system or component in its operational environment. [ISO/IEC 15504]

See also

Glossary:

- Acceptance Testing
- Development Testing
- Qualification Testing
- Testing

Standards:

- IEEE 829
- ISO/IEC 15504

6.236. Operator

Individual or organisation that operates the system. [ISO/IEC 12207, ISO/IEC 15939, ISO/IEC 25000]

Other Definitions

Operator [ISO/IEC/IEEE 24765]: A mathematical or logical symbol that represents an action to be performed in an operation.

Operator [ISO/IEC 12207, ISO/IEC 15288]: Entity that performs the operation of a system.

Operator [IEEE 1220]: An individual or an organization that contributes to the functionality of a system and draws on knowledge, skills, and procedures to contribute the function.

Notes

- The role of operator and the role of user may be vested, simultaneously or sequentially, in the same individual or organization. An individual operator combined with knowledge, skills and procedures may be considered as an element of the system. In the context of this specific definition, the term entity means an individual or an organization. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Operand
- Operator Manual
- User

Standards:

- IEEE 1220
- ISO/IEC 12207
- ISO/IEC 15939

- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.237. Operator Manual

A document that provides the information necessary to initiate and operate a system or component. [ISO/IEC/IEEE 24765]

Notes

- Typically described are procedures for preparation, operation, monitoring, and recovery. An operator manual is distinguished from a user manual when a distinction is made between those who operate a computer system (mounting tapes, etc) and those who use the system for its intended purpose. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Document
- Installation Manual
- Maintenance Manual
- Operator
- Support Manual
- User Manual

Standards:

- ISO/IEC/IEEE 24765

6.238. Optional Attribute

An attribute that may have no value for an instance. [IEEE 1320]

Notes

- Typically described are procedures for preparation, operation, monitoring, and recovery. An operator manual is distinguished from a user manual when a distinction is made between those who operate a computer system (mounting tapes, etc) and those who use the system for its intended purpose. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Attribute

Standards:

- IEEE 1320

6.239. Optional Requirement

Requirement of a normative document that must be fulfilled in order to comply with a particular option permitted by that document. [ISO/IEC 14143]

Notes

- Typically described are procedures for preparation, operation, monitoring, and recovery. An operator manual is distinguished from a user manual when a distinction is made between those who operate a computer system (mounting tapes, etc) and those who use the system for its intended purpose. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Requirement

Standards:

- ISO/IEC 14143

6.240. Organisational Unit

The part of an organisation that is the subject of measurement. [ISO/IEC 15504-9, ISO/IEC 15939]

Notes

- An organisational unit deploys one or more processes that operate within a coherent set of business goals. [ISO/IEC 15939]

See also

Standards:

- ISO/IEC 15504
- ISO/IEC 15939

6.241. Path

In software engineering, a sequence of instructions that may be performed in the execution of a computer program. [ISO/IEC/IEEE 24765]

Other Definitions

Path [ISO/IEC/IEEE 24765]: In file access, a hierarchical sequence of directory and subdirectory names specifying the storage location of a file.

See also

Standards:

- ISO/IEC/IEEE 24765

6.242. Path Analysis

Analysis of a computer program to identify all possible paths through the program, to detect incomplete paths, or to discover portions of the program that are not on any path. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Path
- Path Testing

Standards:

- ISO/IEC/IEEE 24765

6.243. Path Testing

Testing designed to execute all or selected paths through a computer program. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Branch Testing
- Path
- Path Analysis
- Testing

Standards:

- ISO/IEC/IEEE 24765

6.244. Pathological Coupling

A type of coupling in which one software module affects or depends upon the internal implementation of another. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Content Coupling
- Control Coupling
- Coupling
- Data Coupling
- Hybrid Coupling

Standards:

- ISO/IEC/IEEE 24765

6.245. Peer Review

A review of a software work product, following defined procedures, by peers of the producers of the product for the purpose of identifying defects and improvements. [CMMi]

Other Definitions

Peer Review [ISO/IEC/IEEE 24765]: Review of work products performed by peers during development of the work products to identify defects for removal.

See also

Glossary:

→ Inspection

Standards:

→ CMMi

→ ISO/IEC/IEEE 24765

6.246. Performance

The degree to which a system or component accomplishes its designated functions within given constraints, such as speed, accuracy, or memory usage. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Level of Performance

→ Performance Indicator

→ Performance Testing

→ Process Performance

Standards:

→ ISO/IEC/IEEE 24765

6.247. Performance Indicator

An assessment indicator that supports the judgment of the process performance of a specific process. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Indicator

→ Performance

Standards:

→ ISO/IEC/IEEE 24765

6.248. Performance Testing

Testing conducted to evaluate the compliance of a system or component with specified performance requirements. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Functional Testing
- Performance
- Testing

Standards:

- ISO/IEC/IEEE 24765

6.249. Pilot Project

A project designed to test a preliminary version of an information processing system under actual but limited operating conditions and which will then be used to test the definitive version of the system. [ISO/IEC 2382]

See also

Standards:

- ISO/IEC 2382

6.250. Portability

The capability of the software product to be transferred from one environment to another. [ISO/IEC 9126-1]

Other Definitions

Portability [ISO/IEC/IEEE 24765]: The ease with which a system or component can be transferred from one hardware or software environment to another.

Portability [ISO/IEC 2382]: The capability of a program to be executed on various types of data processing systems without converting the program to a different language and with little or no modification.

Notes

- The environment may include organisational, hardware or software environment. [ISO/IEC 9126-1]

See also

- ISO/IEC 9126-1

6.251. Portability Compliance

The capability of the software product to adhere to standards or conventions relating to portability. [ISO/IEC 9126-1]

See also

Glossary:

→ Portability

Standards:

→ ISO/IEC 9126-1

6.252. Practice

An activity that contributes to the purpose or outcomes of a process or enhances the capability of a process.
[ISO/IEC 15504]

Other Definitions

Practice [ISO/IEC/IEEE 24765]: Requirements employed to prescribe a disciplined uniform approach to the software development process.

Practice [IEEE 1490]: A specific type of professional or management activity that contributes to the execution of a process and that may employ one or more techniques and tools.

See also

Glossary:

→ Key Practices

Standards:

→ IEEE 1490

→ ISO/IEC 15504

→ ISO/IEC/IEEE 24765

6.253. Precision

The degree of exactness or discrimination with which a quantity is stated. [ISO/IEC/IEEE 24765]

Notes

→ Example: a precision of 2 decimal places versus a precision of 5 decimal places. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Accuracy

Standards:

→ ISO/IEC/IEEE 24765

6.254. Predictive Metric

A metric applied during development and used to predict the values of a software quality factor. [IEEE 1061]

See also

Glossary:

→ Metric

Standards:

→ IEEE 1061

6.255. Procedure

Ordered series of steps that specify how to perform a task. [ISO/IEC 26514]

Other Definitions

Procedure [ISO/IEC 19770]: Specified way to carry out an activity or process.

Procedure [ISO/IEC/IEEE 24765]: A portion of a computer program that is named and that performs a specific action.

Procedure [ISO/IEC/IEEE 24765]: A routine that does not return a value.

See also

Glossary:

→ Action

→ Step

→ Process

Standards:

→ ISO/IEC 19770

→ ISO/IEC 26514

→ ISO/IEC/IEEE 24765

6.256. Process

System of activities, which use resources to transform inputs into outputs. [ISO/IEC" 25000]

Other Definitions

Process [ISO/IEC 15504-9, ISO/IEC 15939]: Set of interrelated activities that transform inputs into outputs.

Process [ISO/IEC 2382]: Predetermined course of events defined by its purpose or by its effect, achieved under given conditions.

Process (verb) [ISO/IEC/IEEE 24765]: To perform operations on data.

Process [ISO/IEC 15414]: A collection of steps taking place in a prescribed manner and leading to an objective.

Process [ISO/IEC 2382]: In data processing, the predetermined course of events that occur during the execution of all or part of a program.

Notes

- In [ISO 9000:2005] the term "activities" covers use of resources. A process may have multiple starting points and multiple end points. The prescribed manner may be a partially ordered sequence. A process specification can be a workflow specification. An enterprise specification may define types of processes and may define process templates. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC 2382
- ISO/IEC 15504
- ISO/IEC 15939
- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.257. Process Assessment

A disciplined evaluation of an organizational unit's processes against a Process Assessment Model. [ISO/IEC 15504]

See also

Glossary:

- Process
- Process Assessment Model

Standards:

- ISO/IEC 15504

6.258. Process Assessment Model

A model suitable for the purpose of assessing process capability, based on one or more process reference models. [ISO/IEC 15504]

See also

Glossary:

- Process
- Process Assessment

Standards:

- ISO/IEC 15504

6.259. Process Capability

A characterization of the ability of a process to meet current or projected business goals. [ISO/IEC 15504]

See also

Glossary:

- Process
- Process Capability Determination

Standards:

- ISO/IEC 15504

6.260. Process Capability Determination

A systematic assessment and analysis of selected processes within an organization against a target capability, carried out with the aim of identifying the strengths, weaknesses and risks associated with deploying the processes to meet a particular specified requirement. [ISO/IEC 15504]

See also

Glossary:

- Process
- Process Capability

Standards:

- ISO/IEC 15504

6.261. Process Capability Level

A point on the six-point ordinal scale (of process capability) that represents the capability of the process; each level builds on the capability of the level below. [ISO/IEC 15504]

See also

Glossary:

- Process
- Process Capability

Standards:

- ISO/IEC 15504

6.262. Process Context

The set of factors, documented in the assessment input, that influence the judgment, comprehension and comparability of process attribute ratings. [ISO/IEC 15504]

See also

Glossary:

→ Process

Standards:

→ ISO/IEC 15504

6.263. Process Improvement

Actions taken to change an organization's processes so that they more effectively and/or efficiently meet the organization's business goals. [ISO/IEC 15504]

See also

Glossary:

→ Process

Standards:

→ ISO/IEC 15504

6.264. Process Improvement Objective

Set of target characteristics established to guide the effort to improve an existing process in a specific, measurable way, either in terms of resultant product or service characteristics, such as quality, performance, and conformance to standards, or in the way in which the process is executed, such as elimination of redundant process steps, combination of process steps, and improvement of cycle time. [ISO/IEC 15504]

See also

Glossary:

→ Process

→ Process Improvement

Standards:

→ ISO/IEC 15504

6.265. Process Improvement Program

The strategies, policies, goals, responsibilities and activities concerned with the achievement of specified improvement goals. [ISO/IEC 15504]

Notes

→ A process improvement program can span more than one complete cycle of process improvement. [ISO/IEC 15504]

See also

Glossary:

- Process
- Process Improvement

Standards:

- ISO/IEC 15504

6.266. Process Improvement Project

A subset of the Process Improvement Program that forms a coherent set of actions to achieve a specific improvement. [ISO/IEC 15504]

See also

Glossary:

- Process
- Process Improvement Program

Standards:

- ISO/IEC 15504

6.267. Process Metric

A metric used to measure characteristics of the methods, techniques, and tools employed in developing, implementing, and maintaining the software system. [IEEE 1061]

See also

Glossary:

- Process
- Metric

Standards:

- IEEE 1061

6.268. Process Outcome

An observable result of a process. [ISO/IEC 15504]

Other Definitions

Process Outcome [ISO/IEC 12207, ISO/IEC 15288]: Observable result of the successful achievement of the process purpose.

Notes

- An outcome is an artifact, a significant change of state or the meeting of specified constraints. An outcome statement describes one of the following: production of an artifact; a significant change in state; meeting of specified constraints, e.g., requirements, goals. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Process

Standards:

- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC 15504
- ISO/IEC/IEEE 24765

6.269. Process Performance

The extent to which the execution of a process achieves its purpose. [ISO/IEC 15504]

See also

Glossary:

- Performance
- Performance Indicator
- Process

Standards:

- ISO/IEC 15504

6.270. Process Purpose

High-level objective of performing the process and the likely outcomes of effective implementation of the process. [ISO/IEC 15504, ISO/IEC 15288]

Notes

- The implementation of the process should provide tangible benefits to the stakeholders. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Process

Standards:

- ISO/IEC 15288

- ISO/IEC 15504
- ISO/IEC/IEEE 24765

6.271. Product

An artifact that is produced, is quantifiable, and can be either an end item in itself or a component item. [IEEE 1490]

Other Definitions

Product [ISO/IEC 26514]: Complete set of software and documentation.

Product [IEEE 1074]: Output of the software development activities (e.g., document, code, or model).

Product [ISO/IEC 15939]: Result of a process.

Software Product [ISO/IEC 9126, ISO/IEC 12207, ISO/IEC 15939]: Set of computer programs, procedures, and associated documentation and data.

Notes

- In ISO 9000 there are four agreed generic product categories:
- * hardware (e.g., engine mechanical part);
- * software (e.g., computer program);
- * services (e.g., transport); and
- * processed materials (e.g., lubricant).

:Hardware and processed materials are generally tangible products, while software or services are generally intangible. Most products comprise elements belonging to different generic product categories. Whether the product is then called hardware, processed material, software, or service depends on the dominant element. [ISO/IEC/IEEE 24765]

- Products include intermediate products, and products intended for users such as developers and maintainers. [ISO/IEC 9126]

See also

Glossary:

- Work Product

Standards:

- IEEE 1074
- IEEE 1490
- ISO/IEC 9126
- ISO/IEC 12207
- ISO/IEC 15939
- ISO/IEC 15939
- ISO/IEC 26514

6.272. Product Line

Group of products or services sharing a common, managed set of features that satisfy specific needs of a selected market or mission. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Product
- Software Product

Standards:

- ISO/IEC/IEEE 24765

6.273. Product Metric

A metric used to measure the characteristics of any intermediate or final product of the software development process. [IEEE 1061]

See also

Glossary:

- Product
- Metric
- Software Product

Standards:

- IEEE 1061

6.274. Productivity

The capability of the software product to enable users to expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use. [ISO/IEC 9126-1]

Notes

- Relevant resources can include time to complete the task, the user's effort, materials or the financial cost of usage. [ISO/IEC 9126-1]

See also

- ISO/IEC 9126-1

6.275. Programmer Manual

A document that provides the information necessary to develop or modify software for a given computer system. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Document
- Installation Manual
- Maintenance Manual
- Operator Manual
- User Manual

Standards:

- ISO/IEC/IEEE 24765

6.276. Project

Endeavor with defined start and finish dates undertaken to create a product or service in accordance with specified resources and requirements. [ISO/IEC 15288, ISO/IEC 15939]

Other Definitions

Project [ISO/IEC 2382]: An undertaking with pre-specified objectives, magnitude and duration.

Project [IEEE 1490]: A temporary endeavor undertaken to create a unique product, service, or result.

Notes

- A project may be viewed as a unique process comprising coordinated and controlled activities and may be composed of activities from the Project Processes and Technical Processes. [ISO/IEC/IEEE 24765]

See also

Standards:

- IEEE 1490
- ISO/IEC 2382
- ISO/IEC 15288
- ISO/IEC 15939
- ISO/IEC/IEEE 24765

6.277. Project Management

The application of knowledge, skills, tools, and techniques to project activities to meet the project requirements. [IEEE 1490]

Other Definitions

Project Management [ISO/IEC 2382]: The activities concerned with project planning and project control.

See also

Glossary:

→ Project

Standards:

→ IEEE 1490

→ ISO/IEC 2382

6.278. Project Phase

A collection of logically related project activities, usually culminating in the completion of a major deliverable. [IEEE 1490]

Notes

→ Project phases are mainly completed sequentially, but can overlap in some project situations. A project phase is a component of a project life cycle. A project phase is not a project management process group. [IEEE 1490]

See also

Glossary:

→ Project

Standards:

→ IEEE 1490

6.279. Prototype

Model or preliminary implementation of a piece of software suitable for the evaluation of system design, performance or production potential, or for the better understanding of the software requirements. [ISO/IEC 15910]

Other Definitions

Prototype [ISO/IEC/IEEE 24765]: A preliminary type, form, or instance of a system that serves as a model for later stages or for the final, complete version of the system.

Notes

→ A prototype is used to get feedback from users for improving and specifying a complex human interface, for feasibility studies, or for identifying requirements. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Project Phase

Standards:

→ ISO/IEC 15910

→ ISO/IEC/IEEE 24765

6.280. Qualification

Process of demonstrating whether an entity is capable of fulfilling specified requirements. [ISO/IEC 12207, ISO/IEC 15288]

Other Definitions

Qualification [ISO/IEC/IEEE 24765]: The process of determining whether a system or component is suitable for operational use.

See also

Standards:

- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC/IEEE 24765

6.281. Qualification Testing

Testing, conducted by the developer and witnessed by the acquirer (as appropriate), to demonstrate that a software product meets its specifications and is ready for use in its target environment or integration with its containing system. [ISO/IEC 12207]

Other Definitions

Qualification Testing [IEEE 829]: Testing conducted to determine whether a system or component is suitable for operational use.

See also

Glossary:

- Acceptance Testing
- Development Testing
- Operational Testing
- Testing

Standards:

- IEEE 829
- ISO/IEC 12207

6.282. Quality

The totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs. [ISO 8402, ISO/IEC 9126-1]

Other Definitions

Quality [IEEE 829]: The degree to which a system, component, or process meets specified requirements.

Quality [ISO/IEC/IEEE 24765]: Ability of a product,service, system, component, or process to meet customer or user needs, expectations, or requirements.

Quality [IEEE 1490]: The degree to which a set of inherent characteristics fulfils requirements.

Quality [IEEE 829]: The degree to which a system, component, or process meets customer or user needs or expectations.

Notes

- In a contractual environment, or in a regulated environment, such as the nuclear safety field, needs are specified, whereas in other environments, implied needs should be identified and defined. [ISO 8402]
- In ISO/IEC 14598 the relevant entity is a software product. [ISO/IEC 9126-1]

See also

Glossary:

- Quality Assurance
- Software Quality

Standards:

- IEEE 829
- IEEE 1490
- ISO 8402
- ISO/IEC 9126-1
- ISO/IEC 14598
- ISO/IEC/IEEE 24765

6.283. Quality Assurance

The planned and systematic activities implemented within the quality system, and demonstrated as needed, to provide adequate confidence that an entity will fulfil requirements for quality. [ISO/IEC 12207]

Other Definitions

Quality Assurance [IEEE 610.12, ISO/IEC/IEEE 24765]: A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements.

Quality Assurance [IEEE 610.12, ISO/IEC/IEEE 24765]: A set of activities designed to evaluate the process by which products are developed or manufactured.

Quality Assurance [ISO/IEC 15288]: Part of quality management focused on providing confidence that quality requirements will be fulfilled.

Notes

- There are both internal and external purposes for quality assurance: within an organization, quality assurance provides confidence to management; in contractual situations, quality assurance provides confidence to the customer or others. Some quality control and quality assurance actions are interrelated.

Unless requirements for quality fully reflect the needs of the user, quality assurance may not provide adequate confidence. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Quality
- Quality Control
- Quality Management

Standards:

- IEEE 610.12
- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC/IEEE 24765

6.284. Quality Control

A set of activities designed to evaluate the quality of developed or manufactured products. [IEEE 610.12]

Notes

- This term has no standardized meaning in software engineering at this time. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Quality
- Quality Assurance

Standards:

- IEEE 610.12
- ISO/IEC/IEEE 24765

6.285. Quality Evaluation

Systematic examination of the extent to which an entity is capable of fulfilling specified requirements. [ISO 8402, ISO/IEC 9126-1, ISO/IEC 14598]

Notes

- The requirements may be formally specified, as when a product is developed for a specific user under a contract, or specified by the development organisation, as when a product is developed for unspecified users, such as consumer software, or the requirements may be more general, as when a user evaluates products for comparison and selection purpose. [ISO/IEC 9126-1]

See also

Glossary:

→ Quality

Standards:

→ ISO 8402

→ ISO/IEC 9126-1

→ ISO/IEC 14598

6.286. Quality Factor

A management-oriented attribute of software that contributes to its quality. [IEEE 1061]

See also

Glossary:

→ Direct Metric

Standards:

→ IEEE 1061

6.287. Quality Management

Coordinated activities to direct and control an organization with regard to quality. [ISO/IEC 19759]

See also

Glossary:

→ Quality

Standards:

→ ISO/IEC 19759

6.288. Quality Measure Element

Base measure or derived measure that is used for constructing software quality measures. [ISO/IEC 25000]

Notes

→ The software quality characteristics or sub-characteristics of the entity are derived afterwards by calculating a software quality measure. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Quality

Standards:

→ ISO/IEC 25000

- ISO/IEC 25021
- ISO/IEC/IEEE 24765

6.289. Quality Metric

a quantitative measure of the degree to which an item possesses a given quality attribute. [ISO/IEC/IEEE 24765]

Other Definitions

Quality Metric [ISO/IEC/IEEE 24765]: A function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software possesses a given quality attribute.

See also

Glossary:

- Quality

Standards:

- ISO/IEC/IEEE 24765

6.290. Quality Model

Defined set of characteristics, and of relationships between them, which provides a framework for specifying quality requirements and evaluating quality. [ISO/IEC 25000]

Other Definitions

Quality Model [ISO/IEC 9126-1, ISO/IEC 14598-1]: The set of characteristics and the relationships between them which provide the basis for specifying quality requirements and evaluating quality.

See also

Glossary:

- Quality

Standards:

- ISO/IEC 9126
- ISO/IEC 14598
- ISO/IEC 25000

6.291. Quality in Use

The capability of the software product to enable specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in specified contexts of use. [ISO/IEC 9126-1, ISO/IEC/IEEE 24765, ISO/IEC 25000]

Notes

- Quality in use is the user's view of the quality of an environment containing software, and is measured from the results of using the software in the environment, rather than properties of the software itself. [ISO/IEC 9126-1]
- This definition of quality in use is similar to the definition of usability in ISO 9241-11. In ISO/IEC 14598 the term usability is used to refer to the software quality characteristic described in ISO/IEC 9126-1. [ISO/IEC 9126-1]
- The definition of quality in use in ISO/IEC 14598-1 does not currently include the new characteristic of safety. [ISO/IEC 9126-1]
- Usability is defined in ISO 9241-11 in a similar way to the definition of quality in use in this part of ISO/IEC 9126. Quality in use may be influenced by any of the quality characteristics, and is thus broader than usability, which is defined in this part of ISO/IEC 9126 in terms of understandability, learnability, operability, attractiveness and compliance. [ISO/IEC 9126-1]
- Before the product is released, quality in use can be specified and measured in a test environment for the intended users, goals and contexts of use. Once in use, it can be measured for actual users, goals and contexts of use. The actual needs of users may not be the same as those anticipated in requirements, so actual quality in use may be different from quality in use measured earlier in a test environment. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Internal Quality
- External Quality

Standards:

- ISO/IEC 9126-1
- ISO/IEC 9126-4
- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.292. Rating

The action of mapping the measured value to the appropriate rating level. Used to determine the rating level associated with the software for a specific quality characteristic. [ISO/IEC 9126-1, ISO/IEC 14598-1, ISO/IEC 25000]

Notes

- Used to determine the rating level associated with the software for a specific quality characteristic. Rating and rating levels can be applied to characteristics other than quality characteristics. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Rating Level

Standards:

- ISO/IEC 9126
- ISO/IEC 14598
- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.293. Rating Level

A scale point on an ordinal scale which is used to categorise a measurement scale. [ISO/IEC 9126-1, ISO/IEC 14598, ISO/IEC 25000]

Notes

- The rating level enables software to be classified (rated) in accordance with the stated or implied needs. [ISO/IEC 9126-1]
- Appropriate rating levels may be associated with the different views of quality i.e. Users', Managers' or Developers'. [ISO/IEC 9126-1]

See also

Glossary:

- Rating
- Scale

Standards:

- ISO/IEC 9126
- ISO/IEC 14598
- ISO/IEC 25000

6.294. Readability

The ease with which a system's source code can be read and understood, especially at the detailed, statement level. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.295. Recoverability

The capability of the software product to re-establish a specified level of performance and recover the data directly affected in the case of a failure. [ISO/IEC 9126-1]

Notes

- Following a failure, a software product will sometimes be down for a certain period of time, the length of which is assessed by its recoverability. [ISO/IEC 9126-1]
- Availability is the capability of the software product to be in a state to perform a required function at a given point in time, under stated conditions of use. Externally, availability can be assessed by the proportion

of total time during which the software product is in an up state. Availability is therefore a combination of maturity (which governs the frequency of failure), fault tolerance and recoverability (which governs the length of down time following each failure). For this reason it has not been included as a separate sub-characteristic. [ISO/IEC 9126-1]

See also

Glossary:

→ Availability

Standards:

→ ISO/IEC 9126-1

6.296. Recovery

The restoration of a system, program, database, or other system resource to a state in which it can perform required functions. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC/IEEE 24765

6.297. Reengineering

The examination and alteration of software to reconstitute it in a new form, including the subsequent implementation of the new form. [ISO/IEC 19759]

See also

Glossary:

→ Process

Standards:

→ ISO/IEC 19759

6.298. Regression Testing

Selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements. [ISO/IEC 90003]

Other Definitions

Regression Testing [ISO/IEC 90003]: Testing required to determine that a change to a system component has not adversely affected functionality, reliability or performance and has not introduced additional defects.

See also

Glossary:

→ Testing

Standards:

→ ISO/IEC 90003

6.299. Release

Collection of new and/or changed configuration items which are tested and introduced into the live environment together. [ISO/IEC 20000]

Other Definitions

Release [ISO/IEC/IEEE 24765]: A software version that is made formally available to a wider community.

Release [IEEE 829, ISO/IEC 12207]: Particular version of a configuration item that is made available for a specific purpose.

Release [IEEE 829]: The formal notification and distribution of an approved version.

Notes

→ Release management includes defining acceptable quality levels for release, authority to authorize the release, and release procedures. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Process

Standards:

→ IEEE 829

→ ISO/IEC 12207

→ ISO/IEC 20000

→ ISO/IEC/IEEE 24765

6.300. Reliability

The capability of the software product to maintain a specified level of performance in cases of software faults or of infringement of its specified interface. [ISO/IEC 9126-1]

Other Definitions

Reliability [ISO/IEC/IEEE 24765]: The ability of a system or component to perform its required functions under stated conditions for a specified period of time.

Software Reliability [ISO/IEC/IEEE 24765]: The probability⁷ that software will not cause the failure of a system for a specified time under specified conditions.

⁷The probability is a function of the inputs to and use of the system as well as a function of the existence of faults in the software. The inputs to the system determine whether existing faults, if any, are encountered.

Notes

- ISO/IEC 9126-1 provides the exact same definition for reliability and Fault Tolerance.
- Wear or ageing does not occur in software. Limitations in reliability are due to faults in requirements, design, and implementation. Failures due to these faults depend on the way the software product is used and the program options selected rather than on elapsed time. [ISO/IEC 9126-1]
- The definition of reliability in ISO/IEC 2382-14:1997 is "The ability of functional unit to perform a required function...". In ISO/IEC 9126-1, functionality is only one of the characteristics of software quality. Therefore, the definition of reliability has been broadened to "maintain a specified level of performance..." instead of "...perform a required function". [ISO/IEC 9126-1]

See also

Glossary:

- Fault
- Fault Tolerance

Standards:

- ISO/IEC 9126-1

6.301. Reliability Compliance

The capability of the software product to adhere to standards, conventions or regulations relating to reliability. [ISO/IEC 9126-1]

See also

Glossary:

- Reliability

Standards:

- ISO/IEC 9126-1

6.302. Repeatability of Results of Measurements

Closeness of the agreement between the results of successive measurements of the same measurand carried out under the same conditions of measurement. [ISO/IEC 14143]

Notes

- These conditions are called repeatability conditions. Repeatability conditions include the same measurement procedure, the same observer, the same measuring instrument, used under the same conditions; the same location; repetition over a short period of time. Repeatability may be expressed quantitatively in terms of the dispersion characteristics of the results. [ISO/IEC 99]

See also

Glossary:

- Measurement

→ Reproducibility of Results of Measurements

Standards:

→ ISO/IEC 99

→ ISO/IEC 14143

6.303. Replaceability

The capability of the software product to be used in place of another specified software product for the same purpose in the same environment. [ISO/IEC 9126-1]

Notes

→ For example, the replaceability of a new version of a software product is important to the user when upgrading. [ISO/IEC 9126-1]

→ Replaceability is used in place of compatibility in order to avoid possible ambiguity with interoperability. [ISO/IEC 9126-1]

→ Replaceability may include attributes of both installability and adaptability. The concept has been introduced as a sub-characteristic of its own because of its importance. [ISO/IEC 9126-1]

See also

Standards:

→ ISO/IEC 9126-1

6.304. Reproducibility of Results of Measurements

Closeness of the agreement between the results of measurements of the same measurand carried out under changed conditions of measurement. [ISO/IEC 14143]

Notes

→ A valid statement of reproducibility requires specification of the conditions changed. The changed conditions may include the principle of measurement; method of measurement; observer; measuring instrument; reference standard; location; conditions of use; time. Reproducibility may be expressed quantitatively in terms of the dispersion characteristics of the results. Results are here usually understood to be corrected results. [ISO/IEC 99]

See also

Glossary:

→ Measurement

→ Repeatability of Results of Measurements

Standards:

→ ISO/IEC 99

→ ISO/IEC 14143

6.305. Request For Change

Form or screen used to record details of a request for a change to any configuration item within a service or infrastructure. [ISO/IEC 20000]

See also

Standards:

→ ISO/IEC 20000

6.306. Request For Information

A type of procurement document whereby the buyer requests a potential seller to provide various pieces of information related to a product or service or seller capability. [IEEE 1490]

See also

Standards:

→ IEEE 1490

6.307. Request For Proposal

A document used by the acquirer as a means to announce intention to potential bidders to acquire a specified system, product, or service. [ISO/IEC 15288]

Other Definitions

Request for Proposal [IEEE 1362]: A request for services, research, or a product prepared by a customer and delivered to prospective developers with the expectation that prospective developers will respond with their proposed cost, schedule, and development approach.

Request for Proposal [IEEE 1490]: A type of procurement document used to request proposals from prospective sellers of products or services. In some application areas, it may have a narrower or more specific meaning.

Request for Proposal [ISO/IEC/IEEE 24765]: A collection of formal documents that includes a description of the desired form of response from a potential supplier, the relevant statement of work for the supplier, and required provisions in the supplier agreement.

See also

Standards:

→ IEEE 1362

→ IEEE 1490

→ ISO/IEC 15288

→ ISO/IEC/IEEE 24765

6.308. Requirement

A condition or capability that must be met or possessed by a system, system component, product, or service to satisfy an agreement, standard, specification, or other formally imposed documents. [ISO/IEC/IEEE 24765]

Other Definitions

Requirement [ISO/IEC/IEEE 24765]: A condition or capability needed by a user to solve a problem or achieve an objective.

Requirement [IEEE 1490]: A condition or capability that must be met or possessed by a system, product, service, result, or component to satisfy a contract, standard, specification, or other formally imposed document. Requirements include the quantified and documented needs, wants, and expectations of the sponsor, customer, and other stakeholders.

Software Requirement [ISO/IEC/IEEE 24765]: A software capability needed by a user to solve a problem to achieve an objective.

Software Requirement [ISO/IEC/IEEE 24765]: A software capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.

Notes

- There are design requirement, functional requirement, implementation requirement, interface requirement, performance requirement, physical requirement. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Functional Requirement
- Nonfunctional Requirement
- Nontechnical Requirement
- Optional Requirement
- Technical Requirement

Standards:

- IEEE 1490
- ISO/IEC/IEEE 24765

6.309. Requirements Analysis

The process of studying user needs to arrive at a definition of system, hardware, or software requirements. [ISO/IEC/IEEE 24765]

Other Definitions

Requirements Analysis [ISO/IEC/IEEE 24765]: The process of studying and refining system, hardware, or software requirements.

Requirements Analysis [ISO/IEC 2382]: A systematic investigation of user requirements to arrive at a definition of a system.

Requirements Analysis [ISO/IEC/IEEE 24765]: Determination of product- or service-specific performance and functional characteristics based on analyses of customer needs, expectations, and constraints; operational

concept; projected utilization environments for people, products, services, and processes; and measures of effectiveness

See also

Glossary:

→ Requirement

Standards:

→ ISO/IEC 2382

→ ISO/IEC/IEEE 24765

6.310. Requirements Derivation

The changing or translation of a requirement through analysis into a form that is suitable for low-level analysis or design. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Requirement

Standards:

→ ISO/IEC/IEEE 24765

6.311. Requirements Document

Document containing any combination of requirements or regulations to be met by a COTS software product. [ISO/IEC 25051]

Example

→ A technical or ergonomic standard, a requirements list (or model requirements specification) from a group (e.g. a market sector, technical or user association), a law or a decree. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Document

→ Requirement

Standards:

→ ISO/IEC 25051

→ ISO/IEC/IEEE 24765

6.312. Requirements Engineering

The science and discipline concerned with analyzing and documenting requirements. [ISO/IEC/IEEE 24765]

Notes

- It comprises needs analysis, requirements analysis, and requirements specification. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Requirement
- Requirements Analysis
- Requirements Specification

Standards:

- ISO/IEC/IEEE 24765

6.313. Requirements Partitioning

The separation or decomposing of a top-level requirement or design into successively lower-level detailed requirements or design. [ISO/IEC/IEEE 24765]

Notes

- Requirements Decomposition is a synonym for Requirements Partitioning. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.314. Requirements Review

A process or meeting during which the requirements for a system, hardware item, or software item are presented to project personnel, managers, users, customers, or other interested parties for comment or approval. [ISO/IEC/IEEE 24765]

Notes

- Types of Requirements Review include system requirements review, software requirements review. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Requirement

Standards:

- ISO/IEC/IEEE 24765

6.315. Requirements Specification

A document that specifies the requirements for a system or component. [ISO/IEC/IEEE 24765]

Notes

→ Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Requirement
- Requirements Document

Standards:

- ISO/IEC/IEEE 24765

6.316. Requirements Traceability

Discernible association between a requirement and related requirements, implementations, and verifications. [ISO/IEC/IEEE 24765]

Other Definitions

Requirements Traceability [ISO/IEC/IEEE 24765]: the identification and documentation of the derivation path (upward) and allocation/ flow-down path

(downward) of requirements in the requirements hierarchy.

See also

Glossary:

- Requirement

Standards:

- ISO/IEC/IEEE 24765

6.317. Requirements Traceability Matrix

A table that links requirements to their origin and traces them throughout the project life cycle. [IEEE 1490]

See also

Glossary:

- Requirement
- Requirements Traceability

Standards:

- IEEE 1490

6.318. Resource

Skilled human resources (specific disciplines either individually or in crews or teams), equipment, services, supplies, commodities, materiel, budgets, or funds. [IEEE 1490]

Other Definitions

Resource [ISO/IEC 12207, ISO/IEC 15288]: Asset that is utilized or consumed during the execution of a process.

Resource [ISO/IEC 15414]: A role (with respect to that action) in which the enterprise object fulfilling the role is essential to the action, requires allocation, or may become unavailable.

Resource [ISO/IEC 15414]: An enterprise object which is essential to some behavior and which requires allocation or may become unavailable.

Example

→ Diverse entities such as funding, personnel, facilities, capital equipment, tools, and utilities such as power, water, fuel and communication infrastructures. [ISO/IEC/IEEE 24765]

Notes

→ Allocation of a resource may constrain other behaviors for which that resource is essential. Resources may be reusable, renewable or consumable. A consumable resource may become unavailable after some amount of use or after some amount of time (in case a duration or expiry has been specified for the resource). [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC 15414
- ISO/IEC/IEEE 24765

6.319. Resource Utilisation

The capability of the software product to use appropriate amounts and types of resources when the software performs its function under stated conditions. [ISO/IEC 9126-1]

Notes

→ Human resources are included as part of productivity. [ISO/IEC 9126-1]

See also

- ISO/IEC 9126-1

6.320. Result

An output from performing project management processes and activities. Results include outcomes (e.g., integrated systems, revised process, restructured organization, tests, trained personnel, etc.) and documents (e.g., policies, plans, studies, procedures, specifications, reports, etc.). [IEEE 1490]

Notes

- May include values as well as status information indicating that exceptional conditions were raised in attempting to perform the requested service. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Deliverable
- Product

Standards:

- IEEE 1490
- ISO/IEC/IEEE 24765

6.321. Retirement

Withdrawal of active support by the operation and maintenance organization, partial or total replacement by a new system, or installation of an upgraded system. [ISO/IEC 12207, ISO/IEC 15288]

Other Definitions

Retirement [ISO/IEC/IEEE 24765]: Removal of support from an operational system or component.

Retirement [ISO/IEC/IEEE 24765]: Permanent removal of a system or component from its operational environment.

See also

Standards:

- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC/IEEE 24765

6.322. Reverse Engineering

Determining what existing software will do and how it is constructed (to make intelligent changes). [ISO/IEC/IEEE 24765]

Notes

Reverse Engineering [ISO/IEC/IEEE 24765]: Software engineering approach that derives a system's design or requirements from its code.

See also

Standards:

→ ISO/IEC/IEEE 24765

6.323. Risk

An uncertain event or condition that, if it occurs, has a positive or negative effect on a project's objectives. [IEEE 1490]

Other Definitions

Risk [IEEE 829]: The combination of the probability of an abnormal event or failure and the consequence(s) of that event or failure to a system's components, operators, users, or environment.

Risk [ISO/IEC 15026]: A function of the probability of occurrence of a given threat and the potential adverse consequences of that threat's occurrence.

Risk [IEEE 829]: The combination of the probability of occurrence and the consequences of a given future undesirable event.

Notes

→ Generally used only when there is at least the possibility of negative consequences. In some situations, risk arises from the possibility of deviation from the expected outcome or event. Risk can be associated with software, systems, products, and projects. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Risk Acceptance

→ Risk Analysis

Standards:

→ IEEE 829

→ IEEE 1490

→ ISO/IEC/IEEE 24765

6.324. Risk Acceptance

Acknowledgment of a risk factor's existence along with a decision to accept the consequences if the corresponding problem occurs. [ISO/IEC/IEEE 24765]

Other Definitions

Risk Acceptance [IEEE 1490]: A risk response planning technique that indicates that the project team has decided not to change the project management plan to deal with a risk, or is unable to identify any other suitable response strategy.

Notes

→ Risk acceptance depends on risk criteria. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Risk

Standards:

→ IEEE 1490

→ ISO/IEC/IEEE 24765

6.325. Risk Analysis

The process of examining identified risk factors for probability of occurrence, potential loss, and potential risk-handling strategies. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Risk

Standards:

→ ISO/IEC/IEEE 24765

6.326. Robustness

The degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Fault Tolerance

Standards:

→ ISO/IEC/IEEE 24765

6.327. Role

The participation of an entity in a relationship. [ISO/IEC 15474-1]

Other Definitions

Role [IEEE 1490]: A defined function to be performed by a project team member, such as testing, filing, inspecting, coding.

Notes

- Each instance of a role has a minimum and maximum cardinality, and may be attributed. The direction of the role indicates how to read the name of the role. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Actor

Standards:

- IEEE 1490
- ISO/IEC 15474
- ISO/IEC/IEEE 24765

6.328. Routine

A subprogram that is called by other programs and subprograms. [ISO/IEC/IEEE 24765]

Other Definitions

Risk [ISO/IEC/IEEE 24765]: A function or procedure invocable for a single purpose.

Risk [ISO/IEC 2382]: A program, or part of a program, that may have some general or frequent use.

Notes

- The terms 'routine,' 'subprogram,' and 'subroutine' are defined and used differently in different programming languages. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Function

Standards:

- ISO/IEC 2382
- ISO/IEC/IEEE 24765

6.329. Run

In software engineering, a single, usually continuous, execution of a computer program. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.330. Safety

The capability of the software product to achieve acceptable levels of risk of harm to people, business, software, property or the environment in a specified context of use. [ISO/IEC 9126-1]

Other Definitions

Safety [ISO/IEC/IEEE 24765, ISO/IEC 15026]: The expectation that a system does not, under defined conditions, lead to a state in which human life, health, property, or the environment is endangered.

Notes

- Risks are usually a result of deficiencies in the functionality (including security), reliability, usability or maintainability. [ISO/IEC 9126-1]

See also

Standards:

- ISO/IEC 9126-1
- ISO/IEC/IEEE 24765
- ISO/IEC 15026

6.331. Satisfaction

The capability of the software product to satisfy users in a specified context of use. [ISO/IEC 9126-1]

Notes

- Satisfaction is the user's response to interaction with the product, and includes attitudes towards use of the product. [ISO/IEC 9126-1]

See also

- ISO/IEC 9126-1

6.332. Scale

Ordered set of values, continuous or discrete, or a set of categories to which the attribute is mapped. [ISO/IEC 99, ISO/IEC 15939, ISO/IEC 25000]

Other Definitions

Scale [ISO/IEC/IEEE 24765]: A set of values with defined properties.

Notes

- The type of scale depends on the nature of the relationship between values on the scale. Four types of scales are commonly defined:

:: Nominal: The measurement values are categorical. For example, the classification of defects by their type does not imply order among the categories.

; Ordinal: The measurement values are rankings. For example, the assignment of defects to a severity level is a ranking.

; Interval: The measurement values have equal distances corresponding to equal quantities of the attribute. For example, cyclomatic complexity has the minimum value of one, but each increment represents an additional path. The value of zero is not possible.

; Ratio: The measurement values have equal distances corresponding to equal quantities of the attribute where the value of zero corresponds to none of the attribute. For example, the size of a software component in terms of LOC is a ratio scale because the value of zero corresponds to no lines of code and each additional increments represents equal amounts of code.

: These are just examples of the types of scales. Roberts⁸ defines more types of scales. [ISO/IEC 15939]

→ The type of scale depends on the nature of the relationship between values on the scale. Metrics using nominal or ordinal scales produce qualitative data, and metrics using interval and ratio scales produce quantitative data. [ISO/IEC/IEEE 24765]

Example

→ A nominal scale which corresponds to a set of categories; an ordinal scale which corresponds to an ordered set of scale points; an interval scale which corresponds to an ordered scale with equidistant scale points; and a ratio scale which not only has equidistant scale point but also possess an absolute zero. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC 99
- ISO/IEC 15939
- ISO/IEC/IEEE 24765

6.333. Security

The capability of the software product to protect information and data so that unauthorised persons or systems cannot read or modify them and authorised persons or systems are not denied access to them. [ISO/IEC 12207, ISO/IEC 9126-1]

Other Definitions

Security [ISO/IEC 15026]: The protection of system items from accidental or malicious access, use, modification, destruction, or disclosure.

Security [ISO/IEC 15288]: All aspects related to defining, achieving, and maintaining confidentiality, integrity, availability, non-repudiation, accountability, authenticity, and reliability of a system.

Notes

- This also applies to data in transmission. [ISO/IEC 9126-1]
- Safety is defined as a characteristic of quality in use, as it does not relate to software alone, but to a whole system. [ISO/IEC 9126-1]

⁸F. Roberts. "Measurement Theory with Applications to Decision Making, Utility, and the Social Sciences". Addison-Wesley, 1979

→ Security also pertains to personnel, data, communications, and the physical protection of computer installations. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Safety

Standards:

- ISO/IEC 12207
- ISO/IEC 15026
- ISO/IEC 15288
- ISO/IEC 9126-1
- ISO/IEC/IEEE 24765

6.334. Service

Performance of activities, work, or duties associated with a product. [ISO/IEC 12207, ISO/IEC 15939]

Other Definitions

Software Service [ISO/IEC 12207, ISO/IEC 15939]: Performance of activities, work, or duties connected with a software product, such as its development, maintenance, and operation.

See also

Standards:

- ISO/IEC 12207
- ISO/IEC 15939

6.335. Service Level Agreement

Written agreement between a service provider and a customer that documents services and agreed service levels. [ISO/IEC 20000]

See also

Standards:

- ISO/IEC 20000

6.336. Simplicity

The degree to which a system or component has a design and implementation that is straightforward and easy to understand. [ISO/IEC/IEEE 24765]

Other Definitions

Simplicity [ISO/IEC/IEEE 24765]: Software attributes that provide implementation of functions in the most understandable manner.

See also

Glossary:

→ Complexity

Standards:

→ ISO/IEC/IEEE 24765

6.337. Software

All or part of the programs, procedures, rules, and associated documentation of an information processing system. [ISO/IEC 2382, ISO/IEC 9126-1]

Other Definitions

Software [IEEE 829]: Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.

Software [ISO/IEC 26514]: Program or set of programs used to run a computer.

Example

→ Command files, job control language. [ISO/IEC/IEEE 24765]

Notes

→ Software is an intellectual creation that is independent of the medium on which it is recorded. [ISO/IEC 9126-1]

→ Includes firmware, documentation, data, and execution control statements. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Software Product

Standards:

→ IEEE 829

→ ISO/IEC 2382

→ ISO/IEC 9126

6.338. Software Asset Management

Effective management, control and protection of software assets within an organization. [ISO/IEC 19770]

See also

Standards:

→ ISO/IEC 19770

6.339. Software Development Process

The process by which user needs are translated into a software product. [ISO/IEC/IEEE 24765]

Notes

- The process involves translating user needs into software requirements, transforming the software requirements into design, implementing the design in code, testing the code, and sometimes, installing and checking out the software for operational use. These activities may overlap or be performed iteratively. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Software Life Cycle

Standards:

- ISO/IEC/IEEE 24765

6.340. Software Engineering

The systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software. [ISO/IEC 2382]

Other Definitions

Software Engineering [ISO/IEC/IEEE 24765]: the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

See also

Standards:

- ISO/IEC 2382
- ISO/IEC/IEEE 24765

6.341. Software Item

Identifiable part of a software product. [ISO/IEC 90003]

Other Definitions

Software Item [ISO/IEC/IEEE 24765]: An aggregation of software, such as a computer program or database, that satisfies an end use function and is designated for specification, qualification testing, interfacing, configuration management, or other purposes.

Software Item [ISO/IEC 12207]: Source code, object code, control code, control data, or a collection of these items.

See also

Glossary:

→ Software Configuration Item

Standards:

→ ISO/IEC 12207

→ ISO/IEC 90003

→ ISO/IEC/IEEE 24765

6.342. Software Life Cycle

The period of time that begins when a software product is conceived and ends when the software is no longer available for use. [ISO/IEC/IEEE 24765]

Other Definitions

Software Life Cycle [IEEE 1074]: The project-specific sequence of activities that is created by mapping the activities of this standard onto a

selected software life cycle model (SLCM).

Software Life Cycle [IEEE 1362]: The system or product cycle initiated by a user need or a perceived customer need and terminated by discontinued use of the product.

See also

Glossary:

→ Software Development Process

Standards:

→ IEEE 1074

→ IEEE 1362

→ ISO/IEC/IEEE 24765

6.343. Software Product Evaluation

Technical operation that consists of producing an assessment of one or more characteristics of a software product according to a specified procedure. [ISO/IEC 25000]

Notes

→ This definition can be compared to that of testing in ISO/IEC Guide 2. However, the term evaluation is preferred in order to avoid confusion with the notion of testing widely accepted in the field of software engineering. Software product evaluation is not necessarily conformity testing (as defined in ISO/IEC Guide 2) in the context of a certification scheme. However, conformity testing can be part of an evaluation. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.344. Software Quality

Capability of a software product to satisfy stated and implied needs when used under specified conditions. [ISO/IEC 25000]

Notes

- This definition differs from the ISO 9000:2000 quality definition mainly because the software quality definition refers to the satisfaction of stated and implied needs, while the ISO 9000 quality definition refers to the satisfaction of requirements. In SQuaRE standards software quality has the same meaning as software product quality. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Software Quality Characteristic
- Software Quality Evaluation

Standards:

- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.345. Software Quality Characteristic

Category of software quality attributes that bears on software quality. [ISO/IEC 25000]

Notes

- Software quality characteristics may be refined into multiple levels of sub-characteristics and finally into software quality attributes. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Software Quality

Standards:

- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.346. Software Quality Evaluation

Systematic examination of the extent to which a software product is capable of satisfying stated and implied needs. [ISO/IEC 25000]

See also

Glossary:

- Software Product Evaluation
- Software Quality

Standards:

- ISO/IEC 25000

6.347. Software Quality Measure

Measure of internal software quality, external software quality or software quality in use. IEEE 1490]

Notes

- Internal software quality, external software quality and software quality in use are described in the quality model in ISO/IEC 9126-1 [ISO/IEC 25010, ISO/IEC/IEEE 24765]

See also

Glossary:

- External Quality
- Internal Quality
- Measure
- Quality in Use

Standards:

- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.348. Software Repository

A software library providing permanent, archival storage for software and related documentation. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.349. Software Unit

Separately compilable piece of code. [ISO/IEC 12207]

Other Definitions

Software Unit [ISO/IEC 12207]: The lowest element in one or more software components.

See also

Standards:

→ ISO/IEC 12207

6.350. Source Code

Computer instructions and data definitions expressed in a form suitable for input to an assembler, compiler, or other translator. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Product

Standards:

→ ISO/IEC/IEEE 24765

6.351. Specification

A document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system, component, product, result, or service and, often, the procedures for determining whether these provisions have been satisfied. [IEEE 1490]

Other Definitions

Specification [ISO/IEC 2382]: A detailed formulation, in document form, which provides a definitive description of a system for the purpose of developing or validating the system.

Specification [IEEE 1220]: A document that fully describes a design element or its interfaces in terms of requirements (functional, performance, constraints, and design characteristics) and the qualification conditions and procedures for each requirement.

See also

Glossary:

→ Requirement

Standards:

→ IEEE 1220

→ IEEE 1490

→ ISO/IEC 2382

6.352. Stability

The capability of the software product to avoid unexpected effects from modifications of the software. [ISO/IEC 9126-1]

See also

→ ISO/IEC 9126-1

6.353. Stage

Period within the life cycle of an entity that relates to the state of its description or realization. [ISO/IEC 12207, ISO/IEC 15288]

Notes

→ Stages relate to major progress and achievement milestones of the system through its life cycle. Stages may be overlapping. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC 12207

→ ISO/IEC 15288

6.354. Stakeholder

Individual or organisation that sponsors measurement, provides data, is a user of the measurement results or otherwise participates in the measurement process. [ISO/IEC 15939]

Other Definitions

Stakeholder [ISO/IEC 12207, ISO/IEC 15288, ISO/IEC 15939]: Individual or organization having a right, share, claim, or interest in a system or in its possession of characteristics that meet their needs and expectations.

Stakeholder [IEEE 1490]: Person or organization (e.g. customer, sponsor, performing organization, or the public) that is actively involved in the project, or whose

interests may be positively or negatively affected by execution or completion of the project. A stakeholder may also exert influence over the project and its deliverables.

Examples

→ End users, end user organizations, supporters, developers, producers, trainers, maintainers, disposers, acquirers, supplier organizations and regulatory bodies. [ISO/IEC/IEEE 24765]

Notes

→ The decision-maker is also a stakeholder. [ISO/IEC/IEEE 24765]

See also

Standards:

- IEEE 1490
- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC 15939

6.355. Standard

Set of mandatory requirements established by consensus and maintained by a recognized body to prescribe a disciplined uniform approach or specify a product, that is, mandatory conventions and practices. [ISO/IEC/IEEE 24765]

Other Definitions

Standard [IEEE 1490]: A document that provides, for common and repeated use, rules, guidelines or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context.

See also

Standards:

- IEEE 1490
- ISO/IEC/IEEE 24765

6.356. Standard Process

The set of definitions of the basic processes that guide all processes in an organization. [ISO/IEC 15504]

Notes

- These process definitions cover the fundamental process elements (and their relationships to each other) that must be incorporated into the defined processes that are implemented in projects across the organization. A standard process establishes consistent activities across the organization and is desirable for long-term stability and improvement. The organization's set of standard processes describes the fundamental process elements that will be part of the projects' defined processes. It also describes the relationships (for example, ordering and interfaces) between these process elements. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Process

Standards:

- ISO/IEC 15504
- ISO/IEC/IEEE 24765

6.357. Statement

In a programming language, a meaningful expression that defines data, specifies program actions, or directs the assembler or compiler. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Statement Testing

Standards:

→ ISO/IEC/IEEE 24765

6.358. Statement Testing

Testing designed to execute each statement of a computer program. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Branch Testing

→ Path Testing

→ Statement

→ Testing

Standards:

→ ISO/IEC/IEEE 24765

6.359. Statement of Work

Document used by the acquirer to describe and specify the tasks to be performed under the contract. [ISO/IEC 12207]

Other Definitions

Statement of Work [IEEE 1490]: A narrative description of products, services, or results to be supplied.

See also

Standards:

→ IEEE 1490

→ ISO/IEC 12207

6.360. Static Analysis

The process of evaluating a system or component based on its form, structure, content, or documentation. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Dynamic Analysis

Standards:

→ ISO/IEC/IEEE 24765

6.361. Statistical Process Control

Statistically based analysis of a process and measures of process performance, which identify common and special causes of variation in process performance and maintain process performance within limits. [ISO/IEC/IEEE 24765]

Notes

""Statistical Process Control"" is an effective method of monitoring a process through the use of control charts. In general, if a process exceeds the limits, we assume that it's out of control and the project team should search for special causes to deal with it. There are many kinds of charts, such as the \bar{x} chart and r-chart, etc.

=== The c-chart ===

The c-chart plots the number of defects in a process. If C_i denotes the number of defects obtained in the i th observation, the c-chart plots the data points at the height $C_1, C_2 \dots C_n$. The c-chart also has a center line (CL) at height \bar{C} (the average of C_i) and the following 3σ lines:

Upper Control Limit: $UCL = \bar{C} + 3\sqrt{\bar{C}}$

Lower Control Limit: $LCL = \bar{C} - 3\sqrt{\bar{C}}$

If LCL is negative, it is set to zero. The c-chart assumes the Poisson distribution of defects and is thus approximative.

Use of SPC in software engineering is still under debate. One major issue is that formal SPC requires data to be independent variables from homogeneous sources of variation. As exposed in Software Engineering Metrics: What Do They Measure And How Do We Know [https://maisqual.squoring.com/wiki/index.php/Software_Engineering_Metrics:_What_Do_They_Measure_And_How_Do_We_Know], software engineering data is often affected by many variations sources. Furthermore, software engineering is domain-specific (requirements may vary from one domain to another) and limits may vary.

See also

Glossary:

→ Process

→ Process Performance

Papers:

→ Monitoring Software Quality Evolution for Defects [https://maisqual.squoring.com/wiki/index.php/Monitoring_Software_Quality_Evolution_for_Defects]

→ Software Engineering Metrics: What Do They Measure And How Do We Know [https://maisqual.squoring.com/wiki/index.php/Software_Engineering_Metrics:_What_Do_They_Measure_And_How_Do_We_Know]

Standards:

→ ISO/IEC/IEEE 24765

6.362. Step

One element (numbered list item) in a procedure that tells a user to perform an action (or actions). [ISO/IEC 26514]

Other Definitions

Step [ISO/IEC 15414]: An abstraction of an action, used in a process, that may leave unspecified objects that participate in that action.

Notes

→ A step contains one or more actions. Responses by the software are not considered to be steps. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Action
- Procedure
- Process

Standards:

- ISO/IEC 15414
- ISO/IEC 26514
- ISO/IEC/IEEE 24765

6.363. Stress Testing

Testing conducted to evaluate a system or component at or beyond the limits of its specified requirements. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Testing

Standards:

- ISO/IEC/IEEE 24765

6.364. Structural Testing

Testing that takes into account the internal mechanism of a system or component. Syn: glass-box testing, white-box testing. [ISO/IEC/IEEE 24765]

Notes

- Types include branch testing, path testing, statement testing. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Functional Testing
- Testing

Standards:

- ISO/IEC/IEEE 24765

6.365. Stub

A skeletal or special-purpose implementation of a software module, used to develop or test a module that calls or is otherwise dependent on it. [ISO/IEC/IEEE 24765]

Other Definitions

Stub [ISO/IEC/IEEE 24765]: A computer program statement substituting for the body of a software module that is or will be defined elsewhere.

Stub [ISO/IEC/IEEE 24765]: Scaffolding code written for the purpose of exercising higher-level code before the lower-level routines that will ultimately be used are available.

See also

Glossary:

- Mock Object
- Testing

Standards:

- IEEE 1362
- ISO/IEC 9126
- ISO/IEC 12207
- ISO/IEC 15939

6.366. Suitability

The capability of the software product to provide an appropriate set of functions for specified tasks and user objectives. [ISO/IEC 9126-1]

Notes

- Examples of appropriateness are task-oriented composition of functions from constituent sub-functions, and capacities of tables. [ISO/IEC 9126-1]
- Suitability corresponds to "suitability for the task" in ISO 9241-10. [ISO/IEC 9126-1]
- Suitability also affects operability. [ISO/IEC 9126-1]

See also

Standards:

- ISO/IEC 9126-1
- ISO 9241-10

6.367. Supplier

Organisation that enters into an agreement with the acquirer for the supply of a system, software product or software service under the terms of that agreement. [ISO/IEC 9126, ISO/IEC 12207, ISO/IEC 15939]

Notes

- The term "supplier" is synonymous with "contractor", "producer", "seller", or "vendor". [ISO/IEC 15939]
- The acquirer may designate a part of its organisation as supplier. [ISO/IEC 15939]

See also

Glossary:

- Acquirer

Standards:

- ISO/IEC 9126
- ISO/IEC 12207
- ISO/IEC 15939

6.368. Support

The set of activities necessary to ensure that an operational system or component fulfills its original requirements and any subsequent modifications to those requirements. [ISO/IEC/IEEE 24765]

Other Definitions

Software Support [ISO 9127]: The act of maintaining the software and its associated documentation in a functional state.

Examples

- Software or hardware maintenance, user training. [ISO/IEC/IEEE 24765]

Notes

- Software support may be given by the manufacturer, marketing organization, supplier or other organization. In special contractually-agreed cases, consumers may be permitted to maintain or enhance the software themselves. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO 9127

→ ISO/IEC/IEEE 24765

6.369. Support Manual

A document that provides the information necessary to service and maintain an operational system or component throughout its life cycle. [ISO/IEC/IEEE 24765]

Notes

→ Support Manual is synonym for Maintenance Manual. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Document
- Installation Manual
- Maintenance Manual
- Operator Manual
- User Manual

Standards:

- ISO/IEC/IEEE 24765

6.370. System

Integrated composite that consists of one or more of the processes, hardware, software, facilities and people, that provides a capability to satisfy a stated need or objective. [ISO/IEC 9126, ISO/IEC 12207, ISO/IEC 15939]

Other Definitions

Software System [IEEE 1362]: A software-intensive system for which software is the only component to be developed or modified.

See also

Standards:

- IEEE 1362
- ISO/IEC 9126
- ISO/IEC 12207
- ISO/IEC 15939

6.371. System Testing

Testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. [IEEE 829]

See also

Glossary:

- Integration Test
- Testing
- Unit Test

Standards:

- IEEE 829

6.372. Task

The activities required to achieve a goal. [ISO/IEC 9126]

Other Definitions

Task [ISO/IEC 12207, ISO/IEC 15288]: Required, recommended, or permissible action, intended to contribute to the achievement of one or more outcomes of a process.

Task [ISO/IEC/IEEE 24765]: In software design, a [[Software Component|software component that can operate in parallel with other software components.

Task [ISO/IEC/IEEE 24765]: A concurrent object with its own thread of control.

Task [ISO/IEC/IEEE 24765]: A sequence of instructions treated as a basic unit of work by the supervisory program of an operating system.

Task [IEEE 829]: Smallest unit of work subject to management accountability; a well-defined work assignment for one or more project members.

Notes

- Related tasks are usually grouped to form activities. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Activity
- Procedure
- Process

Standards:

- IEEE 829
- ISO/IEC 9126
- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC/IEEE 24765

6.373. Technical Requirement

Requirements relating to the technology and environment, for the development, maintenance, support and execution of the software. [ISO/IEC/IEEE 24765]

Examples

- Programming language, testing tools, operating systems, database technology and user interface technologies. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Nontechnical Requirement
- Requirement

Standards:

- ISO/IEC/IEEE 24765

6.374. Technique

Methods and skills required to carry out a specific activity. [ISO/IEC 25001, ISO/IEC 12207, ISO/IEC 15939]

Other Definitions

Technique [ISO/IEC/IEEE 24765]: Technical or managerial procedure that aids in the evaluation and improvement of the software development process.

Technique [IEEE 1490]: A defined systematic procedure employed by a human resource to perform an activity to produce a product or result or deliver a service, and that may employ one or more tools.

See also

Standards:

- IEEE 1490
- ISO/IEC 25001
- ISO/IEC/IEEE 24765

6.375. Test

An activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component. [ISO/IEC/IEEE 24765]

Other Definitions

Test [IEEE 829]: A set of one or more test cases and procedures.

See also

Glossary:

- Test Case
- Testing

Standards:

- IEEE 1362
- ISO/IEC 9126
- ISO/IEC 12207
- ISO/IEC 15939

6.376. Test Case

A set of inputs, execution preconditions, and expected outcomes developed for a particular objective to exercise a particular program path or to verify compliance with a specific requirement. [IEEE 1012, SIGIST]

Other Definitions

Test Case [IEEE 610.12]: A documented instruction for the tester that specifies how a function or a combination of functions shall or should be tested. A test case includes detailed information on the following issues:

- the test objective;
- the functions to be tested;
- the testing environment and other conditions;
- the test data;
- the procedure;
- the expected behaviour of the system.

See also

Glossary:

- Requirement
- Test Case Suite
- Testing

Standards:

- IEEE 1012
- IEEE 610.12
- SIGIST

6.377. Test Case Suite

A collection of one or more test cases for the software under test. [SIGIST]

See also

- SIGIST

6.378. Test Coverage

Extent to which the test cases test the requirements for the system or software product. [ISO/IEC 12207]

Other Definitions

Test Coverage [ISO/IEC/IEEE 24765]: The degree to which a given test or set of tests addresses all specified requirements for a given system or component.

See also

Glossary:

- Branch Coverage
- Code Coverage
- Requirement
- Test
- Testing
- Test Case

Standards:

- ISO/IEC 12207
- ISO/IEC/IEEE 24765

6.379. Test Documentation

Collection of the documentation inherent to the testing activities. [ISO/IEC 25051]

Other Definitions

Test Documentation [ISO/IEC/IEEE 24765]: Documentation describing plans for, or results of, the testing of a system or component.

See also

Glossary:

- Documentation
- Testing

Standards:

- ISO/IEC 25051
- ISO/IEC/IEEE 24765

6.380. Test Environment

Hardware and software configuration necessary to conduct the test case. [ISO/IEC 25051]

See also

Glossary:

- Test Case

→ Testing

Standards:

→ ISO/IEC 25051

6.381. Test Objective

Identified set of software features to be measured under specified conditions by comparing actual behavior with the required behavior. [ISO/IEC 25051, ISO/IEC 25062]

See also

Glossary:

→ Testing

Standards:

→ ISO/IEC 25051

→ ISO/IEC 25062

6.382. Test Plan

A document describing the scope, approach, resources, and schedule of intended test activities. [IEEE 1012, ISO/IEC 12207, ISO/IEC 15939]

Other Definitions

Test Plan [IEEE 1012]: A document that describes the technical and management approach to be followed for testing a system or component.

Test Plan [ISO/IEC 2382]: A plan that establishes detailed requirements, criteria, general methodology, responsibilities, and general planning for test and evaluation of a system.

Notes

→ It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning. Typical contents identify the items to be tested, tasks to be performed, responsibilities, schedules, and required resources for the testing activity. [ISO/IEC/IEEE 24765]

See also

Glossary:

→ Testing

Standards:

→ IEEE 1012

→ ISO/IEC 2382

6.383. Test Procedure

Detailed instructions for the setup, execution, and evaluation of results for a given test case. [IEEE 1012]

Other Definitions

Test Procedure [IEEE 1012]: Documentation that specifies a sequence of actions for the execution of a test.

See also

Glossary:

→ Testing

Standards:

→ IEEE 1012

6.384. Testability

The capability of the software product to enable modified software to be validated. [ISO/IEC 9126-1]

Other Definitions

Testability [ISO/IEC 12207]: Extent to which an objective and feasible test can be designed to determine whether a requirement is met.

Testability [IEEE 1233]: The degree to which a requirement is stated in terms that permit establishment of test criteria and performance of tests to determine whether those criteria have been met.

Testability [ISO/IEC/IEEE 24765]:

1. The degree to which a system can be unit tested and system tested.
2. The effort required to test software.
3. The degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met.

See also

Glossary:

→ Branch Coverage

→ Code Coverage

→ Test Coverage

→ Testing

Standards:

→ IEEE 1233

→ ISO/IEC 9126-1

→ ISO/IEC 12207

→ ISO/IEC/IEEE 24765

6.385. Testing

Activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component. [IEEE 829]

Other Definitions

Software Testing [ISO/IEC 19759]: The dynamic verification of the behavior of a program on a finite set of test cases, suitably selected from the usually infinite executions domain, against the expected behavior.

See also

Glossary:

- Acceptance Testing
- Branch Testing
- Development Testing
- Interface Testing
- Interoperability Testing
- Functional Testing
- Operational Testing
- Path Testing
- Performance Testing
- Regression Testing
- Qualification Testing
- Statement Testing
- Stress Testing
- Structural Testing
- System Testing
- Testing Description

Standards:

- IEEE 829
- ISO/IEC 19759

6.386. Testing Description

Description of the test execution conditions (i.e. test procedure). [ISO/IEC 25051, ISO/IEC 25062]

See also

Glossary:

- Testing

Standards:

- ISO/IEC 25051
- ISO/IEC 25062

6.387. Time Behaviour

The capability of the software product to provide appropriate response and processing times and throughput rates when performing its function, under stated conditions. [ISO/IEC 9126-1]

See also

→ ISO/IEC 9126-1

6.388. Tool

A software product that provides support for software and system life cycle processes. [ISO/IEC 15474]

Other Definitions

Tool [IEEE 1490]: Something tangible, such as a template or software program, used in performing an activity to produce a product or result.

Notes

→ Particularly, but not exclusively, a modeling tool. Also, tool is used as a short form for software tool, and more specifically for CASE tool. [ISO/IEC/IEEE 24765]

See also

Standards:

- IEEE 1490
- ISO/IEC 15474
- ISO/IEC/IEEE 24765

6.389. Total Quality Management

A holistic approach to quality improvement in all life-cycle phases. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC/IEEE 24765

6.390. Traceability

The degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another. [IEEE 1233]

Other Definitions

Traceability [IEEE 1362]: The identification and documentation of derivation paths (upward) and allocation or flowdown paths (downward) of work products in the work product hierarchy.

Traceability [ISO/IEC/IEEE 24765]: The degree to which each element in a software development product establishes its reason for existing.

Traceability [ISO/IEC/IEEE 24765]: Discernible association among two or more logical entities, such as requirements, system elements, verifications, or tasks.

Notes

- The degree to which the requirements and design of a given system element match; the degree to which each element in a bubble chart references the requirement that it satisfies. [ISO/IEC/IEEE 24765]

See also

Standards:

- IEEE 1233
- IEEE 1362
- ISO/IEC/IEEE 24765

6.391. Traceable

Having components whose origin can be determined. [ISO/IEC 12207]

See also

Glossary:

- Traceability

Standards:

- ISO/IEC 12207

6.392. Trunk

The software's main line of development; the main starting point of most branches. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Branch
- Configuration Management

Standards:

- ISO/IEC/IEEE 24765

6.393. Understandability

The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use. [ISO/IEC 9126-1]

Other Definitions

Understandability [ISO/IEC/IEEE 24765]: The ease with which a system can be comprehended at both the system-organizational and detailed-statement levels.

Notes

- This will depend on the documentation and initial impressions given by the software. [ISO/IEC 9126-1]
- Understandability has to do with the system's coherence at a more general level than readability does. [ISO/IEC/IEEE 24765]

See also

Standards:

- ISO/IEC 9126-1
- ISO/IEC/IEEE 24765

6.394. Unit Test

Testing of individual routines and modules by the developer or an independent tester. [ISO/IEC/IEEE 24765]

Other Definitions

Unit Test [ISO/IEC 2382]: A test of individual programs or modules in order to ensure that there are no analysis or programming errors.

Unit Test [ISO/IEC/IEEE 24765]: A test of individual hardware or software units or groups of related units.

See also

Glossary:

- Test
- Testing

Standards:

- ISO/IEC 2382
- ISO/IEC/IEEE 24765

6.395. Unit of Measurement

Particular quantity, defined and adopted by convention, with which other quantities of the same kind are compared in order to express their magnitude relative to that quantity. [ISO/IEC 99, ISO/IEC 15939, ISO/IEC 25000]

Notes

- Units of measurement have conventionally assigned names and symbols. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Measurement

Standards:

- ISO/IEC 99
- ISO/IEC 15939
- ISO/IEC 25000
- ISO/IEC/IEEE 24765

6.396. Usability

The capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions. [ISO/IEC 9126-1]

Other Definitions

Usability [ISO/IEC/IEEE 24765]: The ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component.

Usability [ISO/IEC 25062]: The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

Notes

- Some aspects of functionality, reliability and efficiency will also affect usability, but for the purposes of ISO/IEC 9126 they are not classified as usability. [ISO/IEC 9126-1]
- Users may include operators, end users and indirect users who are under the influence of or dependent on the use of the software. Usability should address all of the different user environments that the software may affect, which may include preparation for usage and evaluation of results. [ISO/IEC 9126-1]
- This term has been deliberately redefined to more properly convey its meaning in the software reuse context. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Understandability

Standards:

- ISO/IEC 9126
- ISO/IEC 9126-1
- ISO/IEC 25062
- ISO/IEC/IEEE 24765

6.397. Usability Compliance

The capability of the software product to adhere to standards, conventions, style guides or regulations relating to usability. [ISO/IEC 9126-1]

See also

- ISO/IEC 9126-1

6.398. User

Individual or organisation that uses the system to perform a specific function. [ISO/IEC 12207, ISO/IEC 15939]

Other Definitions

User [ISO/IEC 9126]: An individual that uses the software product to perform a specific function.

User [ISO/IEC 26514]: Person who performs one or more tasks with software; a member of a specific audience.

User [ISO/IEC 25062]: Person who interacts with the product.

User [IEEE 1362]: Individual or organization who uses a software-intensive system in daily work activities or recreational pursuits.

User [ISO/IEC 15288, ISO/IEC 15939]: Individual or group that benefits from a system during its utilization.

User [ISO/IEC 14143, ISO/IEC 29881]: Any person or thing that communicates or interacts with the software at any time.

Notes

- Users may include operators, recipients of the results of the software, or developers or maintainers of software. [ISO/IEC 9126-1]
- The user may perform other roles such as acquirer or maintainer. The role of user and the role of operator may be vested, simultaneously or sequentially, in the same individual or organization. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Developer

Standards:

- IEEE 1362
- ISO/IEC 9126
- ISO/IEC 12207
- ISO/IEC 14143
- ISO/IEC 15288
- ISO/IEC 15939
- ISO/IEC 25062
- ISO/IEC 26514
- ISO/IEC 29881
- ISO/IEC/IEEE 24765

6.399. User Documentation

Documentation for users of a system, including a system description and procedures for using the system to obtain desired results. [ISO/IEC/IEEE 24765]

Other Definitions

User Documentation [ISO/IEC 26514]: Information to describe, explain, or instruct how to use software.

See also

Glossary:

- Documentation
- User

Standards:

- ISO/IEC 26514
- ISO/IEC/IEEE 24765

6.400. User Manual

A document that presents the information necessary to employ a system or component to obtain desired results. [ISO/IEC/IEEE 24765]

Other Definitions

User Manual [ISO/IEC 2382]: A document that describes how to use a functional unit, and that may include description of the rights and responsibilities of the user, the owner, and the supplier of the unit.

Notes

- Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions. A user manual is distinguished from an operator manual when a distinction is made between those who operate a computer system (mounting tapes, etc.) and those who use the system for its intended purpose. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Document
- Installation Manual
- Maintenance Manual
- Operator Manual
- Support Manual
- User Documentation

Standards:

- ISO/IEC 2382
- ISO/IEC/IEEE 24765

6.401. Validation

Determination of the correctness of the products of software development with respect to the user needs and requirements. [SIGIST]

Other Definitions

Validation [ISO 8402, ISO/IEC 9126-1]: Confirmation by examination and provision of objective evidence that the particular requirements for a specific intended use are fulfilled.

Validation [ISO/IEC 15288]: Confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled.

Validation [IEEE 1012]: The process of providing evidence that the software and its associated products satisfy system requirements allocated to software at the end of each life cycle activity, solve the right problem, and satisfy intended use and user needs.

Validation [ISO/IEC 12207]: In a life cycle context, the set of activities ensuring and gaining confidence that a system is able to accomplish its intended use, goals and objectives.

Validation [IEEE 1233]: The process of evaluating a system or component during or at the end of the development process to determine whether a system or component satisfies specified requirements.

Validation [IEEE 1490]: The assurance that a product, service, or system meets the needs of the customer and other identified stakeholders. It often involves acceptance and suitability with external customers.

Notes

- In design and development, validation concerns the process of examining a product to determine conformity with user needs. [ISO/IEC 9126-1]
- Validation is normally performed on the final product under defined operating conditions. It may be necessary in earlier stages. [ISO/IEC 9126-1]
- "Validated" is used to designate the corresponding status. [ISO/IEC 9126-1]
- Multiple validations may be carried out if there are different intended uses. [ISO/IEC 9126-1]
- Validation demonstrates that the system can be used by the users for their specific tasks. "Validated" is used to designate the corresponding status. [ISO 9000:2005] In design and development, validation concerns the process of examining a product to determine conformity with user needs. Validation is normally performed on the final product under defined operating conditions. It may be necessary in earlier stages. Multiple validations may be carried out if there are different intended uses. [ISO/IEC/IEEE 24765]

See also

Glossary:

- Verification

Standards:

- IEEE 1012
- IEEE 1233
- IEEE 1490
- ISO 8402
- ISO/IEC 9126
- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC/IEEE 24765
- SIGIST

6.402. Value

Number or category assigned to an attribute of an entity by making a measurement. [ISO/IEC 25000]

Other Definitions

Value [ISO/IEC 15939]: Numerical or categorical result assigned to a base measure, derived measure, or indicator. [ISO/IEC 15939]

See also

Glossary:

- Base Measure
- Derived Measure
- Indicator
- Measurement

Standards:

- ISO/IEC 15939
- ISO/IEC 25000

6.403. Verification

Confirmation, through the provision of objective evidence, that specified requirements have been fulfilled. [ISO/IEC 12207, ISO/IEC 15288, ISO/IEC 25000]

Other Definitions

Verification [IEEE 1012, SIGIST]: The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.

Verification [ISO 8402, ISO/IEC 9126]: Confirmation by examination and provision of objective evidence that specified requirements have been fulfilled.

Verification [ISO/IEC/IEEE 24765]: Formal proof of program correctness.

Verification [IEEE 1490]: The evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition. It is often an internal process.

Verification [IEEE 829]: Process of providing objective evidence that the software and its associated products comply with requirements (e.g., for correctness, completeness, consistency, and accuracy) for all life cycle activities during each life cycle process (acquisition, supply, development, operation, and maintenance), satisfy standards, practices, and conventions during life cycle processes, and successfully complete each life cycle activity and satisfy all the criteria for initiating succeeding life cycle activities (e.g., building the software correctly).

Notes

- "Verified" is used to designate the corresponding status. In design and development, verification concerns the process of examining the result of a given activity to determine conformity with the stated requirement for that activity. [ISO/IEC 9126]

See also

Glossary:

→ Validation

Standards:

→ IEEE 829

→ IEEE 1012

→ IEEE 1490

→ ISO 8402

→ ISO/IEC 9126

→ ISO/IEC 12207

→ ISO/IEC 15288

→ ISO/IEC 25000

→ ISO/IEC/IEEE 24765

→ SIGIST

6.404. Version

Identified instance of an item. [ISO/IEC 12207]

Other Definitions

Version [ISO/IEC/IEEE 24765]: An initial release or re- release of a computer software configuration item, associated with a complete compilation or recompilation of the computer software configuration item.

Version [ISO/IEC/IEEE 24765]: An initial release or complete re- release of a document, as opposed to a revision resulting from issuing change pages to a previous release.

Version [ISO/IEC/IEEE 24765]: An operational software product that differs from similar products in terms of capability, environmental requirements, and configuration.

Version [ISO/IEC/IEEE 24765]: An identifiable instance of a specific file or release of a complete system.

Notes

→ Modification to a version of a software product resulting in a new version requires configuration management action. [ISO/IEC/IEEE 24765]

See also

Standards:

→ ISO/IEC 12207

→ ISO/IEC/IEEE 24765

6.405. Work Breakdown Structure

A deliverable-oriented hierarchical decomposition of the work to be executed by the project team to accomplish the project objectives and create the required deliverables. It organizes and defines the total scope of the project. [IEEE 1490]

See also

Standards:

→ IEEE 1490

6.406. Work Product

An artifact associated with the execution of a process. [ISO/IEC 15504]

Other Definitions

Work Product [IEEE 1058]: A tangible item produced during the process of developing or modifying software.

See also

Glossary:

→ Product

Standards:

→ IEEE 1058

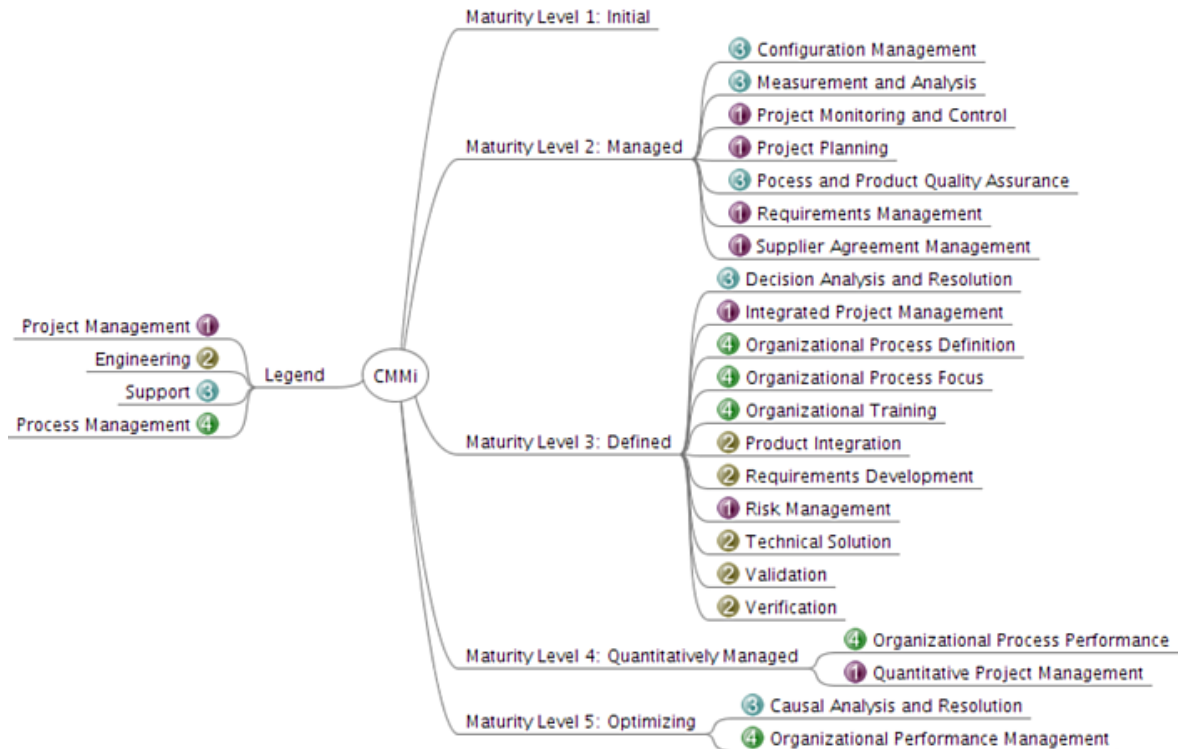
7. Standards

7.1. CMMi

CMMi stands for "Capability Maturity Model Integration".

CMMi is a process developed by the Carnegie Mellon Software Engineering Institute.

Structure



See also

Standards:

→ Team Software Process

External Links:

→ The official SEI CMMi web page: www.sei.cmu.edu/cmmi/ [<http://www.sei.cmu.edu/cmmi/>].

7.2. DOD-STD-2167A

Military Standard - Defense System Software Development

DOD-STD-2167A.

See also

→ The wikipedia article for the DOD-STD-2167: <http://en.wikipedia.org/wiki/DOD-STD-2167A>

7.3. IEC 61508

International Standard IEC 61508

Functional safety of electrical / electronic / programmable electronic safety related systems

Year: 1998, 2000, 2002, 2010

Contents

Part 1: General requirements

Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems

Part 3: Software Requirements

Part 4: Definitions and abbreviations

Part 5: Examples of methods for the determination of safety integrity levels

Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3

Part 7: Overview on techniques and measures

See also

→ IEC 61508-3.

→ IEC 61508-7.

7.4. IEC 61508-3

International Standard IEC 61508-3

Functional safety of electrical / electronic / programmable electronic safety related systems

Part 3: Software requirements

Year: 1998

See also

→ IEC 61508

→ IEC 61508-7

7.5. IEC 61508-7

International Standard IEC 61508-7

Functional safety of electrical / electronic / programmable electronic safety related systems

Part 7: Overview on techniques and measures

Year: 2000

See also

- IEC 61508
- IEC 61508-3

7.6. IEEE 1012

International Standard IEEE 1012

IEEE Standard for Software Verification and Validation

Year: 1986

This standard has been superseded.

Access

Online IEEE Catalog: <http://standards.ieee.org/findstds/standard/1012-1998.html>

7.7. IEEE 1058

International Standard IEEE 1058

IEEE Standard for Software Project Management Plans

Year: 1998

Access

Online IEEE Catalog: <http://standards.ieee.org/findstds/standard/1058-1998.html>

7.8. IEEE 1061

International Standard IEEE 1061

Standard for a Software Quality Metrics Methodology

Year: 1998

Access

Online IEEE Catalog: <http://standards.ieee.org/findstds/standard/1061-1998.html>

7.9. IEEE 1074

International Standard IEEE 1074

IEEE Standard for Developing Software Life Cycle Processes

Year: 1997

This standard has been superseded.

Access

Online IEEE Catalog: <http://standards.ieee.org/findstds/standard/1074-1997.html>

7.10. IEEE 1220

International Standard IEEE 1220-2005

1220-2005 - IEEE Standard for Application and Management of the Systems Engineering Process

Year: 2005

Access

Online IEEE Catalog: <http://standards.ieee.org/findstds/standard/1220-2005.html>

7.11. IEEE 1233

International Standard IEEE 1233

IEEE Guide for Developing System Requirements Specifications

Year: 1996

Access

Online IEEE Catalog: <http://standards.ieee.org/findstds/standard/1233-1996.html>

7.12. IEEE 1320

International Standard IEEE 1320.2

IEEE Standard for Conceptual Modeling Language - Syntax and Semantics for IDEF1X97 (IDEFobject)

Years: 1998

Access

Online IEEE Catalog:

→ <http://standards.ieee.org/findstds/standard/1320.2-1998.html>

7.13. IEEE 1362

International Standard IEEE 1362

IEEE Guide for Information Technology - System Definition - Concept of Operations (ConOps) Document

Year: 1998

Access

Online IEEE Catalog: <http://standards.ieee.org/findstds/standard/1362-1998.html>

7.14. IEEE 1490

International Standard IEEE 1490

IEEE Guide Adoption of PMI Standard - A Guide to the Project Management Body of Knowledge

Year: 2003

This standard has been withdrawn.

Access

Online IEEE Catalog: <http://standards.ieee.org/findstds/standard/1490-2003.html>

7.15. IEEE 610.12

International Standard IEEE 610.12

Standard Glossary of Software Engineering Terminology

Year: 1990

Access

Online IEEE Catalog: <http://standards.ieee.org/findstds/standard/610.12-1990.html>

7.16. IEEE 829

International Standard IEEE 829

IEEE Standard for Software Test Documentation

Year: 1983.

This standard has been superseded.

Access

Online IEEE catalog:

→ <http://standards.ieee.org/findstds/standard/829-1983.html>

7.17. IEEE 830

International Standard IEEE 830

IEEE Recommended Practice for Software Requirements Specifications

Year: 1998.

Access

Online IEEE catalog:

→ <http://standards.ieee.org/findstds/standard/830-1998.html>

7.18. IEEE 982

International Standard IEEE 982

IEEE Standard Dictionary of Measures to Produce Reliable Software

Year: 1988.

Access

Online IEEE catalog:

→ <http://standards.ieee.org/findstds/standard/982.1-1988.html>

7.19. ISO 5806

International Standard ISO 5806

Information processing -- Specification of single-hit decision tables

Year: 1984

Access

Online ISO Catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=11954

7.20. ISO 8402

Quality management and quality assurance - Vocabulary

Year: 1994

7.21. ISO 9001

International Standard ISO 9001.

Quality systems - Model for quality assurance in design, development, production, installation and servicing

Year: 1994, 2000, 2008.

Access

Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=46486

7.22. ISO 9127

International Standard ISO 9127

Information processing systems -- User documentation and cover information for consumer software packages

Year: 1988.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16723

7.23. ISO 9241

International Standard ISO 9241

Ergonomic requirements for office work with visual display terminals (VDTs)

Years: 1992-2011.

Contents / Access

The following parts link to the online ISO catalog:

- Part 1: General introduction [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=21922]
- Part 2: Guidance on task requirements [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16874]
- Part 4: Keyboard requirements [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16876]
- Part 5: Workstation layout and postural requirements [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16877]
- Part 6: Guidance on the work environment [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16878]
- Part 9: Requirements for non-keyboard input devices [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=30030]
- Part 11: Guidance on usability [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16883]
- Part 12: Presentation of information [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16884]
- Part 13: User guidance [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16885]
- Part 14: Menu dialogues [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16886]
- Part 15: Command dialogues [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16887]
- Part 16: Direct manipulation dialogues [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16888]
- Part 17: Form filling dialogues [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16889]
- Part 20: Accessibility guidelines for information/communication technology (ICT) equipment and services [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40727]
- Part 100: Introduction to standards related to software ergonomics [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=52712]

- Part 110: Dialogue principles [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38009]
- Part 129: Guidance on software individualization [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50014]
- Part 151: Guidance on World Wide Web user interfaces [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=37031]
- Part 171: Guidance on software accessibility [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39080]
- Part 210: Human-centred design for interactive systems [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=52075]
- Part 300: Introduction to electronic visual display requirements [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40096]
- Part 302: Terminology for electronic visual displays [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40097]
- Part 303: Requirements for electronic visual displays [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40098]
- Part 304: User performance test methods for electronic visual displays [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40099]
- Part 305: Optical laboratory test methods for electronic visual displays [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40100]
- Part 306: Field assessment methods for electronic visual displays [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40101]
- Part 307: Analysis and compliance test methods for electronic visual displays [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40102]
- Part 308: Surface-conduction electron-emitter displays (SED) [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=44843]
- Part 309: Organic light-emitting diode (OLED) displays [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51308]
- Part 310: Visibility, aesthetics and ergonomics of pixel defects [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=54117]
- Part 400: Principles and requirements for physical input devices [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38896]
- Part 410: Design criteria for physical input devices [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38899]
- Part 420: Selection of physical input devices [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=52938]
- Part 910: Framework for tactile and haptic interaction [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51097]
- Part 920: Guidance on tactile and haptic interactions [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=42904]

See also

- ISO 9241-10
- ISO 9241-11

7.24. ISO 9241-10

International Standard ISO 9241-10

Ergonomic requirements for office work with visual display terminals (VDTs)

Part 10: Dialogue principles

Year: 1996.

This standard is withdrawn, and revised by ISO 9241-110:2006

Access

Online ISO Catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16882

See also

→ ISO 9241

7.25. ISO 9241-11

International Standard ISO 9241-11

Ergonomic requirements for office work with visual display terminals (VDTs)

Part 11: Guidance on usability

Year: 1998.

Contents

Extract from www.ansi.org [<http://webstore.ansi.org/RecordDetail.aspx?sku=ISO+9241-11%3a1998&source=google&adgroup=iso8&keyword=ISO%209241-11&gclid=CJiXjPD8jKoCFcEJtAodkmmDyQ>]:

ISO 9241-11 defines usability and explains how to identify the information which is necessary to take into account when specifying or evaluating usability of a visual display terminal in terms of measures of user performance and satisfaction. Guidance is given on how to describe the context of use of the product (hardware, software or service) and the relevant measures of usability in an explicit way. The guidance is given in the form of general principles and techniques, rather than in the form of requirements to use specific methods.

The guidance in ISO 9241-11 can be used in procurement, design, development, evaluation, and communication of information about usability. ISO 9241-11 includes guidance on how the usability of a product can be specified and evaluated. It applies both to products intended for general application and products being acquired for or being developed within a specific organization.

ISO 9241-11 also explains how measures of user performance and satisfaction can be used to measure how any component of a work system affects the whole work system in use. The guidance includes procedures for measuring usability but does not detail all the activities to be undertaken. Specification of detailed user-based methods of measurement is beyond the scope of ISO 9241-11, but further information can be found in Annex B and the bibliography in Annex E.

ISO 9241-11 applies to office work with visual display terminals. It can also apply in other situations where a user is interacting with a product to achieve goals. ISO 9241 parts 12 to 17 provide conditional recommendations which are applicable in specific contexts of use. The guidance in this Part of ISO 9241 can

be used in conjunction with ISO 9241 Parts 12 to 17 in order to help identify the applicability of individual recommendations.

ISO 9241-11 focuses on usability and does not provide comprehensive coverage of all objectives of ergonomic design referred to in ISO 6385. However, design for usability will contribute positively to ergonomic objectives, such as the reduction of possible adverse effects of use on human health, safety and performance.

ISO 9241-11 does not cover the processes of system development. Human-centred design processes for interactive systems are described in ISO 13407.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16883

See also

→ ISO 9241

7.26. ISO/IEC 12119

Information technology - Software packages - Quality requirements and testing

Year: 1994

7.27. ISO/IEC 12207

International Standard ISO/IEC 12207

Information technology -- Software lifecycle processes

Year: 1995, 2008.

Access

Online ISO/IEC Catalog: http://www.iso.org/iso/catalogue_detail.htm?csnumber=43447

7.28. ISO/IEC 14143

International Standard ISO/IEC 14143

Information technology -- Software measurement -- Functional size measurement

Contents

- Part 1: Definition of concepts
- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998
- Part 3: Verification of functional size measurement methods
- Part 4: Reference model
- Part 5: Determination of functional domains for use with functional size measurement
- Part 6: Guide for use of ISO/IEC 14143 series and related International Standards

See also

Standards:

→ ISO/IEC 14143-1

→ ISO/IEC 14143-3

7.29. ISO/IEC 14143-1

International Standard ISO/IEC 14143-1

Information technology -- Software measurement -- Functional size measurement

Part 1: Definition of concepts

Years: 1998, 2007.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38931

See also

Standards:

→ ISO/IEC 14143

→ ISO/IEC 14143-3

7.30. ISO/IEC 14143-3

International Standard ISO/IEC 14143-1

Information technology -- Software measurement -- Functional size measurement

Part 3: Verification of functional size measurement methods

Year: 2003.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=31918

See also

Standards:

→ ISO/IEC 14143

→ ISO/IEC 14143-1

7.31. ISO/IEC 14598

International Standard ISO/IEC 14598.

Information technology -- Software product evaluation

Contents / Access

Online ISO/IEC catalog:

- Part 1: General overview [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24902]
- Part 2: Planning and management [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24903]
- Part 3: Process for developers [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24904]
- Part 4: Process for acquirers [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24905]
- Part 5: Process for evaluators [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24906]
- Part 6: Documentation of evaluation modules [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24907]

See also

- ISO/IEC 14598-1
- ISO/IEC 14598-2
- ISO/IEC 14598-3
- ISO/IEC 14598-4
- ISO/IEC 14598-5
- ISO/IEC 14598-6

7.32. ISO/IEC 14598-1

International Standard ISO/IEC 14598-1

Information technology - Software product evaluation

Part 1: General overview

Year: 1999

This standard is revised by the ISO/IEC 25040:2011 standard.

Access

Online ISO catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24902

See also

- ISO/IEC 14598
- ISO/IEC 14598-2
- ISO/IEC 14598-3
- ISO/IEC 14598-4
- ISO/IEC 14598-5
- ISO/IEC 14598-6

7.33. ISO/IEC 14598-2

International Standard ISO/IEC 14598-2

Information technology - Software product evaluation

Part 2: Planning and management

Year: 2000.

This standard is revised by the ISO/IEC 25001:2007 standard.

Access

Online ISO catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24903

See also

- ISO/IEC 14598
- ISO/IEC 14598-1
- ISO/IEC 14598-3
- ISO/IEC 14598-4
- ISO/IEC 14598-5
- ISO/IEC 14598-6

7.34. ISO/IEC 14598-3

International Standard ISO/IEC 14598-3

Information technology - Software product evaluation

Part 3: Process for developers

Year: 2000.

Access

Online ISO catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24904

See also

- ISO/IEC 14598
- ISO/IEC 14598-1
- ISO/IEC 14598-2
- ISO/IEC 14598-4
- ISO/IEC 14598-5
- ISO/IEC 14598-6

7.35. ISO/IEC 14598-4

International Standard ISO/IEC 14598-4

Information technology - Software product evaluation

Part 4: Process for acquirers

Year: 1999.

Access

Online ISO catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24905

See also

- ISO/IEC 14598
- ISO/IEC 14598-1
- ISO/IEC 14598-2
- ISO/IEC 14598-3
- ISO/IEC 14598-5
- ISO/IEC 14598-6

7.36. ISO/IEC 14598-5

International Standard ISO/IEC 14598-5

Information technology - Software product evaluation

Part 5: Process for evaluators

Year: 1998

Access

Online ISO catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24906

See also

- ISO/IEC 14598
- ISO/IEC 14598-1
- ISO/IEC 14598-2
- ISO/IEC 14598-3
- ISO/IEC 14598-4
- ISO/IEC 14598-6

7.37. ISO/IEC 14598-6

International Standard ISO/IEC 14598-6

Information technology - Software product evaluation

Part 6: Documentation of evaluation modules

Year: 2001.

This standard is revised by the ISO/IEC DIS 25041 standard.

Access

Online ISO catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=24907

See also

- ISO/IEC 14598
- ISO/IEC 14598-1
- ISO/IEC 14598-2
- ISO/IEC 14598-3
- ISO/IEC 14598-4
- ISO/IEC 14598-5

7.38. ISO/IEC 14756

International Standard ISO/IEC 14756

Information technology -- Measurement and rating of performance of computer-based software systems

Year: 1999

Access

Online IEEE Catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=25492

7.39. ISO/IEC 14764

International Standard ISO/IEC 14764

Software Engineering -- Software Life Cycle Processes -- Maintenance

Years: 1999, 2006.

Access

Online IEEE Catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39064

7.40. ISO/IEC 15026

International Standard ISO/IEC 15026

Information technology -- System and software integrity levels

Year: 1998, 2010, 2011.

Access

Online ISO Catalog:

→ Part 1: Concepts and vocabulary [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50520]

→ Part 2: Assurance case [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=52926]

See also

→ ISO/IEC 15026-1

→ ISO/IEC 15026-2

7.41. ISO/IEC 15026-1

International Standard ISO/IEC 15026-1

Systems and software engineering -- Systems and software assurance -- Part 1: Concepts and vocabulary

Year: 2010.

See also

→ ISO/IEC 15026

→ ISO/IEC 15026-2

7.42. ISO/IEC 15026-2

International Standard ISO/IEC 15026-2

Systems and software engineering -- Systems and software assurance -- Part 2: Assurance case

Year: 2011.

See also

- ISO/IEC 15026
- ISO/IEC 15026-1

7.43. ISO/IEC 15288

International Standard ISO/IEC 15288.

Systems and software engineering -- System life cycle processes

Years: 2002.

Access

Online ISO catalog:

- http://www.iso.org/iso/catalogue_detail?csnumber=43564

See also

Standards:

- ISO/IEC 12207

External Links:

- ISO/IEC 15288 association home page: <http://www.15288.com>

7.44. ISO/IEC 15289

International Standard ISO/IEC 15289.

Systems and software engineering -- Content of systems and software life cycle process information products (Documentation)

Year: 2006.

This standard is revised by the ISO/IEC/IEEE 15289 standard.

Access

Online ISO catalog:

- http://www.iso.org/iso/catalogue_detail?csnumber=43790

See also

Standards:

- ISO/IEC 12207
- ISO/IEC 15288
- ISO/IEC/IEEE 15289

7.45. ISO/IEC 15414

International Standard ISO/IEC 15414

Information technology -- Open distributed processing -- Reference model -- Enterprise language

Years: 2002, 2006.

Access

Online IEEE Catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43767

7.46. ISO/IEC 15474

International Standard ISO/IEC 15474

Information technology -- CDIF framework

Contents

- Part 1: Overview
- Part 2: Modelling and extensibility

See also

- ISO/IEC 15474-1
- ISO/IEC 15474-2

7.47. ISO/IEC 15474-1

International Standard 15474-1

Information technology -- CDIF framework

Part 1: Overview

Year: 2002.

Access

Online ISO catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=27825

See also

Standards:

- ISO/IEC 15474
- ISO/IEC 15474-2

7.48. ISO/IEC 15474-2

International Standard 15474-2

Information technology -- CDIF framework

Part 2: Modelling and extensibility

Year: 2002.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=29029

See also

Standards:

- ISO/IEC 15474
- ISO/IEC 15474-1

7.49. ISO/IEC 15504

International Standard ISO/IEC 15504.

Information technology - Software Process Assessment

Also known as SPICE -- Software Process Improvement and Capability dEtermination.

Years: 1998, 2003, 2004, 2008.

Contents

- Part 1: Concepts and vocabulary
- Part 2: Performing an assessment
- Part 3: Guidance on performing an assessment
- Part 4: Guidance on use for process improvement and process capability determination
- Part 5: An exemplar Process Assessment Model
- Part 6: An exemplar system life cycle process assessment model
- Part 7: Assessment of organizational maturity

See also

Standards:

- ISO/IEC 15504-1
- ISO/IEC 15504-2
- ISO/IEC 15504-3
- ISO/IEC 15504-4

- ISO/IEC 15504-5
- ISO/IEC 15504-6
- ISO/IEC 15504-7

External Links:

- SPICE User Group home page: <http://www.spiceusergroup.org>

7.50. ISO/IEC 15504-1

International Standard ISO/IEC 15504.

Information technology - Software Process Assessment

Part 1: Concepts and vocabulary

Year: 1998, 2004.

This standard revises the ISO/IEC TR 15504-1:1998 and ISO/IEC TR 15504-9:1998 standards.

Access

Online ISO catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38932

See also

Standards:

- ISO/IEC 15504
- ISO/IEC 15504-2
- ISO/IEC 15504-3
- ISO/IEC 15504-4
- ISO/IEC 15504-5
- ISO/IEC 15504-6
- ISO/IEC 15504-7

7.51. ISO/IEC 15504-2

International Standard ISO/IEC 15504.

Information technology - Software Process Assessment

Part 2: Performing an assessment

Year: 1998, 2003.

This standard revises the ISO/IEC TR 15504-2:1998 and ISO/IEC TR 15504-3:1998 standards.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=37458

See also

Standards:

- ISO/IEC 15504
- ISO/IEC 15504-1
- ISO/IEC 15504-3
- ISO/IEC 15504-4
- ISO/IEC 15504-5
- ISO/IEC 15504-6
- ISO/IEC 15504-7

7.52. ISO/IEC 15504-3

International Standard ISO/IEC 15504.

Information technology - Software Process Assessment

Part 3: Guidance on performing an assessment

Year: 1998, 2004.

This standard revises the ISO/IEC TR 15504-4:1998 and ISO/IEC TR 15504-6:1998 standards.

Access

Online ISO catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=37454

See also

Standards:

- ISO/IEC 15504
- ISO/IEC 15504-1
- ISO/IEC 15504-2
- ISO/IEC 15504-4
- ISO/IEC 15504-5
- ISO/IEC 15504-6
- ISO/IEC 15504-7

7.53. ISO/IEC 15504-4

International Standard ISO/IEC 15504.

Information technology - Software Process Assessment

Part 4: Guidance on use for process improvement and process capability determination

Year: 1998, 2004.

This standard revises the ISO/IEC TR 15504-7:1998 and ISO/IEC TR 15504-8:1998 standards.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=37462

See also

Standards:

- ISO/IEC 15504
- ISO/IEC 15504-1
- ISO/IEC 15504-2
- ISO/IEC 15504-3
- ISO/IEC 15504-5
- ISO/IEC 15504-6
- ISO/IEC 15504-7

7.54. ISO/IEC 15504-5

International Standard ISO/IEC 15504.

Information technology - Software Process Assessment

Part 5: An exemplar Process Assessment Model

Year: 1998, 2006.

This standard revises the ISO/IEC TR 15504-5:1998 standard.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=37462

See also

Standards:

- ISO/IEC 15504
- ISO/IEC 15504-1
- ISO/IEC 15504-2
- ISO/IEC 15504-3
- ISO/IEC 15504-4
- ISO/IEC 15504-6

→ ISO/IEC 15504-7

7.55. ISO/IEC 15504-6

International Standard ISO/IEC 15504.

Information technology - Software Process Assessment

Part 6: An exemplar system life cycle process assessment model

Year: 1998, 2008.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43446

See also

Standards:

- ISO/IEC 15504
- ISO/IEC 15504-1
- ISO/IEC 15504-2
- ISO/IEC 15504-3
- ISO/IEC 15504-4
- ISO/IEC 15504-5
- ISO/IEC 15504-7

7.56. ISO/IEC 15504-7

International Standard ISO/IEC 15504.

Information technology - Software Process Assessment

Part 7: Assessment of organizational maturity

Year: 1998, 2008.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50519

See also

Standards:

- ISO/IEC 15504
- ISO/IEC 15504-1

- ISO/IEC 15504-2
- ISO/IEC 15504-3
- ISO/IEC 15504-4
- ISO/IEC 15504-5
- ISO/IEC 15504-6

7.57. ISO/IEC 15846

International Standard ISO/IEC 15846

Information technology -- Software life cycle processes -- Configuration Management

Year: 1998.

This standard has been withdrawn.

Access

Online ISO Catalog:

- http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=30516

See also

Standards:

- ISO/IEC 12207

7.58. ISO/IEC 15910

International Standard ISO/IEC 15910.

Information technology -- Software user documentation process

Year:1999.

This standard is revised by the ISO/IEC 26512:2011 standard.

Notes

Extract from www.techstreet.com [http://www.techstreet.com/cgi-bin/detail?doc_no=iso_iec15910_1999&product_id=862851]:

This International Standard specifies the minimum process for creating all forms of user documentation for software which has a user interface. Such forms of documentation include printed documentation (e.g. user manuals and quick-reference cards), on-line documentation, help text and on-line documentation systems.

This International Standard conforms with ISO/IEC 12207:1995, Information technology Software life cycle processes, as an implementation of the user documentation part of 6.1: Documentation.

If effectively applied, this International Standard will support the development of documentation which meets the needs of the users.

This International Standard is intended for use by anyone who produces or buys user documentation.

This International Standard is applicable to not only printed documentation, but also help screens, the help delivery system, and the on-line text and delivery system.

This International Standard is intended for use in a two-party situation and may be equally applied where the two parties are from the same organization. The situation may range from an informal agreement up to a legally binding contract. This International Standard may be used by a single party as self-imposed tasks.

7.59. ISO/IEC 15939

International Standard ISO/IEC 15939

Software engineering - Software measurement process

Year: 2002, 2007.

Access

Online ISO Catalog:

→ ISO/IEC 15939:2007 [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=44344]

See also

7.60. ISO/IEC 19759

International Standard ISO/IEC 19759

Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK)

Year: 2005.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=33897

See also

External Links:

→ The Official SWEBOK webpage: <http://www.computer.org/portal/web/swebok>

7.61. ISO/IEC 19770

International Standard ISO/IEC 19770

Information technology -- Software asset management

Contents

→ Part 1: Processes

→ Part 2: Software identification tag

See also

Standards:

→ ISO/IEC 19770-1

→ ISO/IEC 19770-2

External Links:

→ The SAM standard Working Group website: <http://www.19770.org>

7.62. ISO/IEC 19770-1

International Standard ISO/IEC 19770-1

Information technology -- Software asset management

Part 1: Processes

Year: 2006.

Access

Online ISO catalog:

→ http://www.iso.org/iso/catalogue_detail?csnumber=33908

See also

→ ISO/IEC 19770

→ ISO/IEC 19770-2

7.63. ISO/IEC 19770-2

International Standard ISO/IEC 19770-2

Information technology -- Software asset management

Part 2: Software identification tag

Year: 2009.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=53670

See also

→ ISO/IEC 19770

→ ISO/IEC 19770-1

7.64. ISO/IEC 20000

International Standard ISO/IEC 20000

Information technology -- Service management

Contents/Access

This list links to the online ISO catalog:

- Part 1: Specification [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51986]
- Part 2: Code of practice [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=41333]
- Part 3: Guidance on scope definition and applicability of ISO/IEC 20000-1 [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51235]
- Part 4: Process reference model [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50624]
- Part 5: Exemplar implementation plan for ISO/IEC 20000-1 [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51988]

7.65. ISO/IEC 2382

International Standard ISO/IEC 2382

Information processing systems -- Vocabulary

Contents

- Part 1: Quality Model [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7229]
- Part 2: Arithmetic and logic operations [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7230]
- Part 3: Equipment technology [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7232]
- Part 4: Organization of data [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=27922]
- Part 5: Representation of data [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=30851]
- Part 6: Preparation and handling of data [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7238]
- Part 7: Computer programming [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7241]
- Part 8: Security [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7243]
- Part 9: Data communication [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=20929]
- Part 10: Operating techniques and facilities [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7247]
- Part 12: Peripheral equipment [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7251]

- Part 13: Computer graphics [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7253]
- Part 14: Reliability, maintainability and availability [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7255]
- Part 15: Programming languages [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7257]
- Part 16: Information theory [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7259]
- Part 17: Databases [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=30853]
- Part 18: Distributed data processing [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=26734]
- Part 19: Analog computing [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7263]
- Part 20: System development [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7264]
- Part 21: Interfaces between process computer systems and technical processes [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7265]
- Part 23: Text processing [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7267]
- Part 24: Computer-integrated manufacturing [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7268]
- Part 25: Local area networks [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7269]
- Part 26: Open systems interconnection [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7270]
- Part 27: Office automation [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7271]
- Part 28: Artificial intelligence -- Basic concepts and expert systems [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7272]
- Part 29: Artificial intelligence -- Speech recognition and synthesis [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7273]
- Part 31: Artificial intelligence -- Machine learning [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=21845]
- Part 32: Electronic Mail [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=21846]
- Part 34: Artificial intelligence -- Neural networks [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=21848]
- Part 36: Learning, education and training [http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=46152]

See also

- ISO/IEC 2382-1

7.66. ISO/IEC 2382-1

International Standard ISO/IEC 2382

Information technology - Vocabulary

Part 1: Fundamental terms

Year: 1993.

Access

Online ISO Catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7229

See also

Standards:

→ ISO/IEC 2382

7.67. ISO/IEC 25000

International Standard ISO/IEC 25000

Software Engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE.

year: 2005.

This series of standards revises the ISO/IEC 9126 and ISO/IEC 14598 series.

Access

→ Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35683

See also

→ ISO/IEC 9126

→ ISO/IEC 14598

→ ISO/IEC SQuaRE

7.68. ISO/IEC 25001

International Standard ISO/IEC 25001

Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Planning and management

year: 2007.

This standard revises the ISO/IEC 14598-2.

Access

→ Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35724

See also

Standards:

- ISO/IEC 9126
- ISO/IEC 14598
- ISO/IEC SQuaRE

7.69. ISO/IEC 25010

International Standard ISO/IEC 25010

Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models

year: 2011.

Access

- Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35733

See also

- ISO/IEC 9126
- ISO/IEC 25000

7.70. ISO/IEC 25012

International Standard ISO/IEC 25012

Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Data quality model

year: 2008.

Access

- Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35736

See also

- ISO/IEC 9126
- ISO/IEC 25010
- ISO/IEC SQuaRE

7.71. ISO/IEC 25020

International Standard ISO/IEC 25020

Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Measurement reference model and guide

Year: 2007.

Access

→ Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35744

See also

- ISO/IEC 25010
- ISO/IEC 25030
- ISO/IEC 25040
- ISO/IEC SQuaRE

7.72. ISO/IEC 25021

International Standard ISO/IEC 25021

Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Quality measure elements

Year: 2007.

Access

→ Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35745

See also

- ISO/IEC 9126
- ISO/IEC 25030
- ISO/IEC 25040
- ISO/IEC SQuaRE

7.73. ISO/IEC 25030

International Standard ISO/IEC 25030

Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Quality requirements

year: 2007.

Access

→ Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35765

See also

- ISO/IEC 9126-1

- ISO/IEC 25010
- ISO/IEC SQuaRE

7.74. ISO/IEC 25040

International Standard ISO/IEC 25040

Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- Evaluation process

year: 2011.

This standard revises the ISO/IEC 14598-1 standard.

Access

- Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35765

See also

- ISO/IEC 9126
- ISO/IEC 14598
- ISO/IEC SQuaRE

7.75. ISO/IEC 25045

International Standard ISO/IEC 25045

Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- Evaluation module for recoverability

Year: 2010.

Access

- Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35683

See also

- ISO/IEC SQuaRE

7.76. ISO/IEC 25051

International Standard ISO/IEC 25000

Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing

year: 2006, 2007.

This standard revises the ISO/IEC 12119 standard.

Access

→ Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=37457

See also

Glossary:

→ COTS

Standards:

→ ISO/IEC 9126

→ ISO/IEC 14598

→ ISO/IEC SQuaRE

7.77. ISO/IEC 25060

International Standard ISO/IEC 25060

Systems and software engineering -- Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Common Industry Format (CIF) for usability: General framework for usability-related information

Year: 2010.

Access

→ Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35786

See also

→ ISO/IEC 9126

→ ISO/IEC SQuaRE

7.78. ISO/IEC 25062

International Standard ISO/IEC 25062

Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Common Industry Format (CIF) for usability test reports

Year: 2006.

Access

→ Online ISO catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43046

See also

→ ISO/IEC SQuaRE

7.79. ISO/IEC 26514

International Standard ISO/IEC 26514

Systems and software engineering -- Requirements for designers and developers of user documentation

Year: 2008

Access

Online IEEE Catalog: http://www.iso.org/iso/catalogue_detail?csnumber=43073

7.80. ISO/IEC 29881

International Standard ISO/IEC/IEEE 29881

Information Technology — Software and Systems Engineering

Year: 2008, 2010

Access

Online IEEE Catalog: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=56418

7.81. ISO/IEC 90003

International Standard ISO/IEC 90003

Software engineering -- Guidelines for the application of ISO 9001:2000 to computer software

Year: 2004

Access

Online IEEE Catalog: http://www.iso.org/iso/catalogue_detail?csnumber=35867

7.82. ISO/IEC 9126

International Standard ISO/IEC 9126

Software engineering -- Product quality

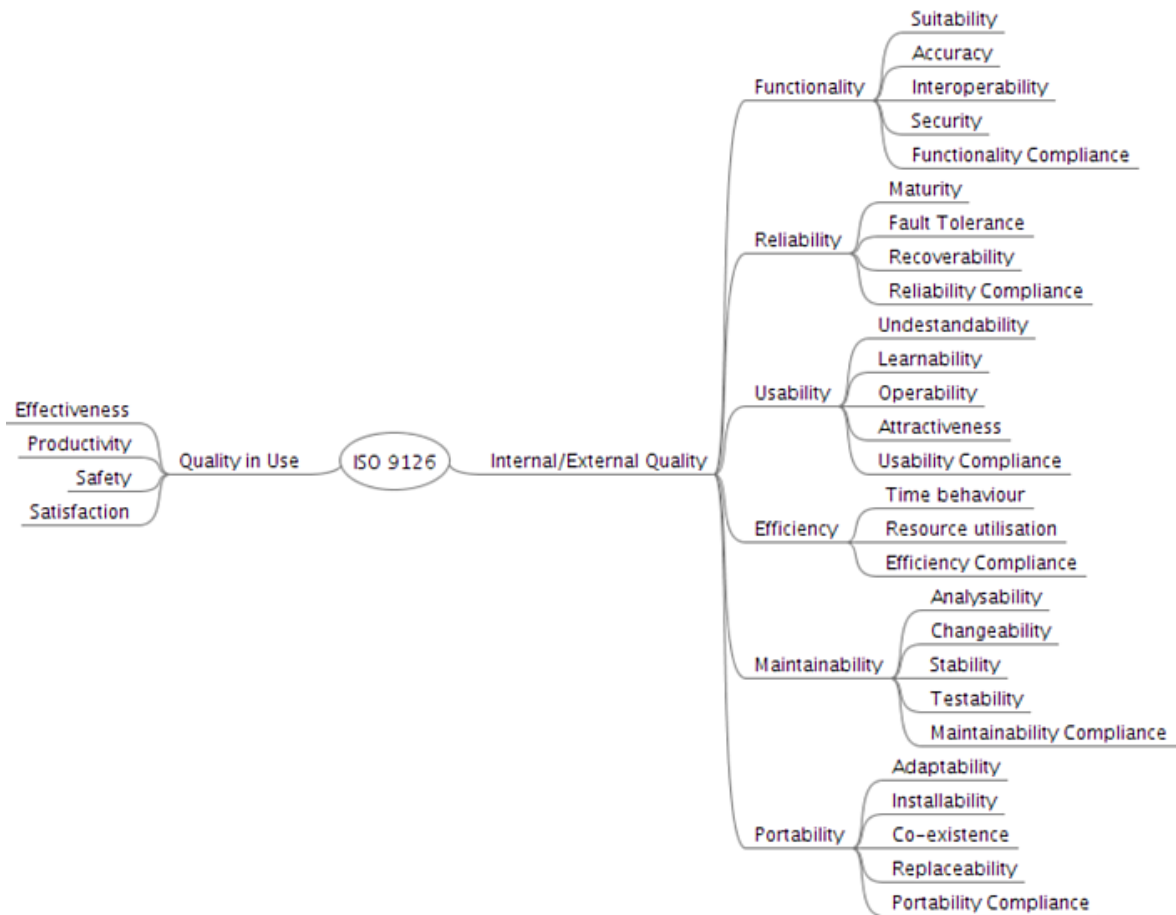
Years: 1991, 2001

This standard is revised by ISO/IEC 25010:2011.

Contents

- Part 1: Quality Model
- Part 2: External metrics
- Part 3: Internal metrics
- Part 4: Quality in use metrics

Structure



See also

- ISO/IEC 9126-1
- ISO/IEC 9126-2
- ISO/IEC 9126-3
- ISO/IEC 9126-4
- ISO/IEC 25000
- ISO/IEC 25010

7.83. ISO/IEC 9126-1

International Standard ISO/IEC 9126-1

Software engineering -- Product quality

Part 1: Quality Model

Years: 1991, 2001.

This standard is revised by ISO/IEC 25010:2011.

Access

Online ISO Catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=22749

See also

- ISO 9001
- ISO/IEC 9126
- ISO/IEC 9126-2
- ISO/IEC 9126-3
- ISO/IEC 9126-4
- ISO/IEC 12207
- ISO/IEC 15504
- ISO/IEC 14598

7.84. ISO/IEC 9126-2

International Standard ISO/IEC 9126-2

Software engineering -- Product quality

Part 2: External metrics

Years: 1991, 2001.

This standard is revised by ISO/IEC 25010:2011.

Access

Online ISO Catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=22750

See also

- ISO/IEC 9126
- ISO/IEC 9126-1
- ISO/IEC 9126-3
- ISO/IEC 9126-4

7.85. ISO/IEC 9126-3

International Standard ISO/IEC 9126-3

Software engineering -- Product quality

Part 3: Internal metrics

Years: 1991, 2001.

This standard is revised by ISO/IEC 25010:2011.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=22891

See also

- ISO/IEC 9126
- ISO/IEC 9126-1
- ISO/IEC 9126-2
- ISO/IEC 9126-4

7.86. ISO/IEC 9126-4

International Standard ISO/IEC 9126-4

Software engineering -- Product quality

Part 4: Quality in use metrics

Years: 1991, 2001, 2004.

This standard is revised by ISO/IEC 25010:2011.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39752

See also

- ISO/IEC 9126
- ISO/IEC 9126-1
- ISO/IEC 9126-2
- ISO/IEC 9126-3

7.87. ISO/IEC 9294

International Standard ISO/IEC 9294

Information technology -- Guidelines for the management of software documentation

Years: 1990, 2005.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=37460

See also

Glossary:

- Document
- Documentation

7.88. ISO/IEC 99

International Standard ISO/IEC 99

International vocabulary of metrology -- Basic and general concepts and associated terms

Years: 1993, 2007.

See also

- The Joint Committee for Guides in Metrology [<http://www.iso.org/sites/JCGM/JCGM-introduction.htm>] has an online version of a document presenting the main points of the ISO/IEC 99: http://www.iso.org/sites/JCGM/VIM/JCGM_200e.html.

7.89. ISO/IEC SQuaRE

International Standard ISO/IEC SQuaRE

Systems and software Quality Requirements and Evaluation (SQuaRE)

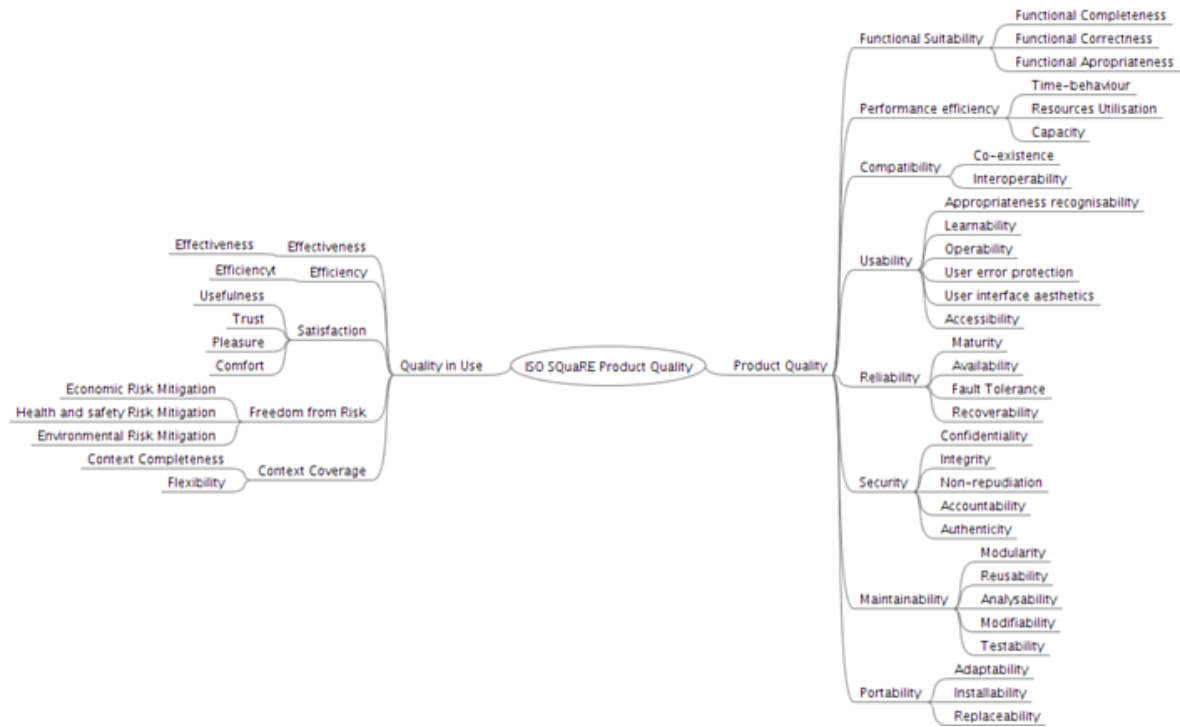
SQuaRE is a series of International Standards (25000-25099) edited by the ISO/IEC organisation and related to Systems and Software Quality.

It is composed of the following ISO/IEC standards:

- ISO/IEC 25000 -- Guide to SQuaRE
- ISO/IEC 25001 -- Planning and management
- ISO/IEC 25010 -- System and software quality models
- ISO/IEC 25012 -- Data quality model
- ISO/IEC 25020 -- Measurement reference model and guide
- ISO/IEC 25021 -- Quality measure elements
- ISO/IEC 25030 -- Quality requirements
- ISO/IEC 25040 -- Evaluation process
- ISO/IEC 25045 -- Evaluation module for recoverability
- ISO/IEC 25051 -- Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing
- ISO/IEC 25060 -- Common Industry Format (CIF) for usability: General framework for usability-related information
- ISO/IEC 25062 -- Common Industry Format (CIF) for usability test reports

They are meant to replace older standards addressing the same topics, mainly (but not only) ISO/IEC 9126 and ISO/IEC 14598.

Structure



See also

- ISO/IEC 9126
- ISO/IEC 14598
- ISO/IEC 25000
- ISO/IEC 25001
- ISO/IEC 25010
- ISO/IEC 25012
- ISO/IEC 25020
- ISO/IEC 25021
- ISO/IEC 25030
- ISO/IEC 25040
- ISO/IEC 25045
- ISO/IEC 25051
- ISO/IEC 25060
- ISO/IEC 25062

7.90. ISO/IEC/IEEE 15289

International Standard ISO/IEC/IEEE 15289.

Systems and software engineering -- Content of life-cycle information products (documentation)

Years: 2006, 2011.

This standard revises the ISO/IEC 15289 standard.

Access

Online ISO catalog:

→ http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=54388

See also

Standards:

→ ISO/IEC 12207

→ ISO/IEC 15288

→ ISO/IEC 15289

→ ISO/IEC 20000

7.91. ISO/IEC/IEEE 24765

International Standard ISO/IEC/IEEE 24765

Systems and software engineering — Vocabulary

First edition: 2010-12-15

Access

Online IEEE Catalog: http://www.iso.org/iso/catalogue_detail.htm?csnumber=50518

7.92. RTCA/EUROCAE

Software Considerations in Airborne Systems and Equipments Certification

Requirements and Technical Concepts for Aviation - RTCA SC167/DO-178B

European Organization for Civil Aviation Electronics - EUROCAE ED-12B

7.93. SIGIST

Glossary of terms used in Software testing

British Computer Society - Specialist Interest Group In Software Testing

7.94. Team Software Process

Team Software Process is a process developed by the Carnegie-Mellon Software Engineering Institute.

The Team Software Process (TSP) helps engineering teams develop and deliver high-quality software-intensive systems within planned cost and schedule commitments. TSP integrates software engineering, estimating,

planning and tracking, quality management, and self-directed teaming concepts into a defined process and measurement framework. TSP was designed to be easily integrated with an organization's existing practices, and complements CMMI.

= See also =

→ CMMi

→ The official SEI website for TSP: www.sei.cmu.edu/tsp/ [<http://www.sei.cmu.edu/tsp/>].

Appendix A. Data Provider Frameworks

```
=====
= Csv =
=====
```

The Csv framework is used to import metrics or textual information and attach them to artefacts of type Application, File or Function. While parsing one or more input CSV files, if it finds the same metric for the same artefact several times, it will only use the last occurrence of the metric and ignore the previous ones. Note that the type of artefacts you can attach metrics to is limited to Application, File and Function artefacts. If you are working with File artefacts, you can let the Data Provider create the artefacts by itself if they do not exist already.

```
=====
= form.xml =
=====
```

You can customise form.xml to either:

- specify the path to a single CSV file to import
- specify a pattern to import all csv files matching this pattern in a directory

In order to import a single CSV file:

```
=====
<?xml version="1.0" encoding="UTF-8"?>
<tags baseName="Csv" needSources="true">
  <tag type="text" key="csv" defaultValue="/path/to/mydata.csv" />
</tags>
```

Notes:

- The csv key is mandatory.
- Since Csv-based data providers commonly rely on artefacts created by Squan Sources, you can set the needSources attribute to force users to specify at least one repository connector when creating a project.

In order to import all files matching a pattern in a folder:

```
=====
<?xml version="1.0" encoding="UTF-8"?>
<tags baseName="Csv" needSources="true">
  <!-- Root directory containing Csv files to import-->
  <tag type="text" key="dir" defaultValue="/path/to/mydata" />
  <!-- Pattern that needs to be matched by a file name in order to import it-->
  <tag type="text" key="ext" defaultValue="*.csv" />
  <!-- search for files in sub-folders -->
  <tag type="booleanChoice" defaultValue="true" key="sub" />
</tags>
```

Notes:

- The dir and ext keys are mandatory
- The sub key is optional (and its value set to false if not specified)

```
=====
= config.tcl =
=====
```

Sample config.tcl file:

```
=====
# The separator used in the input CSV file
# Usually \t or ;
set Separator "\t"

# The delimiter used in the input CSV file
# This is normally left empty, except when you know that some of the values in
the CSV file
# contain the separator itself, for example:
# "A text containing ; the separator";no problem;end
# In this case, you need to set the delimiter to \" in order for the data
provider to find 3 values instead of 4.
# To include the delimiter itself in a value, you need to escape it by
duplicating it, for example:
# "A text containing \" the delimiter";no problemo;end
# Default: none
set Delimiter \"

# ArtefactLevel is one of:
#   Application: to import data at application level
#   File: to import data at file level. In this case ArtefactKey has to be set
#   to the value of the header (key) of the column containing the file
path
#   in the input CSV file.
#   Function : to import data at function level, in this case:
#   ArtefactKey has to be set to the value of the header (key) of
the column containing the path of the file
#   FunctionKey has to be set to the value of the header (key) of
the column containing the name and signature of the function
# Note that the values are case-sensitive.
set ArtefactLevel File
set ArtefactKey File

# Should the File paths be case-insensitive?
# true or false (default)
# This is used when searching for a matching artefact in already-existing
artefacts.
set PathsAreCaseInsensitive "false"

# Should file artefacts declared in the input CSV file be created automatically?
# true (default) or false
set CreateMissingFile "true"

# FileOrganisation defines the layout of the input CSV file and is one of:
#   header::column: values are referenced from the column header
#   header::line: NOT AVAILABLE
#   alternate::line: lines are a sequence of {Key Value}
#   alternate::column: columns are a sequence of {Key Value}
# There are more examples of possible CSV layouts later in this document
set FileOrganisation header::column

# Metric2Key contains a case-sensitive list of paired metric IDs:
#   {MeasureID KeyName [Format]}
# where:
#   - MeasureID is the id of the measure as defined in your analysis model
#   - KeyName, depending on the FileOrganisation, is either the name of the
column or the name
#   in the cell preceding the value to import as found in the input CSV file
#   - Format is the optional format of the data, the only accepted format
```

```
#      is "text" to attach textual information to an artefact, for normal metrics
omit this field
set Metric2Key {
  {BRANCHES Branchs}
  {VERSIONS Versions}
  {CREATED Created}
  {IDENTICAL Identical}
  {ADDED Added}
  {REMOV Removed}
  {MODIF Modified}
  {COMMENT Comment text}
}

=====
= Sample CSV Input Files =
=====

Example 1:
=====
FileOrganisation : header::column
ArtefactLevel   : File
ArtefactKey     : Path

Path Branchs Versions
./foo.c 15 105
./bar.c 12 58

Example 2:
=====
FileOrganisation : alternate::line
ArtefactLevel   : File
ArtefactKey     : Path

Path ./foo.c Branchs 15 Versions 105
Path ./bar.c Branchs 12 Versions 58

Example 3:
=====
FileOrganisation : header::column
ArtefactLevel   : Application

ChangeRequest Corrected Open
27 15 11

Example 4:
=====
FileOrganisation : alternate::column
ArtefactLevel   : Application

ChangeRequest 15
Corrected 11

Example 5:
=====
FileOrganisation : alternate::column
ArtefactLevel   : File
ArtefactKey     : Path
```

```
Path ./foo.c
Branchs 15
Versions 105
Path ./bar.c
Branchs 12
Versions 58
```

Example 6:

```
=====
FileOrganisation : header::column
ArtefactLevel : Function
ArtefactKey : Path
FunctionKey : Name
```

```
Path Name Decisions Tested
./foo.c end_game(int*,int*) 15 3
./bar.c bar(char) 12 6
```

Working With Paths:

```
=====
```

- Path separators are unified: you do not need to worry about handling differences between Windows and Linux
- With the option `PathsAreCaseInsensitive`, case is ignored when searching for files in the Squore internal data
- Paths known by Squore are relative paths starting at the root of what was specified in the repository connector during the analysis. This relative path is the one used to match with a path in a csv file.

Here is a valid example of file matching:

1. You provide `C:\A\B\C\D` as the root folder in a repository connector
2. `C:\A\B\C\D` contains `E\e.c` then Squore will know `E/e.c` as a file
3. You provide a csv file produced on linux and containing `/tmp/X/Y/E/e.c` as path, then Squore will be able to match it with the known file.

Squore uses the longest possible match.

In case of conflict, no file is found and a message is sent to the log.

```
=====
= csv_findings =
=====
```

The `csv_findings` data provider is used to import findings (rule violations) and attach them to artefacts of type `Application`, `File` or `Function`. The format of the csv file given as parameter has to be:

```
FILE;FUNCTION;RULE_ID;MESSAGE;LINE;COL;STATUS;STATUS_MESSAGE;TOOL
```

where:

```
=====
```

`FILE` : is the full path of the file where the finding is located
`FUNCTION` : is the name of the function where the finding is located
`RULE_ID` : is the Squore ID of the rule which is violated
`MESSAGE` : is the specific message of the violation
`LINE`: is the line number where the violation occurs
`COL`: (optional, leave empty if not provided) is the column number where the violation occurs

STATUS: (optional, leave empty if not provided) is the status of the relaxation if the violation has to be relaxed (DEROGATION, FALSE_POSITIVE, LEGACY)
 STATUS_MSG: (optional, leave empty if not provided) is the message for the relaxation when relaxed
 TOOL: is the tool providing the violation

The header line is read and ignored (it has to be there)
 The separator (semicolon by default) can be changed in the config.tcl file (see below)
 The delimiter (no delimiter by default) can be changed in the config.tcl (see below)

```

=====
= config.tcl =
=====
  
```

Sample config.tcl file:

```

=====
# The separator used in the input CSV file
# Usually ; or \t
set Separator \;

# The delimiter used in the CSV input file
# This is normally left empty, except when you know that some of the values in
# the CSV file
# contain the separator itself, for example:
# "A text containing ; the separator";no problem;end
# In this case, you need to set the delimiter to \" in order for the data
# provider to find 3 values instead of 4.
# To include the delimiter itself in a value, you need to escape it by
# duplicating it, for example:
# "A text containing \" the delimiter";no problemo;end
# Default: none
set Delimiter \"
  
```

```

=====
= CsvPerl =
=====
  
```

The CsvPerl framework offers the same functionality as Csv, but instead of dealing with the raw input files directly, it allows you to run a perl script to modify them and produce a CSV file with the expected input format for the Csv framework.

```

=====
= form.xml =
=====
  
```

In your form.xml, specify the input parameters you need for your Data Provider. Our example will use two parameters: a path to a CSV file and another text parameter:

```

<?xml version="1.0" encoding="UTF-8"?>
<tags baseName="CsvPerl" needSources="true">
  <tag type="text" key="csv" defaultValue="/path/to/csv" />
  <tag type="text" key="param" defaultValue="MyValue" />
</tags>
  
```

- Since Csv-based data providers commonly rely on artefacts created by Squan Sources, you can set the needSources attribute to force users to specify at least one repository connector when creating a project.

```
=====
= config.tcl =
=====
```

Refer to the description of config.tcl for the Csv framework.

For CsvPerl one more option is possible:

```
# The variable NeedSources is used to request the perl script to be executed once
  for each
# repository node of the project. In that case an additional parameter is sent to
  the
# perl script (see below for its position)
#set ::NeedSources 1
```

```
=====
= Sample CSV Input Files =
=====
```

Refer to the examples for the Csv framework.

```
=====
= Perl Script =
=====
```

The perl script will receive as arguments:

- all parameters defined in form.xml (as `-${key} $value`)
- the input directory to process (only if `::NeedSources` is set to 1 in the config.tcl file)
- the location of the output directory where temporary files can be generated
- the full path of the csv file to be generated

For the form.xml we created earlier in this document, the command line will be:
`perl <configuration_folder>/tools/CustomDP/CustomDP.pl -csv /path/to/csv -param MyValue <output_folder> <output_folder>/CustomDP.csv`

Example of perl script:

```
=====
#!/usr/bin/perl
use strict;
use warnings;
$|=1 ;

($csvKey, $csvValue, $paramKey, $paramValue, $output_folder, $output_csv) =
  @ARGV;

# Parse input CSV file
# ...

# Write results to CSV
open(CSVFILE, ">" . ${output_csv}) || die "perl: can not write: ${!\n}";
```

```
binmode(CSVFILE, ":utf8");
print CSVFILE "ChangeRequest;15";
close CSVFILE;
```

```
exit 0;
```

```
=====
= Generic =
=====
```

The Generic framework is the most flexible Data Provider framework, since it allows attaching metrics, findings, textual information and links to artefacts. If the artefacts do not exist in your project, they will be created automatically. It takes one or more CSV files as input (one per type of information you want to import) and works with any type of artefact.

```
=====
= form.xml =
=====
```

In form.xml, allow users to specify the path to a CSV file for each type of data you want to import.

You can set needSources to true or false, depending on whether or not you want to require the use of a repository connector when your custom Data Provider is used.

Example of form.xml file:

```
=====
<?xml version="1.0" encoding="UTF-8"?>
<tags baseName="Generic" needSources="false">
  <!-- Path to CSV file containing Metrics data -->
  <tag type="text" key="csv" defaultValue="mydata.csv" />
  <!-- Path to CSV file containing Findings data: -->
  <tag type="text" key="fdg" defaultValue="mydata_fdg.csv" />
  <!-- Path to CSV file containing Information data: -->
  <tag type="text" key="inf" defaultValue="mydata_inf.csv" />
  <!-- Path to CSV file containing Links data: -->
  <tag type="text" key="lnk" defaultValue="mydata_lnk.csv" />
</tags>
```

Note: All tags are optional. You only need to specify the tag element for the type of data you want to import with your custom Data Provider.

```
=====
= config.tcl =
=====
```

Sample config.tcl file:

```
=====
# The separator used in the input csv files
# Usually \t or ; or ,
# In our example below, a space is used.
set Separator " "

# The delimiter used in the input CSV file
# This is normally left empty, except when you know that some of the values in
the CSV file
```

```
# contain the separator itself, for example:
# "A text containing ; the separator";no problem;end
# In this case, you need to set the delimiter to \" in order for the data
# provider to find 3 values instead of 4.
# To include the delimiter itself in a value, you need to escape it by
# duplicating it, for example:
# "A text containing \" the delimiter";no problemo;end
# Default: none
set Delimiter \"

# The path separator in an artefact's path
# in the input CSV file.
# Note that artefact is spelled with an "i"
# and not an "e" in this option.
set ArtifactPathSeparator "/"

# If the data provider needs to specify a different toolName (optional)
set SpecifyToolName 1

# Metric2Key contains a case-sensitive list of paired metric IDs:
#   {MeasureID KeyName [Format]}
# where:
# - MeasureID is the id of the measure as defined in your analysis model
# - KeyName is the name in the cell preceding the value to import as found in
#   the input CSV file
# - Format is the optional format of the data, the only accepted format
#   is "text" to attach textual information to an artefact. Note that the same
#   result can also
#   be achieved with Info2Key (see below). For normal metrics omit this
#   field.
set Metric2Key {
  {CHANGES Changed}
}

# Finding2Key contains a case-sensitive list of paired rule IDs:
#   {FindingID KeyName}
# where:
# - FindingID is the id of the rule as defined in your analysis model
# - KeyName is the name in the finding name in the input CSV file
set Finding2Key {
  {R_NOTLINKED NotLinked}
}

# Info2Key contains a case-sensitive list of paired info IDs:
#   {InfoID KeyName}
# where:
# - InfoID is the id of the textual information as defined in your analysis
#   model
# - KeyName is the name of the information name in the input CSV file
set Info2Key
  {SPECIAL_LABEL Label}
}

# Ignore findings for artefacts that are not part of the project (orphan
# findings)
# When set to 1, the findings are ignored
# When set to 0, the findings are imported and attached to the APPLICATION node
# (default: 1)
set IgnoreIfArtefactNotFound 1
```



```

# For findings of a type that is not in your ruleset, set a default rule ID.
# The value for this parameter must be a valid rule ID from your analysys model.
# (default: empty)
set UnknownRuleId UNKNOWN_RULE

# Save the total count of orphan findings as a metric at application level
# Specify the ID of the metric to use in your analysys model
# to store the information
# (default: empty)
set OrphanArteCountId NB_ORPHANS

# Save the total count of unknown rules as a metric at application level
# Specify the ID of the metric to use in your analysys model
# to store the information
# (default: empty)
set OrphanRulesCountId NB_UNKNOWN_RULES

# Save the list of unknown rule IDs as textual information at application level
# Specify the ID of the metric to use in your analysys model
# to store the information
# (default: empty)
set OrphanRulesListId UNKNOWN_RULES_INFO

```

```

=====
= CSV File Format =
=====

```

All the examples listed below assume the use of the following config.tcl:

```

set Separator ","
set ArtifactPathSeparator "/"
set Metric2Key {
  {CHANGES Changed}
}
set Finding2Key {
  {R_NOTLINKED NotLinked}
}
set Info2Key
  {SPECIAL_LABEL Label}
}

```

Layout for Metrics File:

```

=====
==> artefact_type artefact_path (Key Value)*

```

When the parent artefact type is not given it defaults to <artefact_type>_FOLDER.

Example:

```

REQ_MODULE,Requirements/Module
REQUIREMENT,Requirements/Module/My_Req,Changed,1

```

will produce the following artefact tree:

```

Application
  Requirements (type: REQ_MODULE_FOLDER)
    Module (type: REQ_MODULE)
      My_Req : (type: REQUIREMENT) with 1 metric CHANGES = 1

```

Note: the key "Changed" is mapped to the metric "CHANGES", as specified by the Metric2Key parameter, so that it matches what is expected by the model.

Layout for Findings File:

=====

==> artefact_type artefact_path key message

When the parent artefact type is not given it defaults to <artefact_type>_FOLDER.

Example:

REQ_MODULE,Requirements/Module

REQUIREMENT,Requirements/Module/My_Req,NotLinked,A Requirement should always been linked

will produce the following artefact tree:

Application

 Requirements (type: REQ_MODULE_FOLDER)

 Module (type: REQ_MODULE)

 My_Req (type: REQUIREMENT) with 1 finding R_NOTLINKED whose description is "A Requirement should always been linked"

Note: the key "NotLinked" is mapped to the finding "R_NOTLINKED", as specified by the Finding2Key parameter, so that it matches what is expected by the model.

Layout for Textual Information File:

=====

==> artefact_type artefact_path label value

When the parent artefact type is not given it defaults to <artefact_type>_FOLDER.

Example:

REQ_MODULE,Requirements/Module

REQUIREMENT,Requirements/Module/My_Req,Label,This is the label of the req

will produce the following artefact tree:

Application

 Requirements (type: REQ_MODULE_FOLDER)

 Module (type: REQ_MODULE)

 My_Req (type: REQUIREMENT) with 1 information of type SPECIAL_LABEL whose content is "This is the label of the req"

Note: the label "Label" is mapped to the finding "SPECIAL_LABEL", as specified by the Info2Key parameter, so that it matches what is expected by the model.

Layout for Links File:

=====

==> artefact_type artefact_path dest_artefact_type dest_artefact_path link_type

When the parent artefact type is not given it defaults to <artefact_type>_FOLDER

Example:

REQ_MODULE Requirements/Module

TEST_MODULE Tests/Module

REQUIREMENT Requirements/Module/My_Req TEST Tests/Module/My_test TESTED_BY

will produce the following artefact tree:

Application

```

Requirements (type: REQ_MODULE_FOLDER)
  Module (type: REQ_MODULE)
    My_Req (type: REQUIREMENT) ----->
Tests (type: TEST_MODULE_FOLDER)          |
  Module (type: TEST_MODULE)              |
    My_Test (type: TEST) <-----+ link (type: TESTED_BY)
  
```

The TESTED_BY relationship is created with My_Req as source of the link and My_test as the destination

CSV file organisation when SpecifyToolName is set to 1
 =====

When the variable SpecifyToolName is set to 1 (or true) a column has to be added at the beginning of each line in each csv file. This column can be empty or filled with a different toolName.

Example:

```

,REQ_MODULE,Requirements/Module
MyReqChecker,REQUIREMENT,Requirements/Module/My_Req Label,This is the label of
the req
  
```

The finding of type Label will be set as reported by the tool "MyReqChecker".

```

=====
= GenericPerl =
=====
  
```

The GenericPerl framework is an extension of the Generic framework that starts by running a perl script in order to generate the metrics, findings, information and links files. It is useful if you have an input file whose format needs to be converted to match the one expected by the Generic framework, or if you need to retrieve and modify information exported from a web service on your network.

```

=====
= form.xml =
=====
  
```

In your form.xml, specify the input parameters you need for your Data Provider. Our example will use two parameters: a path to a CSV file and another text parameter:

```

<?xml version="1.0" encoding="UTF-8"?>
<tags baseName="CsvPerl" needSources="false">
  <tag type="text" key="csv" defaultValue="/path/to/csv" />
  <tag type="text" key="param" defaultValue="MyValue" />
</tags>
  
```

```

=====
= config.tcl =
=====
  
```

Refer to the description of config.tcl for the Generic framework for the basic options.

Additionally, the following options are available for the GenericPerl framework, in order to know which type of information your custom Data Provider should try to import.

```
# If the data provider needs to specify a different toolName (optional)
#set SpecifyToolName 1

# Set to 1 to import metrics csv file, 0 otherwise

# ImportMetrics
# When set to 1, your custom Data Provider (CustomDP) will try to import
# metrics from a file called CustomDP.mtr.csv that your perl script
# should generate according to the expected format described in the
# documentation of the Generic framework.
set ImportMetrics 1

# ImportInfos
# When set to 1, your custom Data Provider (CustomDP) will try to import
# textual information from a file called CustomDP.inf.csv that your perl script
# should generate according to the expected format described in the
# documentation of the Generic framework.
set ImportInfos 0

# ImportFindings
# When set to 1, your custom Data Provider (CustomDP) will try to import
# findings from a file called CustomDP.fdg.csv that your perl script
# should generate according to the expected format described in the
# documentation of the Generic framework.
set ImportFindings 1

# ImportLinks
# When set to 1, your custom Data Provider (CustomDP) will try to import
# artefact links from a file called CustomDP.lnk.csv that your perl script
# should generate according to the expected format described in the
# documentation of the Generic framework.
set ImportLinks 0

# Ignore findings for artefacts that are not part of the project (orphan
findings)
# When set to 1, the findings are ignored
# When set to 0, the findings are imported and attached to the APPLICATION node
# (default: 1)
set IgnoreIfArtefactNotFound 1

# For findings of a type that is not in your ruleset, set a default rule ID.
# The value for this parameter must be a valid rule ID from your analysys model.
# (default: empty)
set UnknownRuleId UNKNOWN_RULE

# Save the total count of orphan findings as a metric at application level
# Specify the ID of the metric to use in your analysys model
# to store the information
# (default: empty)
set OrphanArteCountId NB_ORPHANS

# Save the total count of unknown rules as a metric at application level
# Specify the ID of the metric to use in your analysys model
# to store the information
# (default: empty)
set OrphanRulesCountId NB_UNKNOWN_RULES

# Save the list of unknown rule IDs as textual information at application level
```

```
# Specify the ID of the metric to use in your analysys model
# to store the information
# (default: empty)
set OrphanRulesListId UNKNOWN_RULES_INFO
```

```
=====
= CSV File Format =
=====
```

Refer to the examples in the Generic framework.

```
=====
= Perl Script =
=====
```

The perl script will receive as arguments:

- all parameters defined in form.xml (as `-${key} $value`)
- the location of the output directory where temporary files can be generated
- the full path of the metric csv file to be generated (if `ImportMetrics` is set to 1 in `config.tcl`)
- the full path of the findings csv file to be generated (if `ImportFindings` is set to 1 in `config.tcl`)
- the full path of the textual information csv file to be generated (if `ImportInfos` is set to 1 in `config.tcl`)
- the full path of the links csv file to be generated (if `ImportLinks` is set to 1 in `config.tcl`)
- the full path to the output directory used by this data provider in the previous analysis

For the `form.xml` and `config.tcl` we created earlier in this document, the command line will be:

```
perl <configuration_folder>/tools/CustomDP/CustomDP.pl -csv /path/to/csv -
param MyValue <output_folder> <output_folder>/CustomDP.mtr.csv <output_folder>/
CustomDP.fdg.csv <previous_output_folder>
```

The following perl functions are made available in the perl environment so you can use them in your script:

- `get_tag_value(key)` (returns the value for `$key` parameter from your `form.xml`)
- `get_output_metric()`
- `get_output_finding()`
- `get_output_info()`
- `get_output_link()`
- `get_output_dir()`
- `get_input_dir()` (returns the folder containing sources if `needSources` is set to 1)
- `get_previous_dir()`

Example of perl script:

```
=====
#!/usr/bin/perl
use strict;
use warnings;
$|=1 ;

# Parse input CSV file
my $csvFile = get_tag_value("csv");
my $param = get_tag_value("param");
```

```
# ...

# Write metrics to CSV
open(METRICS_FILE, ">" . get_output_metric()) || die "perl: can not write: $!
\n";
binmode(METRICS_FILE, ":utf8");
print METRICS_FILE "REQUIREMENTS;Requirements/All_Requirements;NB_REQ;15";
close METRICS_FILE;

# Write findings to CSV
open(FINDINGS_FILE, ">" . get_output_findings()) || die "perl: can not write: $!
\n";
binmode(FINDINGS_FILE, ":utf8");
print FINDINGS_FILE "REQUIREMENTS;Requirements/All_Requirements;R_LOW_REQS;
\n\"The minimum number of requirement should be at least 25.\n\"";
close FINDINGS_FILE;

exit 0;
```

```
=====
= FindingsPerl =
=====
```

The FindingsPerl framework is used to import findings and attach them to existing artefacts. Optionally, if an artefact cannot be found in your project, the finding can be attached to the root node of the project instead. When launching a Data Provider based on the FindingsPerl framework, a perl script is run first. This perl script is used to generate a CSV file with the expected format which will then be parsed by the framework.

```
=====
= form.xml =
=====
```

In your form.xml, specify the input parameters you need for your Data Provider. Our example will use two parameters: a path to a CSV file and another text parameter:

```
<?xml version="1.0" encoding="UTF-8"?>
<tags baseName="CsvPerl" needSources="true">
  <tag type="text" key="csv" defaultValue="/path/to/csv" />
  <tag type="text" key="param" defaultValue="MyValue" />
</tags>
```

- Since FindingsPerl-based data providers commonly rely on artefacts created by Squan Sources, you can set the needSources attribute to force users to specify at least one repository connector when creating a project.

```
=====
= config.tcl =
=====
```

Sample config.tcl file:

```
=====
# The separator to be used in the generated CSV file
# Usually \t or ;
```

```
set Separator ";"

# The delimiter used in the input CSV file
# This is normally left empty, except when you know that some of the values in
# the CSV file
# contain the separator itself, for example:
# "A text containing ; the separator";no problem;end
# In this case, you need to set the delimiter to \" in order for the data
# provider to find 3 values instead of 4.
# To include the delimiter itself in a value, you need to escape it by
# duplicating it, for example:
# "A text containing \" the delimiter";no problemo;end
# Default: none
set Delimiter \"

# Should the perl script executed once for each repository node of the project ?
# 1 or 0 (default)
# If true an additional parameter is sent to the
# perl script (see below for its position)
set ::NeedSources 0

# Should the violated rules definitions be generated?
# true or false (default)
# This creates a ruleset file with rules that are not already
# part of your analysis model so you can review it and add
# the rules manually if needed.
set generateRulesDefinitions false

# Should the File paths be case-insensitive?
# true or false (default)
# This is used when searching for a matching artefact in already-existing
# artefacts.
set PathsAreCaseInsensitive false

# Should file artefacts declared in the input CSV file be created automatically?
# true (default) or false
set CreateMissingFile true

# Ignore findings for artefacts that are not part of the project (orphan
# findings)
# When set to 0, the findings are imported and attached to the APPLICATION node
# instead of the real artefact
# When set to 1, the findings are not imported at all
# (default: 0)
set IgnoreIfArtefactNotFound 0

# For findings of a type that is not in your ruleset, set a default rule ID.
# The value for this parameter must be a valid rule ID from your analysis model.
# (default: empty)
set UnknownRuleId UNKNOWN_RULE

# Save the total count of orphan findings as a metric at application level
# Specify the ID of the metric to use in your analysis model
# to store the information
# (default: empty)
set OrphanArteCountId NB_ORPHANS

# Save the total count of unknown rules as a metric at application level
# Specify the ID of the metric to use in your analysis model
```

```

# to store the information
# (default: empty)
set OrphanRulesCountId NB_UNKNOWN_RULES

# Save the list of unknown rule IDs as textual information at application level
# Specify the ID of the metric to use in your analysis model
# to store the information
# (default: empty)
set OrphanRulesListId UNKNOWN_RULES_INFO

# The tool version to specify in the generated rules definitions
# The default value is ""
# Note that the toolName is the name of the folder you created
# for your custom Data Provider
set ToolVersion ""

# FileOrganisation defines the layout of the CSV file that is produced by your
perl script:
#   header::column: values are referenced from the column header
#   header::line: NOT AVAILABLE
#   alternate::line: NOT AVAILABLE
#   alternate::column: NOT AVAILABLE
set FileOrganisation header::column

# In order to attach a finding to an artefact of type FILE:
# - Tool (optional) if present it overrides the name of the tool providing the
finding
# - Path has to be the path of the file
# - Type has to be set to FILE
# - Line can be either empty or the line in the file where the finding is
located
# Rule is the rule identifier, can be used as is or translated using Rule2Key
# Descr is the description message, which can be empty
#
# In order to attach a finding to an artefact of type FUNCTION:
# - Tool (optional) if present it overrides the name of the tool providing the
finding
# - Path has to be the path of the file containing the function
# - Type has to be FUNCTION
# - If line is an integer, the system will try to find an artefact function
# at the given line of the file
# - If no Line or Line is not an integer, Name is used to find an artefact in
# the given file having name and signature as found in this column.
# (Line and Name are optional columns)

# Rule2Key contains a case-sensitive list of paired rule IDs:
#   {RuleID KeyName}
# where:
# - RuleID is the id of the rule as defined in your analysis model
# - KeyName is the rule ID as written by your perl script in the produced CSV
file
# Note: Rules that are not mapped keep their original name. The list of unmapped
rules is in the log file generated by your Data Provider.
set Rule2Key {
  { ExtractedRuleID_1 MappedRuleId_1 }
  { ExtractedRuleID_2 MappedRuleId_2 }
}

```



```

=====
= CSV File Format =
=====

According to the options defined earlier in config.tcl, a valid csv file would
be:

Path;Type;Line;Name;Rule;Descr
/src/project/module1/f1.c;FILE;12;;R1;Rule R1 is violated because variable v1
/src/project/module1/f1.c;FUNCTION;202;;R4;Rule R4 is violated because function
f1
/src/project/module2/f2.c;FUNCTION;42;;R1;Rule R1 is violated because variable v2
/src/project/module2/f2.c;FUNCTION;;skip_line(int);R1;Rule R1 is violated because
variable v2

Working With Paths:
=====

- Path separators are unified: you do not need to worry about handling
differences between Windows and Linux
- With the option PathsAreCaseInsensitive, case is ignored when searching for
files in the Squore internal data
- Paths known by Squore are relative paths starting at the root of what was
specified in the repository connector durign the analysis. This relative path is
the one used to match with a path in a csv file.

Here is a valid example of file matching:
  1. You provide C:\A\B\C\D as the root folder in a repository connector
  2. C:\A\B\C\D contains E\e.c then Squore will know E/e.c as a file

  3. You provide a csv file produced on linux and containing
     /tmp/X/Y/E/e.c as path, then Squore will be able to match it with the known
     file.

Squore uses the longest possible match.
In case of conflict, no file is found and a message is sent to the log.

=====
= Perl Script =
=====

The perl script will receive as arguments:
- all parameters defined in form.xml (as -${key} $value)
- the input directory to process (only if ::NeedSources is set to 1)
- the location of the output directory where temporary files can be generated
- the full path of the findings csv file to be generated

For the form.xml and config.tcl we created earlier in this document, the command
line will be:
perl <configuration_folder>/tools/CustomDP/CustomDP.pl -csv /path/to/csv -
param MyValue <output_folder> <output_folder>/CustomDP.fdg.csv <output_folder>/
CustomDP.fdg.csv

Example of perl script:
=====
#!/usr/bin/perl
use strict;
use warnings;

```

```

$|=1 ;

($csvKey, $csvValue, $paramKey, $paramValue, $output_folder, $output_csv) =
@ARGV;

# Parse input CSV file
# ...

# Write results to CSV
open(CSVFILE, ">" . ${output_csv}) || die "perl: can not write: $!\n";
binmode(CSVFILE, ":utf8");
print CSVFILE "Path;Type;Line;Name;Rule;Descr";
print CSVFILE "/src/project/module1/fl.c;FILE;12;;R1;Rule R1 is violated because
variable v1";
close CSVFILE;

exit 0;

```

```

=====
= ExcelMetrics =
=====

The ExcelMetrics framework is used to extract information from one or more
Microsoft Excel files (.xls or .xlsx). A detailed configuration file allows
defining how the Excel document should be read and what information should
be extracted. This framework allows importing metrics, findings and textual
information to existing artefacts or artefacts that will be created by the Data
Provider.

```

```

=====
= form.xml =
=====

```

You can customise form.xml to either:

- specify the path to a single Excel file to import
- specify a pattern to import all Excel files matching this pattern in a directory

In order to import a single Excel file:

```

=====
<?xml version="1.0" encoding="UTF-8"?>
<tags baseName="ExcelMetrics" needSources="false">
  <tag type="text" key="excel" defaultValue="/path/to/mydata.xlsx" />
</tags>

```

Notes:

- The excel key is mandatory.

In order to import all files matching a patter in a folder:

```

=====
<?xml version="1.0" encoding="UTF-8"?>
<tags baseName="ExcelMetrics" needSources="false">
  <!-- Root directory containing Excel files to import-->
  <tag type="text" key="dir" defaultValue="/path/to/mydata" />
  <!-- Pattern that needs to be matched by a file name in order to import it-->
  <tag type="text" key="ext" defaultValue="*.xlsx" />
  <!-- search for files in sub-folders -->
  <tag type="booleanChoice" defaultValue="true" key="sub" />

```

```
</tags>
```

Notes:

- The dir and ext keys are mandatory
- The sub key is optional (and its value set to false if not specified)

```
=====  
= config.tcl =  
=====
```

Sample config.tcl file:

```
=====  
# The separator to be used in the generated csv file  
# Usually \t or ; or ,  
set Separator ";"  
  
# The delimiter used in the input CSV file  
# This is normally left empty, except when you know that some of the values in  
# the CSV file  
# contain the separator itself, for example:  
# "A text containing ; the separator";no problem;end  
# In this case, you need to set the delimiter to \" in order for the data  
# provider to find 3 values instead of 4.  
# To include the delimiter itself in a value, you need to escape it by  
# duplicating it, for example:  
# "A text containing \" the delimiter";no problemo;end  
# Default: none  
set Delimiter \"  
  
# The path separator in an artefact's path  
# in the generated CSV file.  
set ArtefactPathSeparator "/"  
  
# Ignore findings for artefacts that are not part of the project (orphan  
# findings)  
# When set to 1, the findings are ignored  
# When set to 0, the findings are imported and attached to the APPLICATION node  
# (default: 1)  
set IgnoreIfArtefactNotFound 1  
  
# For findings of a type that is not in your ruleset, set a default rule ID.  
# The value for this parameter must be a valid rule ID from your analysis model.  
# (default: empty)  
set UnknownRuleId UNKNOWN_RULE  
  
# Save the total count of orphan findings as a metric at application level  
# Specify the ID of the metric to use in your analysis model  
# to store the information  
# (default: empty)  
set OrphanArteCountId NB_ORPHANS  
  
# Save the total count of unknown rules as a metric at application level  
# Specify the ID of the metric to use in your analysis model  
# to store the information  
# (default: empty)  
set OrphanRulesCountId NB_UNKNOWN_RULES
```

```

# Save the list of unknown rule IDs as textual information at application level
# Specify the ID of the metric to use in your analysys model
# to store the information
# (default: empty)
set OrphanRulesListId UNKNOWN_RULES_INFO

# The list of the Excel sheets to read, each sheet has the number of the first
line to read
# A Perl regexp pattern can be used instead of the name of the sheet (the first
sheet matching
# the pattern will be considered)
set Sheets {{Baselines 5} {ChangeNotes 5}}

# #####
# # COMMON DEFINITIONS #
# #####
#
# - <value> is a list of column specifications whose values will be concatenated.
When no column name is present, the
#     text is taken as it appears. Optional sheet name can be added (with !
char to separate from the column name)
# Examples:
#     - {C:} the value will be the value in column C on the current row
#     - {C: B:} the value will be the concatenation of values found in
column C and B of the current row
#     - {Deliveries} the value will be Deliveries
#     - {BJ: " - " BL:} the value will be the concatenation of value found
in column BJ,
#         string " - " and the value found in column BL fo the current row
#     - {OtherSheet!C:} the value will be the value in column C from the
sheet OtherSheet on the current row
#
# - <condition> is a list of conditions. An empty condition is always true. A
condition is a column name followed by colon,
#     optionally followed by a perl regexp. Optional sheet name can be
added (with ! char to separate from the column name)
# Examples:
#     - {B:} the value in column B must be empty on the current row
#     - {B:.+} the value in column B can not be empty on the current row
#     - {B:R_.+} the value in column B is a word starting by R_ on the current
row
#     - {A: B:.+ C:R_.+} the value in column A must be empty and the value in
column B must contain something and
#         the column C contains a word starting with R_ on the current row
#     - {OtherSheet!B:.+} the value in column B from sheet OtherSheet on the
current row can not be empty.

# #####
# # ARTEFACTS #
# #####
# The variable is a list of artefact hierarchy specification:
# {ArtefactHierarchySpec1 ArtefactHierarchySpec2 ... ArtefactHierarchySpecN}
# where each ArtefactHierarchySpecx is a list of ArtefactSpec
#
# An ArtefactSpec is a list of items, each item being:
# {<(sheetName!)?artefactType> <conditions> <name> <parentType>? <parentName>?}
# where:
#     - <(sheetName!)?artefactType>: allows specifying the type. Optional
sheetName can be added (with ! char to separate from the type) to limit

```

```

#           the artefact search in one specific sheet.
When Sheets are given with regexp, the same regexp has to be used
#           for the sheetName.
#           If the type is followed by a question mark
#           (?), this level of artefact is optional.
#           If the type is followed by a plus char (+),
this level is repeatable on the next row
#   - <condition>: see COMMON DEFINITIONS
#   - <value>: the name of the artefact to build, see COMMON DEFINITIONS
#
#   - <parentType>: This element is optional. When present, it means that the
current element will be attached to a parent having this type
#   - <parentValue>: This is a list like <value> to build the name of the
artefact of type <parentType>. If such artefact is not found,
#           the current artefact does not match
#
# Note: to add metrics at application level, specify an APPLICATION artefact
which will match only one line:
#   e.g. {APPLICATION {A:.+} {}} will recognize as application the line
having column A not empty.
set ArtefactsSpecs {
{
{DELIVERY {} {Deliveries}}
{RELEASE {E:.+} {E:}}
{SPRINT {O:SW_Software} {Q:}}
}
{
{DELIVERY {} {Deliveries}}
{RELEASE {O:SY_System} {Q:}}
}
{
{WP {BL:..+ AF:..+} {BJ: " - " BL:} SPRINT {AF:}}
{ChangeNotes!TASK {D:(added|changed|unchanged) T:imes} {W: AD:}}
}
{
{WP {} {{Unplanned imes}} SPRINT {AF:}}
{TASK {BL: D:(added|changed|unchanged) T:imes W:..+} {W: AD:}}
}
}
}

# #####
# # METRICS #
# #####
# Specification of metrics to be retrieved
# This is a list where each element is:
# {<artefactTypeList> <metricId> <condition> <value> <format>}
# Where:
#   - <artefactTypeList>: the list of artefact types for which the metric has
to be used
#           each element of the list is (sheetName!)?artefactType
where sheetName is used
#           to restrict search to only one sheet. sheetName is
optional.
#   - <metricId>: the name of the MeasureId to be injected into Squore, as
defined in your analysis model
#   - <condition>: see COMMON DEFINITIONS above. This is the condition for the
metric to be generated.
#   - <value> : see COMMON DEFINITIONS above. This is the value for the metric
(can be built from multi column)
    
```

```

# - <format> : optional, defaults to NUMBER
#           Possible format are:
#           * DATE_FR, DATE_EN for date stored as string
#           * DATE for cell formatted as date
#             * NUMBER_FR, NUMBER_EN for number stored as string
#           * NUMBER for cell formatted as number
#             * LINES for counting the number of text lines in a
cell
# - <formatPattern> : optional
#           Only used by the LINES format.
#           This is a pattern (can contain perl regexp) used to filter lines to count
set MetricsSpecs {
  {{RELEASE SPRINT} TIMESTAMP {} {A:} DATE_EN}
  {{RELEASE SPRINT} DATE_ACTUAL_RELEASE {} {S:} DATE_EN}
  {{RELEASE SPRINT} DATE_FINISH {} {T:} DATE_EN}
  {{RELEASE SPRINT} DELIVERY_STATUS {} {U:}}
  {{WP} WP_STATUS {} {BO:}}
  {{ChangeNotes!TASK} IS_UNPLAN {} {BL:}}
  {{TASK WP} DATE_LABEL {} {BP:} DATE_EN}
  {{TASK WP} DATE_INTEG_PLAN {} {BD:} DATE_EN}
  {{TASK} TASK_STATUS {} {AE:}}
  {{TASK} TASK_TYPE {} {AB:}}
}

# #####
# # FINDINGS #
# #####
# This is a list where each element is:
# {<artefactTypeList> <findingId> <condition> <value> <localisation>}
# Where:
# - <artefactTypeList>: the list of artefact type for which the metric has to
be used
#           each element of the list is (sheetName!)?artefactType
where sheetName is used
#           to restrict search to only one sheet. sheetName is
optional.
# - <findingId>: the name of the FindingId to be injected into Squore, as
defined in your analysis model
# - <condition>: see COMMON DEFINITIONS above. This is the condition for the
finding to be triggered.
# - <value>: see COMMON DEFINITIONS above. This is the value for the message
of the finding (can be built from multi column)
# - <localisation>: this a <value> representing the localisation of the
finding (free text)
set FindingsSpecs {
  {{WP} {BAD_WP} {BL:.+ AF:.+} {{This WP is not in a correct state } AF:.+} {A:}}
}

# #####
# # TEXTUAL INFORMATION #
# #####
# This is a list where each element is:
# {<artefactTypeList> <infoId> <condition> <value>}
# Where:
# - <artefactTypeList> the list of artefact types for which the info has to
be used
#           each element of the list is (sheetName!)?artefactType
where sheetName is used

```

```

#           to restrict search to only one sheet. sheetName is
optional.
#   - <infoId> : is the name of the Information to be attached to the artefact,
as defined in your analysis model
#   - <confition> : see COMMON DEFINITIONS above. This is the condition for the
info to be generated.
#   - <value> : see COMMON DEFINITIONS above. This is the value for the info
(can be built from multi column)
set InfosSpecs {
  {{TASK} ASSIGN_TO {} {XB:}}
}

#####
# # LABEL TRANSFORMATION #
#####
# This is a list value specification for MeasureId or InfoId:
#   <MeasureId|InfoId> { {<LABEL1> <value1>} ... {<LABELn> <valuen>}}
# Where:
#   - <MeasureId|InfoId> : is either a MeasureId, an InfoId, or * if it is
available for every measureid/infoid
#   - <LABELx> : is the label to machth (can contain perl regexp)
#   - <valuex> : is the value to replace the label by, it has to match the
correct format for the metrics (no format for infoid)
#
# Note: only metrics which are labels in the excel file or information which need
to be rewritten, need to be described here.
set Label2ValueSpec {
  {
    STATUS {
      {OPENED 0}
      {ANALYZED 1}
      {CLOSED 2}
      {.* -1}
    }
  }
  {
    * {
      {FATAL 0}
      {ERROR 1}
      {WARNING 2}
      {{LEVEL:\s*0} 1}
      {{LEVEL:\s*1} 2}
      {{LEVEL:\s*[2-9]+} 3}
    }
  }
}

```

Note that a sample Excel file with its associated config.tcl is available in \$SQUORE_HOME/addons/tools/ExcelMetrics in order to further explain available configuration options.

Appendix B. Squore XML Schemas

Download input-data-2.xsd [../shared_manual/input-data-2.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <xs:simpleType name="id">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z_][A-Z0-9_]+" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="relax-status">
<xs:restriction base="id">
  <xs:enumeration value="RELAXED_DEROGATION"/>
  <xs:enumeration value="RELAXED_LEGACY"/>
  <xs:enumeration value="RELAXED_FALSE_POSITIVE"/>
</xs:restriction>
</xs:simpleType>

  <xs:element name="bundle">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element ref="artifact"/>
        <xs:element ref="finding"/>
        <xs:element ref="link"/>
        <xs:element ref="metric"/>
      </xs:choice>
      <xs:attribute name="version" use="required" type="xs:integer" fixed="2"/>
    </xs:complexType>
  </xs:element>

  <xs:element name="artifact">
    <xs:complexType>
      <xs:sequence>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="artifact"/>
        </xs:choice>
        <xs:element ref="finding"/>
        <xs:element ref="metric"/>
        <xs:element ref="key"/>
        <xs:element ref="info"/>
        <xs:element ref="link"/>
        <xs:element ref="milestone"/>
      </xs:sequence>
      <xs:attribute name="alias"/>
      <xs:attribute name="art-location"/>
      <xs:attribute name="id"/>
      <xs:attribute name="local-art-location"/>
      <xs:attribute name="local-key"/>
      <xs:attribute name="local-parent"/>
      <xs:attribute name="location"/>
      <xs:attribute name="name"/>
      <xs:attribute name="parent"/>
      <xs:attribute name="path"/>
      <xs:attribute name="type" use="required" type="id"/>
      <xs:attribute name="view-path"/>
    </xs:complexType>
  </xs:element>

```



```
</xs:complexType>
</xs:element>

<xs:element name="info">
  <xs:complexType>
    <xs:attribute name="name" use="required" type="id"/>
    <xs:attribute name="tool"/>
    <xs:attribute name="value" use="required"/>
  </xs:complexType>
</xs:element>

<xs:element name="key">
  <xs:complexType>
    <xs:attribute name="value" use="required"/>
  </xs:complexType>
</xs:element>

<xs:element name="metric">
  <xs:complexType>
    <xs:attribute name="local-ref"/>
    <xs:attribute name="name" use="required" type="id"/>
    <xs:attribute name="ref"/>
    <xs:attribute name="tool"/>
    <xs:attribute name="value" type="xs:decimal"/>
  </xs:complexType>
</xs:element>

<xs:element name="link">
  <xs:complexType>
    <xs:attribute name="dst"/>
    <xs:attribute name="local-dst" type="xs:integer"/>
    <xs:attribute name="local-src" type="xs:integer"/>
    <xs:attribute name="name" use="required" type="id"/>
    <xs:attribute name="src"/>
  </xs:complexType>
</xs:element>

<xs:element name="finding">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="location"/>
      <xs:element minOccurs="0" maxOccurs="1" ref="relax"/>
    </xs:sequence>
    <xs:attribute name="descr"/>
    <xs:attribute name="local-ref"/>
    <xs:attribute name="location" use="required"/>
    <xs:attribute name="name" use="required" type="id"/>
    <xs:attribute name="p0"/>
    <xs:attribute name="p1"/>
    <xs:attribute name="p2"/>
    <xs:attribute name="p3"/>
    <xs:attribute name="p4"/>
    <xs:attribute name="p5"/>
    <xs:attribute name="p6"/>
    <xs:attribute name="p7"/>
    <xs:attribute name="p8"/>
    <xs:attribute name="p9"/>
    <xs:attribute name="ref"/>
    <xs:attribute name="tool"/>
  </xs:complexType>
</xs:element>
```

```

    </xs:complexType>
  </xs:element>

  <xs:element name="location">
    <xs:complexType>
      <xs:attribute name="local-ref" />
      <xs:attribute name="location" use="required" />
      <xs:attribute name="ref" />
    </xs:complexType>
  </xs:element>

  <xs:element name="relax">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="status" type="relax-status" />
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name="milestone">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="goal" />
      </xs:sequence>
      <xs:attribute name="date" type="xs:integer" />
      <xs:attribute name="name" use="required" type="id" />
    </xs:complexType>
  </xs:element>

  <xs:element name="goal">
    <xs:complexType>
      <xs:attribute name="name" use="required" type="id" />
      <xs:attribute name="value" use="required" type="xs:decimal" />
    </xs:complexType>
  </xs:element>
</xs:schema>

```

Download properties-1.2.xsd [../shared_manual/properties-1.2.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" version="1.0">

  <xs:element name="Bundle" type="bundleType" />

  <xs:complexType name="bundleType">
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="help" type="helpType" minOccurs="0" maxOccurs="unbounded" />
        <xs:element name="hideObsoleteModels" type="obsoleteType" minOccurs="0"
maxOccurs="1" />
        <xs:element name="hideModel" type="hiddenType" minOccurs="0"
maxOccurs="unbounded" />
        <xs:element name="explorerTabs" type="tabsType" minOccurs="0" maxOccurs="1" />
        <xs:element name="explorerTrees" type="treesType" minOccurs="0"
maxOccurs="1" />
        <xs:element name="option" type="optionType" minOccurs="0"
maxOccurs="unbounded" />
      </xs:choice>
    </xs:sequence>
  </xs:complexType>

```

```

    </xs:choice>
  </xs:sequence>
  <xs:attribute name="version" use="required" type="xs:string"/>
</xs:complexType>

<xs:complexType name="helpType">
  <xs:attribute name="label" use="required" type="xs:string"/>
  <xs:attribute name="url" use="required" type="xs:anyURI"/>
  <xs:attribute name="profiles" use="optional" type="xs:string"/>
</xs:complexType>

<xs:complexType name="optionType">
  <xs:attribute name="name" use="required" type="xs:string"/>
  <xs:attribute name="value" use="required" type="xs:string"/>
</xs:complexType>

<xs:complexType name="obsoleteType">
  <xs:attribute name="value" use="optional" default="false" type="xs:boolean"/>
</xs:complexType>

<xs:complexType name="hiddenType">
  <xs:attribute name="name" use="required" type="xs:string"/>
</xs:complexType>

<xs:complexType name="tabsType">
  <xs:sequence maxOccurs="unbounded">
    <xs:element name="tab" type="tabType" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="hideSettings" use="optional" default="false"
type="xs:boolean"/>
</xs:complexType>

<xs:complexType name="tabType">
  <xs:attribute name="name" use="required" type="xs:string"/>
  <xs:attribute name="default" use="optional" default="false" type="xs:boolean"/>
  <xs:attribute name="mandatory" use="optional" default="false"
type="xs:boolean"/>
  <xs:attribute name="rendered" use="optional" default="true" type="xs:boolean"/>
</xs:complexType>

<xs:complexType name="treesType">
  <xs:sequence maxOccurs="unbounded">
    <xs:element name="tree" type="treeType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="treeType">
  <xs:attribute name="name" use="required" type="xs:string"/>
  <xs:attribute name="rendered" use="optional" default="true" type="xs:boolean"/>
</xs:complexType>
</xs:schema>

```

Download config-1.2.xsd [../shared_manual/config-1.2.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" version="1.0">

  <xs:element name="square" type="squareType"/>

```

```
<xs:complexType name="squoreType">
  <xs:sequence>
    <xs:element name="paths" type="pathsType"/>
    <xs:element name="database" type="databaseType" minOccurs="0"/>
    <xs:element name="phantomjs" type="phantomjsType" minOccurs="0"/>
    <xs:element name="configuration" type="directoriesType"/>
    <xs:element name="addons" type="directoriesType"/>
    <xs:element name="client" type="dataDirectoriesType" minOccurs="0"/>
    <xs:element name="tmp" type="directoryType" minOccurs="0"/>
    <xs:element name="projects" type="projectType" minOccurs="0"/>
    <xs:element name="sources" type="directoryType" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="type" use="required" type="xs:string"/>
  <xs:attribute name="version" use="required" type="xs:string"/>
</xs:complexType>

<xs:complexType name="pathsType">
  <xs:sequence maxOccurs="unbounded">
    <xs:element name="path" type="pathType"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="pathType">
  <xs:attribute name="name" use="required" type="xs:string"/>
  <xs:attribute name="path" use="required" type="xs:string"/>
</xs:complexType>

<xs:complexType name="directoriesType">
  <xs:sequence maxOccurs="unbounded">
    <xs:element name="path" type="directoryType"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="directoryType">
  <xs:attribute name="directory" use="required" type="xs:string"/>
</xs:complexType>

<xs:complexType name="databaseType">
  <xs:sequence>
    <xs:element name="postgresql" type="directoryType" minOccurs="0"/>
    <xs:element name="cluster" type="directoryType" minOccurs="0"/>
    <xs:element name="backup" type="directoryType"/>
    <xs:element name="security" type="securityType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="phantomjsType">
  <xs:sequence>
    <xs:element name="socket-binding" type="socketBindingType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="socketBindingType">
  <xs:attribute name="address" type="xs:string" default="127.0.0.1"/>
  <xs:attribute name="port" type="xs:short" default="3003"/>
  <xs:attribute name="squore-url" type="xs:string" default=""/>
  <xs:attribute name="distant-url" type="xs:string" default=""/>
</xs:complexType>
```

```

<xs:complexType name="securityType">
  <xs:sequence>
    <xs:element name="user-name" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="dataDirectoriesType">
  <xs:sequence>
    <xs:element name="tmp" type="directoryType" minOccurs="0"/>
    <xs:element name="projects" type="projectType" minOccurs="0"/>
    <xs:element name="sources" type="directoryType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="projectType">
  <xs:sequence>
    <xs:element name="data-providers" type="dpType" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="directory" use="required" type="xs:string"/>
</xs:complexType>

<xs:complexType name="dpType">
  <xs:attribute name="keep-data-files" use="required" type="xs:boolean"/>
</xs:complexType>

</xs:schema>

```

Download analysis.xsd [../shared_manual/analysis.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <xs:simpleType name="id">
    <xs:restriction base="xs:string">
      <xs:pattern value='[A-Z_][A-Z0-9_]+' />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="list-id">
    <xs:restriction base="xs:string">
      <xs:pattern value='[A-Z_][A-Z0-9_] +( ; [A-Z_][A-Z0-9_] ) * ' />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="families">
    <xs:restriction base="xs:string">
      <xs:pattern value='[A-Z0-9_] +( ; [A-Z_][A-Z0-9_] ) * ' />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="categories">
    <xs:restriction base="xs:string">
      <xs:pattern value='[A-Z_][A-Z0-9_] + \. [A-Z_][A-Z0-9_] +( ; [A-Z_][A-Z0-9_] + \. [A-Z_][A-Z0-9_] ) * ' />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="measure-type">
    <xs:restriction base="id">

```

```

    <xs:enumeration value="METRIC" />
    <xs:enumeration value="RULE" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="format">
  <xs:restriction base="id">
    <xs:enumeration value="NUMBER" />
    <xs:enumeration value="PERCENT" />
    <xs:enumeration value="INTEGER" />
    <xs:enumeration value="DATE" />
    <xs:enumeration value="DATETIME" />
    <xs:enumeration value="TIME" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="datetime-style">
  <xs:restriction base="id">
    <xs:enumeration value="DEFAULT" />
    <xs:enumeration value="SHORT" />
    <xs:enumeration value="MEDIUM" />
    <xs:enumeration value="LONG" />
    <xs:enumeration value="FULL" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="rounding-mode">
  <xs:restriction base="id">
    <xs:enumeration value="UP" />
    <xs:enumeration value="DOWN" />
    <xs:enumeration value="CEILING" />
    <xs:enumeration value="FLOOR" />
    <xs:enumeration value="HALF_UP" />
    <xs:enumeration value="HALF_DOWN" />
    <xs:enumeration value="HALF_EVEN" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="bounds-type">
  <xs:restriction base="xs:string">
    <xs:pattern value='[\[\]]((-)*[0-9](\.[0-9]+)?)*;((-)*[0-9](\.[0-9]+)?)*[\[\]]' />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="path-scope">
  <xs:restriction base="id">
    <xs:enumeration value="CHILDREN" />
    <xs:enumeration value="DESCENDANTS" />
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="elements">
  <xs:sequence>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="ArtefactType" />
      <xs:element ref="Indicator" />
      <xs:element ref="Measure" />
      <xs:element ref="Package" />
    </xs:choice>
  </xs:sequence>
</xs:complexType>

```

```

<xs:element ref="package" />
<xs:element ref="Scale" />
<xs:element ref="ScaleMacro" />
<xs:element ref="Constant" />
<xs:element ref="RootIndicator" />
<xs:element ref="UpdateRules" />
<xs:element ref="UpdateRule" />
<xs:element ref="Link" />
<xs:element ref="ComputedLink" />
</xs:choice>
</xs:sequence>
<xs:attribute name="providedBy" />
</xs:complexType>

<xs:element name="Bundle" type="elements" />

<xs:element name="Package" type="elements" />
<xs:element name="package" type="elements" />

<xs:element name="Constant">
  <xs:complexType>
    <xs:attribute name="id" use="required" type="id" />
    <xs:attribute name="value" use="required" />
  </xs:complexType>
</xs:element>

<xs:element name="RootIndicator">
  <xs:complexType>
    <xs:attribute name="artefactTypes" use="required" type="list-id" />
    <xs:attribute name="indicatorId" use="required" type="id" />
  </xs:complexType>
</xs:element>

<xs:element name="UpdateRules">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="UpdateRule" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="UpdateRule">
  <xs:complexType>
    <xs:attribute name="categories" type="categories" />
    <xs:attribute name="disabled" type="xs:boolean" />
    <xs:attribute name="measureId" use="required" type="id" />
  </xs:complexType>
</xs:element>

<xs:element name="Measure">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="Computation" />
    </xs:sequence>
    <xs:attribute name="acceptMissingValue" type="xs:boolean" />
    <xs:attribute name="categories" type="categories" />
    <xs:attribute name="dataBounds" type="bounds-type" />
    <xs:attribute name="dateStyle" type="datetime-style" />
    <xs:attribute name="decimals" type="xs:integer" />
  </xs:complexType>
</xs:element>

```

```

<xs:attribute name="defaultValue" type="xs:decimal"/>
<xs:attribute name="excludingTypes" type="list-id"/>
<xs:attribute name="invalidValue"/>
<xs:attribute name="families" type="families"/>
<xs:attribute name="format" type="format"/>
<xs:attribute name="manual" type="xs:boolean"/>
<xs:attribute name="measureId" use="required" type="id"/>
<xs:attribute name="noValue"/>
<xs:attribute name="pattern"/>
<xs:attribute name="roundingMode" type="rounding-mode"/>
<xs:attribute name="suffix"/>
<xs:attribute name="targetArtefactTypes"/>
<xs:attribute name="timeStyle" type="datetime-style"/>
<xs:attribute name="toolName"/>
<xs:attribute name="toolVersion"/>
<xs:attribute name="type" type="measure-type"/>
<xs:attribute name="usedForRelaxation" type="xs:boolean"/>
</xs:complexType>
</xs:element>

<xs:element name="Computation">
  <xs:complexType>
    <xs:attribute name="continueOnRelaxed" type="xs:boolean"/>
    <xs:attribute name="excludingTypes" type="list-id"/>
    <xs:attribute name="result" use="required"/>
    <xs:attribute name="stored" type="xs:boolean"/>
    <xs:attribute name="targetArtefactTypes" use="required" type="list-id"/>
  </xs:complexType>
</xs:element>

<xs:element name="Indicator">
  <xs:complexType>
    <xs:attribute name="displayedScale" type="id"/>
    <xs:attribute name="displayedValue" type="id"/>
    <xs:attribute name="displayTypes" type="list-id"/>
    <xs:attribute name="excludingTypes" type="list-id"/>
    <xs:attribute name="families" type="families"/>
    <xs:attribute name="indicatorId" use="required" type="id"/>
    <xs:attribute name="measureId" type="id"/>
    <xs:attribute name="scaleId" type="id"/>
    <xs:attribute name="targetArtefactTypes" type="list-id"/>
  </xs:complexType>
</xs:element>

<xs:element name="Scale">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="ScaleLevel"/>
    </xs:sequence>
    <xs:attribute name="isDynamic" type="xs:boolean"/>
    <xs:attribute name="macro" type="id"/>
    <xs:attribute name="scaleId" use="required" type="id"/>
    <xs:attribute name="targetArtefactTypes" type="list-id"/>
    <xs:attribute name="vars"/>
  </xs:complexType>
</xs:element>

<xs:element name="ScaleMacro">
  <xs:complexType>

```



```

<xs:sequence>
  <xs:element maxOccurs="unbounded" ref="ScaleLevel"/>
</xs:sequence>
<xs:attribute name="id" use="required" type="id"/>
<xs:attribute name="isDynamic" type="xs:boolean"/>
</xs:complexType>
</xs:element>

<xs:element name="ArtefactType">
  <xs:complexType>
    <xs:attribute name="heirs" type="list-id"/>
    <xs:attribute name="id" use="required" type="id"/>
    <xs:attribute name="manual" type="xs:boolean"/>
    <xs:attribute name="parents" type="list-id"/>
  </xs:complexType>
</xs:element>

<xs:element name="ScaleLevel">
  <xs:complexType>
    <xs:attribute name="bounds" use="required"/>
    <xs:attribute name="levelId" use="required" type="id"/>
    <xs:attribute name="rank" use="required"/>
  </xs:complexType>
</xs:element>

<xs:element name="Link">
  <xs:complexType>
    <xs:attribute name="id" use="required" type="id"/>
    <xs:attribute name="inArtefactTypes" type="list-id"/>
    <xs:attribute name="outArtefactTypes" type="list-id"/>
    <xs:attribute name="srcArtefactTypes" type="list-id"/>
    <xs:attribute name="dstArtefactTypes" type="list-id"/>
  </xs:complexType>
</xs:element>

<xs:element name="ComputedLink">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" ref="StartPath"/>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="NextPath"/>
    </xs:sequence>
    <xs:attribute name="id" use="required" type="id"/>
  </xs:complexType>
</xs:element>

<xs:element name="StartPath">
  <xs:complexType>
    <xs:attribute name="link" type="id"/>
    <xs:attribute name="scope" type="path-scope"/>
    <xs:attribute name="srcArtefactTypes" type="list-id"/>
    <xs:attribute name="dstArtefactTypes" type="list-id"/>
    <xs:attribute name="srcCondition"/>
    <xs:attribute name="dstCondition"/>
    <xs:attribute name="recurse" type="xs:boolean"/>
    <xs:attribute name="keepIntermediateLinks" type="xs:boolean"/>
  </xs:complexType>
</xs:element>

<xs:element name="NextPath">

```

```

<xs:complexType>
  <xs:attribute name="link" type="id"/>
  <xs:attribute name="scope" type="path-scope"/>
  <xs:attribute name="dstArtefactTypes" type="list-id"/>
  <xs:attribute name="dstCondition"/>
  <xs:attribute name="recurse" type="xs:boolean"/>
  <xs:attribute name="keepIntermediateLinks" type="xs:boolean"/>
</xs:complexType>
</xs:element>
</xs:schema>

```

Download decision.xsd [../shared_manual/decision.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <xs:simpleType name="id">
    <xs:restriction base="xs:string">
      <xs:pattern value='[A-Z_][A-Z0-9_]+' />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="list-id">
    <xs:restriction base="xs:string">
      <xs:pattern value='[A-Z_][A-Z0-9_]+(;[A-Z_][A-Z0-9_]+)*' />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="categories">
    <xs:restriction base="xs:string">
      <xs:pattern value='[A-Z_][A-Z0-9_]+\.[A-Z_][A-Z0-9_]+(;[A-Z_][A-Z0-9_]+\.[A-Z_][A-Z0-9_]+)*' />
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="elements">
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="Package"/>
        <xs:element ref="package"/>
        <xs:element ref="DecisionCriteria"/>
        <xs:element ref="DecisionCriterion"/>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>

  <xs:element name="Bundle" type="elements"/>
  <xs:element name="Package" type="elements"/>
  <xs:element name="package" type="elements"/>
  <xs:element name="DecisionCriteria" type="elements"/>

  <xs:element name="DecisionCriterion">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Triggers"/>
      </xs:sequence>
      <xs:attribute name="categories" type="categories"/>
      <xs:attribute name="dcId" use="required" type="id"/>
      <xs:attribute name="excludingTypes" type="list-id"/>
    </xs:complexType>
  </xs:element>

```

```

<xs:attribute name="families" type="list-id"/>
<xs:attribute name="roles" type="list-id"/>
<xs:attribute name="targetArtefactTypes" use="required" type="list-id"/>
</xs:complexType>
</xs:element>

<xs:element name="Triggers">
<xs:complexType>
<xs:sequence>
<xs:element maxOccurs="unbounded" ref="Trigger"/>
</xs:sequence>
</xs:complexType>
</xs:element>

<xs:element name="Trigger">
<xs:complexType>
<xs:sequence>
<xs:element maxOccurs="unbounded" ref="Test"/>
</xs:sequence>
</xs:complexType>
</xs:element>

<xs:element name="Test">
<xs:complexType>
<xs:attribute name="bounds" use="required"/>
<xs:attribute name="descrId" type="id"/>
<xs:attribute name="expr" use="required"/>
<xs:attribute name="p0"/>
<xs:attribute name="p1"/>
<xs:attribute name="p2"/>
<xs:attribute name="p3"/>
<xs:attribute name="p4"/>
<xs:attribute name="p5"/>
<xs:attribute name="p6"/>
<xs:attribute name="p7"/>
<xs:attribute name="p8"/>
<xs:attribute name="p9"/>
<xs:attribute name="suspect"/>
</xs:complexType>
</xs:element>
</xs:schema>

```

Download description.xsd [../shared_manual/description.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">

<xs:complexType name="elements">
<xs:sequence>
<xs:choice minOccurs="0" maxOccurs="unbounded">
<xs:element ref="Package"/>
<xs:element ref="package"/>
<xs:element ref="Properties"/>
</xs:choice>
</xs:sequence>
</xs:complexType>

<xs:element name="Bundle">
<xs:complexType>

```

```

<xs:sequence>
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="Package" />
    <xs:element ref="package" />
    <xs:element ref="Properties" />
  </xs:choice>
</xs:sequence>
<xs:attribute name="available" />
<xs:attribute name="default" />
</xs:complexType>
</xs:element>

<xs:element name="Package" type="elements" />
<xs:element name="package" type="elements" />

<xs:element name="Properties">
  <xs:complexType>
    <xs:attribute name="src" use="required" />
  </xs:complexType>
</xs:element>
</xs:schema>

```

Download exports.xsd [../shared_manual/exports.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <xs:simpleType name="type-id">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z0-9_]*" />
    </xs:restriction>
  </xs:simpleType>

  <xs:element name="Bundle">
    <xs:complexType>
      <xs:sequence maxOccurs="unbounded">
        <xs:element ref="Role" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="Role">
    <xs:complexType>
      <xs:sequence maxOccurs="unbounded">
        <xs:element ref="Export" />
      </xs:sequence>
      <xs:attribute name="name" use="required" type="xs:string" />
    </xs:complexType>
  </xs:element>

  <xs:element name="Export">
    <xs:complexType>
      <xs:sequence maxOccurs="unbounded">
        <xs:element ref="ExportScript" />
      </xs:sequence>
      <xs:attribute name="type" use="required" type="type-id" />
    </xs:complexType>
  </xs:element>

```

```

<xs:element name="ExportScript">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="arg" />
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string" />
    <xs:attribute name="script" use="required" type="xs:string" />
  </xs:complexType>
</xs:element>

<xs:element name="arg">
  <xs:complexType>
    <xs:attribute name="value" use="required" type="xs:string" />
  </xs:complexType>
</xs:element>

</xs:schema>

```

Download highlights.xsd [../shared_manual/highlights.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:simpleType name="id">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z0-9_]*" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="measure-id">
    <xs:restriction base="xs:string">
      <xs:pattern value="([BD].)?[A-Z0-9_]*" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="info-id">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z0-9_]*" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="indicator-id">
    <xs:restriction base="xs:string">
      <xs:pattern value="([I].)?[A-Z0-9_]*" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="bounds-type">
    <xs:restriction base="xs:string">
      <xs:pattern value="[[\]]((-)*[0-9](\[0-9\]+)?)*;((-)*[0-9](\[0-9\]+)?)*[[\]]" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="top-order">
    <xs:restriction base="xs:string">
      <xs:enumeration value="ASC" />
      <xs:enumeration value="DESC" />
    </xs:restriction>
  </xs:simpleType>

```

```
<xs:simpleType name="result-size">
  <xs:union>
    <xs:simpleType>
      <xs:restriction base="xs:positiveInteger" />
    </xs:simpleType>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="*" />
      </xs:restriction>
    </xs:simpleType>
  </xs:union>
</xs:simpleType>

<xs:simpleType name="header-display-type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="MNEMONIC" />
    <xs:enumeration value="NAME" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="display-type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="VALUE" />
    <xs:enumeration value="RANK" />
    <xs:enumeration value="ICON" />
    <xs:enumeration value="DATE" />
    <xs:enumeration value="DATETIME" />
    <xs:enumeration value="TIME" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="date-style">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SHORT" />
    <xs:enumeration value="MEDIUM" />
    <xs:enumeration value="DEFAULT" />
    <xs:enumeration value="LONG" />
    <xs:enumeration value="FULL" />
  </xs:restriction>
</xs:simpleType>

<xs:element name="Bundle">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:element ref="Role" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="Role">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:element ref="Filters" />
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string" />
    <xs:attribute name="preSelectedType" use="optional" type="xs:string" />
  </xs:complexType>
</xs:element>
```

```

<xs:element name="Filters">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:choice>
        <xs:element ref="TopArtefacts" />
        <xs:element name="TopDeltaArtefacts" type="top-artefacts" />
        <xs:element ref="TopBorderlineArtefacts" />
        <xs:element name="TopNewArtefacts" type="top-artefacts" />
        <xs:element ref="TopModifiedArtefacts" />
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="type" use="required" type="xs:string" />
  </xs:complexType>
</xs:element>

<xs:element name="TopArtefacts">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:choice>
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="Column" />
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="Where" />
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="OrderBy" />
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="id" use="required" type="id" />
    <xs:attribute name="name" use="optional" type="xs:string" />
    <xs:attribute name="artefactTypes" use="optional" type="xs:string" />
    <xs:attribute name="excludingTypes" use="optional" type="xs:string" />
    <xs:attribute name="measureId" use="optional" default="LEVEL" type="measure-
id" />
    <xs:attribute name="order" use="optional" default="ASC" type="top-order" />
    <xs:attribute name="altMeasureId" use="optional" type="measure-id" />
    <xs:attribute name="altOrder" use="optional" type="top-order" />
    <xs:attribute name="resultSize" use="required" type="result-size" />
  </xs:complexType>
</xs:element>

<xs:element name="TopBorderlineArtefacts">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:choice>
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="Column" />
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="Where" />
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="OrderBy" />
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="id" use="required" type="id" />
    <xs:attribute name="name" use="optional" type="xs:string" />
    <xs:attribute name="artefactTypes" use="optional" type="xs:string" />
    <xs:attribute name="excludingTypes" use="optional" type="xs:string" />
    <xs:attribute name="indicatorId" use="optional" default="LEVEL"
type="indicator-id" />
    <xs:attribute name="minLevel" use="optional" type="xs:string" />
    <xs:attribute name="order" use="optional" default="ASC" type="top-order" />
    <xs:attribute name="resultSize" use="required" type="result-size" />
  </xs:complexType>
</xs:element>

```

```

<xs:element name="TopModifiedArtefacts">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:choice>
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="Column" />
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="Where" />
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="OrderBy" />
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="id" use="required" type="id" />
    <xs:attribute name="name" use="optional" type="xs:string" />
    <xs:attribute name="artefactTypes" use="optional" type="xs:string" />
    <xs:attribute name="excludingTypes" use="optional" type="xs:string" />
    <xs:attribute name="order" use="optional" default="ASC" type="top-order" />
    <xs:attribute name="resultSize" use="required" type="result-size" />
  </xs:complexType>
</xs:element>

<xs:element name="Column">
  <xs:complexType>
    <xs:attribute name="measureId" use="optional" type="measure-id" />
    <xs:attribute name="infoId" use="optional" type="info-id" />
    <xs:attribute name="indicatorId" use="optional" type="indicator-id" />
    <xs:attribute name="artefactTypes" use="optional" type="xs:string" />
    <xs:attribute name="excludingTypes" use="optional" type="xs:string" />
    <xs:attribute name="headerDisplayType" use="optional" default="NAME"
type="header-display-type" />
    <xs:attribute name="displayType" use="optional" default="VALUE" type="display-
type" />
    <xs:attribute name="decimals" use="optional" default="2" type="xs:integer" />
    <xs:attribute name="dateStyle" use="optional" default="DEFAULT" type="date-
style" />
    <xs:attribute name="timeStyle" use="optional" default="DEFAULT" type="date-
style" />
    <xs:attribute name="datePattern" use="optional" type="xs:string" />
    <xs:attribute name="suffix" use="optional" type="xs:string" />
  </xs:complexType>
</xs:element>

<xs:element name="Where">
  <xs:complexType>
    <xs:attribute name="measureId" use="optional" type="measure-id" />
    <xs:attribute name="infoId" use="optional" type="info-id" />
    <xs:attribute name="value" use="optional" type="xs:string" />
    <xs:attribute name="bounds" use="optional" type="bounds-type" />
  </xs:complexType>
</xs:element>

<xs:element name="OrderBy">
  <xs:complexType>
    <xs:attribute name="measureId" use="required" type="measure-id" />
    <xs:attribute name="order" use="optional" default="ASC" type="top-order" />
  </xs:complexType>
</xs:element>

<xs:complexType name="top-artefacts">
  <xs:sequence maxOccurs="unbounded">
    <xs:choice>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="Column" />

```



```

    <xs:element minOccurs="0" maxOccurs="unbounded" ref="Where" />
    <xs:element minOccurs="0" maxOccurs="unbounded" ref="OrderBy" />
  </xs:choice>
</xs:sequence>
<xs:attribute name="id" use="required" type="id" />
<xs:attribute name="name" use="optional" type="xs:string" />
<xs:attribute name="artefactTypes" use="optional" type="xs:string" />
<xs:attribute name="excludingTypes" use="optional" type="xs:string" />
<xs:attribute name="measureId" use="optional" default="LEVEL" type="measure-
id" />
<xs:attribute name="order" use="optional" default="ASC" type="top-order" />
<xs:attribute name="resultSize" use="required" type="result-size" />
</xs:complexType>
</xs:schema>

```

Download properties.xsd [../shared_manual/properties.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <xs:simpleType name="id">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z_][A-Z0-9_]+" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="list-id">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z_][A-Z0-9_]+(;[A-Z_][A-Z0-9_]+)*" />
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="elements">
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="package" />
        <xs:element ref="hideMeasure" />
        <xs:element ref="findingsTab" />
        <xs:element ref="actionItemsTab" />
        <xs:element ref="rulesEdition" />
      </xs:choice>
    </xs:sequence>
  </xs:complexType>

  <xs:element name="bundle" type="elements"/>
  <xs:element name="package" type="elements"/>

  <xs:element name="hideMeasure">
    <xs:complexType>
      <xs:attribute name="path" use="required" />
      <xs:attribute name="targetArtefactTypes" type="list-id" />
    </xs:complexType>
  </xs:element>

  <xs:element name="findingsTab">
    <xs:complexType>
      <xs:attribute name="orderBy" type="list-id" />
      <xs:attribute name="hideColumns" type="list-id" />
    </xs:complexType>
  </xs:element>

```

```

    <xs:attribute name="artefactFilters" />
  </xs:complexType>
</xs:element>

<xs:element name="actionItemsTab">
  <xs:complexType>
    <xs:attribute name="orderBy" type="list-id" />
    <xs:attribute name="hideColumns" type="list-id" />
  </xs:complexType>
</xs:element>

<xs:element name="rulesEdition">
  <xs:complexType>
    <xs:attribute name="scales" use="required" type="list-id" />
  </xs:complexType>
</xs:element>
</xs:schema>

```

Download tutorials.xsd [../shared_manual/tutorials.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:simpleType name="external-id">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z]{1}[A-Z0-9_]*" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="positive-integer">
    <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]+" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="opacity">
    <xs:restriction base="xs:string">
      <xs:pattern value="(0|1){1}\.?[0-9]{0,2}" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="actions">
    <xs:restriction base="xs:string">
      <xs:enumeration value="EXPAND_PORTFOLIO_TREE" />
      <xs:enumeration value="EXPAND_ARTEFACT_TREE" />
      <xs:enumeration value="EXPAND_MEASURE_TREE" />
      <xs:enumeration value="COLLAPSE_PORTFOLIO_TREE" />
      <xs:enumeration value="COLLAPSE_ARTEFACT_TREE" />
      <xs:enumeration value="COLLAPSE_MEASURE_TREE" />
      <xs:enumeration value="SELECT_MODEL" />
      <xs:enumeration value="SELECT_PROJECT" />
      <xs:enumeration value="SELECT_ARTEFACT" />
      <xs:enumeration value="SELECT_ARTEFACT_LEAF" />
      <xs:enumeration value="CLOSE_MEASURE_POPUP" />
      <xs:enumeration value="SELECT_MEASURE" />
      <xs:enumeration value="SHOW_REVIEW_SET" />
      <xs:enumeration value="SHOW_PORTFOLIO_TREE" />
      <xs:enumeration value="SHOW_DASHBOARD_TAB" />
      <xs:enumeration value="SHOW_ACTION_ITEMS_TAB" />
      <xs:enumeration value="SHOW_HIGHLIGHTS_TAB" />
    </xs:restriction>
  </xs:simpleType>

```

```
<xs:enumeration value="SHOW_FINDINGS_TAB" />
<xs:enumeration value="SHOW_REPORTS_TAB" />
<xs:enumeration value="SHOW_FORMS_TAB" />
<xs:enumeration value="SHOW_INDICATORS_TAB" />
<xs:enumeration value="SHOW_MEASURES_TAB" />
<xs:enumeration value="SHOW_COMMENTS_TAB" />
<xs:enumeration value="SHOW_ACTION_ITEMS_ADVANCED_SEARCH" />
<xs:enumeration value="EXPAND_ACTION_ITEM" />
<xs:enumeration value="SHOW_FINDINGS_ADVANCED_SEARCH" />
<xs:enumeration value="SELECT_FINDING" />
<xs:enumeration value="SELECT_FINDING_ARTEFACT" />
<xs:enumeration value="EXPAND_FINDING" />
<xs:enumeration value="EXPAND_ATTRIBUTE" />
<xs:enumeration value="SWITCH_INDICATORS_PAGE" />
<xs:enumeration value="SWITCH_MEASURES_PAGE" />
<xs:enumeration value="SWITCH_COMMENTS_PAGE" />
<xs:enumeration value="CLOSE_CHART_POPUP" />
<xs:enumeration value="OPEN_CHART_POPUP" />
<xs:enumeration value="OPEN_MODEL_CHART_POPUP" />
<xs:enumeration value="SELECT_DESCR_TAB" />
<xs:enumeration value="SELECT_COMMENTS_TAB" />
<xs:enumeration value="SELECT_FAVORITES_TAB" />
<xs:enumeration value="COMPARE_CHART" />
<xs:enumeration value="QUIT_COMPARATIVE_MODE" />
<xs:enumeration value="QUIT_FULLDISPLAY_MODE" />
<xs:enumeration value="CLOSE_ARTEFACT_TREE_FILTER" />
<xs:enumeration value="SHOW_ARTEFACT_TREE_FILTER" />
<xs:enumeration value="OPEN_TABLE" />
<xs:enumeration value="CHANGE_PAGE" />
<xs:enumeration value="CREATE_NEW_PROJECT" />
<xs:enumeration value="SELECT_WIZARD" />
<xs:enumeration value="VALIDATE_WIZARD" />
<xs:enumeration value="VALIDATE_INFORMATION" />
<xs:enumeration value="VALIDATE_DP_OPTIONS" />
<xs:enumeration value="RUN_PROJECT_CREATION" />
<xs:enumeration value="OPEN_SUB_MENU_HELP" />
<xs:enumeration value="CLOSE_TUTORIAL_POPUP" />
<xs:enumeration value="OPEN_TUTORIAL_POPUP" />
<xs:enumeration value="NONE" />
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="alias">
  <xs:restriction base="xs:string">
    <xs:enumeration value="CUSTOM" />
    <xs:enumeration value="BODY" />
    <xs:enumeration value="BREADCRUMBS" />
    <xs:enumeration value="MENU_HELP" />
    <xs:enumeration value="SUB_MENU_HELP" />
    <xs:enumeration value="SUB_MENU_HELP_ROW" />
    <xs:enumeration value="SUB_MENU_HELP_ROW_FIRST" />
    <xs:enumeration value="TUTORIAL_POPUP" />
    <xs:enumeration value="TUTORIAL_POPUP_MODEL" />
    <xs:enumeration value="TUTORIAL_POPUP_MODEL_FIRST" />
    <xs:enumeration value="TUTORIAL_POPUP_TUTORIAL_NAME" />
    <xs:enumeration value="TUTORIAL_POPUP_TUTORIAL_NAME_FIRST" />
    <xs:enumeration value="TUTORIAL_POPUP_TUTORIAL_DESCR" />
    <xs:enumeration value="TUTORIAL_POPUP_TUTORIAL_DESCR_FIRST" />
    <xs:enumeration value="EXPLORER" />
  </xs:restriction>
</xs:simpleType>
```

```
<xs:enumeration value="DRILLDOWN" />
<xs:enumeration value="EXPLORER_TAB" />
<xs:enumeration value="ARTEFACT_TREE" />
<xs:enumeration value="MEASURE_TREE" />
<xs:enumeration value="EXPLORER_HEADER" />
<xs:enumeration value="PORTFOLIO_HEADER" />
<xs:enumeration value="ARTEFACT_TREE_SEARCH" />
<xs:enumeration value="ARTEFACT_TREE_FILTER" />
<xs:enumeration value="REVIEW_SET" />
<xs:enumeration value="PORTFOLIO_TREE" />
<xs:enumeration value="PORTFOLIO_TREE_PROJECT" />
<xs:enumeration value="PORTFOLIO_TREE_PROJECT_FIRST" />
<xs:enumeration value="MODEL_DASHBOARD" />
<xs:enumeration value="MODEL_CHARTS" />
<xs:enumeration value="MODEL_CHART_FIRST" />
<xs:enumeration value="MODEL_TABLE" />
<xs:enumeration value="MODEL_TABLE_ROW_FIRST" />
<xs:enumeration value="MODEL_CHART" />
<xs:enumeration value="MODEL_TABLE_ROW" />
<xs:enumeration value="MODEL_CHART_POPUP" />
<xs:enumeration value="MODEL_CHART_POPUP_GRAPH" />
<xs:enumeration value="MODEL_CHART_POPUP_PREVIOUS_ARROW" />
<xs:enumeration value="MODEL_CHART_POPUP_NEXT_ARROW" />
<xs:enumeration value="MODEL_CHART_POPUP_NAV_BAR" />
<xs:enumeration value="MODEL_CHART_POPUP_ASIDE" />
<xs:enumeration value="MODEL_CHART_POPUP_ASIDE_HEAD" />
<xs:enumeration value="MODEL_CHART_POPUP_DESCR" />
<xs:enumeration value="FILTER_POPUP" />
<xs:enumeration value="FILTER_LEVEL" />
<xs:enumeration value="FILTER_TYPE" />
<xs:enumeration value="FILTER_EVOLUTION" />
<xs:enumeration value="FILTER_STATUS" />
<xs:enumeration value="ARTEFACT_TREE_LEAF" />
<xs:enumeration value="MEASURE_TREE_LEAF" />
<xs:enumeration value="MENU_INDICATOR_ARTEFACT" />
<xs:enumeration value="DASHBOARD" />
<xs:enumeration value="SCORECARD" />
<xs:enumeration value="KPI" />
<xs:enumeration value="CHARTS" />
<xs:enumeration value="TABLES" />
<xs:enumeration value="CHART_FIRST" />
<xs:enumeration value="LINE" />
<xs:enumeration value="CHART" />
<xs:enumeration value="CHART_FIRST" />
<xs:enumeration value="TABLE" />
<xs:enumeration value="TABLE_FIRST" />
<xs:enumeration value="MEASURE_POPUP" />
<xs:enumeration value="MEASURE_POPUP_CONTENT" />
<xs:enumeration value="MEASURE_POPUP_LEVELS" />
<xs:enumeration value="MEASURE_POPUP_ROW_FIRST" />
<xs:enumeration value="MEASURE_POPUP_ROW" />
<xs:enumeration value="CHART_POPUP" />
<xs:enumeration value="CHART_POPUP_GRAPH" />
<xs:enumeration value="CHART_POPUP_COMPARE_OPTION" />
<xs:enumeration value="CHART_POPUP_PREVIOUS_ARROW" />
<xs:enumeration value="CHART_POPUP_NEXT_ARROW" />
<xs:enumeration value="CHART_POPUP_NAV_BAR" />
<xs:enumeration value="CHART_POPUP_ASIDE" />
<xs:enumeration value="CHART_POPUP_ASIDE_HEAD" />
```

```
<xs:enumeration value="CHART_POPUP_DESCR" />
<xs:enumeration value="CHART_POPUP_COMMENTS" />
<xs:enumeration value="CHART_POPUP_FAVORITES" />
<xs:enumeration value="CHART_POPUP_COMPARATIVE_CHART" />
<xs:enumeration value="ACTION_ITEMS" />
<xs:enumeration value="ACTION_ITEMS_TABLE" />
<xs:enumeration value="ACTION_ITEMS_TABLE_HEAD" />
<xs:enumeration value="ACTION_ITEMS_TABLE_HEAD_CHECK" />
<xs:enumeration value="ACTION_ITEMS_ADD_REVIEW_SET" />
<xs:enumeration value="ACTION_ITEMS_EXPORT_LIST" />
<xs:enumeration value="ACTION_ITEMS_EXPORT_BUTTON" />
<xs:enumeration value="ACTION_ITEMS_SEARCH" />
<xs:enumeration value="ACTION_ITEMS_ROW" />
<xs:enumeration value="ACTION_ITEMS_REASON" />
<xs:enumeration value="ACTION_ITEMS_ADVANCED_SEARCH" />
<xs:enumeration value="ACTION_ITEMS_ADVANCED_SEARCH_SELECT_FIRST" />
<xs:enumeration value="ACTION_ITEMS_ADVANCED_SEARCH_SELECT" />
<xs:enumeration value="HIGHLIGHTS" />
<xs:enumeration value="HIGHLIGHTS_TABLE" />
<xs:enumeration value="HIGHLIGHTS_TABLE_HEAD" />
<xs:enumeration value="HIGHLIGHTS_TABLE_HEAD_CHECK" />
<xs:enumeration value="HIGHLIGHTS_SEARCH" />
<xs:enumeration value="HIGHLIGHTS_SEARCH_FILTER" />
<xs:enumeration value="HIGHLIGHTS_SEARCH_TYPE" />
<xs:enumeration value="HIGHLIGHTS_EXPORT_BUTTON" />
<xs:enumeration value="HIGHLIGHTS_ADD_REVIEW_SET" />
<xs:enumeration value="HIGHLIGHTS_ROW_FIRST" />
<xs:enumeration value="FINDINGS" />
<xs:enumeration value="FINDINGS_TABLE" />
<xs:enumeration value="FINDINGS_TABLE_HEAD" />
<xs:enumeration value="FINDINGS_SEARCH" />
<xs:enumeration value="FINDINGS_INFO" />
<xs:enumeration value="FINDINGS_RULE" />
<xs:enumeration value="FINDINGS_ARTEFACT" />
<xs:enumeration value="FINDINGS_ROW_FIRST" />
<xs:enumeration value="FINDINGS_ADVANCED_SEARCH" />
<xs:enumeration value="FINDINGS_ADVANCED_SEARCH_SELECT_FIRST" />
<xs:enumeration value="FINDINGS_ADVANCED_SEARCH_SELECT" />
<xs:enumeration value="REPORTS" />
<xs:enumeration value="REPORTS_REGION" />
<xs:enumeration value="REPORTS_OPTIONS" />
<xs:enumeration value="REPORTS_OPTION_TEMPLATE" />
<xs:enumeration value="REPORTS_OPTION_FORMAT" />
<xs:enumeration value="REPORTS_OPTION_SYNTHETIC_VIEW" />
<xs:enumeration value="REPORTS_CREATE" />
<xs:enumeration value="EXPORT_REGION" />
<xs:enumeration value="EXPORT_OPTIONS" />
<xs:enumeration value="EXPORT_CREATE" />
<xs:enumeration value="FORMS" />
<xs:enumeration value="FORMS_ATTRIBUTE" />
<xs:enumeration value="FORMS_ATTRIBUTE_FIELD" />
<xs:enumeration value="FORMS_ATTRIBUTE_COMMENT" />
<xs:enumeration value="FORMS_HISTORY" />
<xs:enumeration value="FORMS_BLOCK" />
<xs:enumeration value="INDICATORS" />
<xs:enumeration value="INDICATORS_TABLE" />
<xs:enumeration value="INDICATORS_TABLE_HEAD" />
<xs:enumeration value="INDICATORS_ROW" />
<xs:enumeration value="MEASURES" />
```

```

<xs:enumeration value="MEASURES_TABLE" />
<xs:enumeration value="MEASURES_TABLE_HEAD" />
<xs:enumeration value="MEASURES_ROW" />
<xs:enumeration value="COMMENTS" />
<xs:enumeration value="COMMENTS_TABLE" />
<xs:enumeration value="COMMENTS_TABLE_HEAD" />
<xs:enumeration value="COMMENTS_ROW" />
<xs:enumeration value="CREATE_PROJECT_BUTTON" />
<xs:enumeration value="WIZARD_PANEL" />
<xs:enumeration value="WIZARD_ROW" />
<xs:enumeration value="WIZARD_ROW_FIRST" />
<xs:enumeration value="WIZARD_NEXT_BUTTON" />
<xs:enumeration value="GENERAL_INFORMATION" />
<xs:enumeration value="PROJECT_IDENTIFICATION_BLOCK " />
<xs:enumeration value="GENERAL_INFO_BLOCK" />
<xs:enumeration value="GENERAL_INFO_ROW" />
<xs:enumeration value="PROJECT_NEXT_BUTTON" />
<xs:enumeration value="DP_PANEL" />
<xs:enumeration value="DP_PANEL_BLOCK" />
<xs:enumeration value="DP_PANEL_ROW" />
<xs:enumeration value="DP_PANEL_NEXT_BUTTON" />
<xs:enumeration value="CONFIRMATION_PANEL" />
<xs:enumeration value="SUMMARY" />
<xs:enumeration value="CONFIRMATION_PANEL_PARAMETERS" />
<xs:enumeration value="RUN_NEW_PROJECT_BUTTON" />
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="colors">
  <xs:union>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:pattern value="#"[A-Fa-f0-9]{6}" />
      </xs:restriction>
    </xs:simpleType>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:pattern value="(rgb|RGB)\([0-9]{3},[0-9]{3},[0-9]{3}\)" />
      </xs:restriction>
    </xs:simpleType>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="aqua" />
        <xs:enumeration value="black" />
        <xs:enumeration value="blue" />
        <xs:enumeration value="gray" />
        <xs:enumeration value="lime" />
        <xs:enumeration value="green" />
        <xs:enumeration value="maroon" />
        <xs:enumeration value="navy" />
        <xs:enumeration value="olive" />
        <xs:enumeration value="orange" />
        <xs:enumeration value="purple" />
        <xs:enumeration value="red" />
        <xs:enumeration value="silver" />
        <xs:enumeration value="teal" />
        <xs:enumeration value="white" />
        <xs:enumeration value="yellow" />
        <xs:enumeration value="transparent" />
      </xs:restriction>
    </xs:simpleType>
  </xs:union>
</xs:simpleType>

```

```

    </xs:restriction>
  </xs:simpleType>
</xs:union>
</xs:simpleType>

<xs:simpleType name="text-positions">
  <xs:restriction base="xs:string">
    <xs:enumeration value="INTERNAL" />
    <xs:enumeration value="EXTERNAL" />
    <xs:enumeration value="LEFT" />
    <xs:enumeration value="RIGHT" />
    <xs:enumeration value="TOP" />
    <xs:enumeration value="BOTTOM" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="phase-type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="PARALLEL" />
    <xs:enumeration value="PROGRESSIVE" />
    <xs:enumeration value="SEQUENTIAL" />
    <xs:enumeration value="FREE" />
  </xs:restriction>
</xs:simpleType>

<xs:element name="item">
  <xs:complexType>
    <xs:attribute name="element" use="required" type="external-id" />
    <xs:attribute name="param" use="optional" default="" type="xs:string" />
    <xs:attribute name="descrId" use="required" type="xs:string" />
    <xs:attribute name="textPosition" use="optional" default="EXTERNAL"
type="text-positions" />
    <xs:attribute name="maskColor" use="optional" default="#2aa0d5"
type="colors" />
    <xs:attribute name="maskOpacity" use="optional" default="0.8" type="opacity" /
>
    <xs:attribute name="textSize" use="optional" default="25" type="positive-
integer" />
    <xs:attribute name="textColor" use="optional" default="white" type="colors" />
  </xs:complexType>
</xs:element>

<xs:element name="preAction">
  <xs:complexType>
    <xs:attribute name="action" use="required" type="actions" />
    <xs:attribute name="param" use="optional" default="" type="xs:string" />
    <xs:attribute name="clickIndicator" use="optional" default="false"
type="xs:boolean" />
  </xs:complexType>
</xs:element>

<xs:element name="phase">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:choice>
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="item" />
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="preAction" />
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```



```

<xs:attribute name="element" use="required" type="external-id" />
<xs:attribute name="param" use="optional" default="" type="xs:string" />
<xs:attribute name="type" use="optional" default="PARALLEL" type="phase-
type" />
<xs:attribute name="textPosition" use="optional" default="EXTERNAL"
type="text-positions" />
<xs:attribute name="textSize" use="optional" default="25" type="positive-
integer" />
<xs:attribute name="textColor" use="optional" default="white" type="colors" />
<xs:attribute name="maskColor" use="optional" default="#2aa0d5"
type="colors" />
<xs:attribute name="maskOpacity" use="optional" default="0.6" type="opacity" /
>
</xs:complexType>
</xs:element>

<xs:element name="help">
<xs:complexType>
<xs:sequence minOccurs="1" maxOccurs="unbounded">
<xs:element ref="phase" />
</xs:sequence>
<xs:attribute name="name" use="required" type="external-id" />
<xs:attribute name="descrId" use="required" type="external-id" />
<xs:attribute name="opacity" use="optional" default="0.4" type="opacity" />
<xs:attribute name="textPosition" use="optional" default="EXTERNAL"
type="text-positions" />
<xs:attribute name="textSize" use="optional" default="25" type="positive-
integer" />
<xs:attribute name="textColor" use="optional" default="white" type="colors" />
<xs:attribute name="maskColor" use="optional" default="#2aa0d5"
type="colors" />
<xs:attribute name="maskOpacity" use="optional" default="0.6" type="opacity" /
>
<xs:attribute name="firstConnexionGroup" use="optional" default=""
type="xs:string" />
<xs:attribute name="icon" use="optional" default="" type="xs:string" />
</xs:complexType>
</xs:element>
</xs:schema>

```

Download wizards.xsd [../shared_manual/wizards.xsd]

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">

<xs:simpleType name="id">
<xs:restriction base="xs:string">
<xs:pattern value='[A-Z_][A-Z0-9_]+' />
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="list-id">
<xs:restriction base="xs:string">
<xs:pattern value='[A-Z_][A-Z0-9_] +( ;[A-Z_][A-Z0-9_] ) *' />
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="alignment">
<xs:restriction base="id">

```



```

<xs:enumeration value="LEFT" />
<xs:enumeration value="RIGHT" />
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="project-status">
<xs:restriction base="id">
<xs:enumeration value="IGNORE" />
<xs:enumeration value="WARNING" />
<xs:enumeration value="ERROR" />
</xs:restriction>
</xs:simpleType>

<xs:element name="Bundle">
<xs:complexType>
<xs:sequence>
<xs:choice minOccurs="0" maxOccurs="unbounded">
<xs:element ref="tags" />
<xs:element ref="wizard" />
</xs:choice>
</xs:sequence>
</xs:complexType>
</xs:element>

<xs:element name="tags">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="0" maxOccurs="unbounded" ref="tag" />
</xs:sequence>
<xs:attribute name="textAlign" type="alignment" />
<xs:attribute name="valueAlign" type="alignment" />
</xs:complexType>
</xs:element>

<xs:element name="tag">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="0" maxOccurs="unbounded" ref="value" />
</xs:sequence>
<xs:attribute name="defaultValue" />
<xs:attribute name="displayType" /> <!-- Not display-type because it is case
insensitive -->
<xs:attribute name="group" />
<xs:attribute name="measureId" use="required" type="id" />
<xs:attribute name="name" />
<xs:attribute name="placeholder" />
<xs:attribute name="required" type="xs:boolean" />
<xs:attribute name="review" type="xs:boolean" />
<xs:attribute name="suffix" />
<xs:attribute name="targetArtefactTypes" type="list-id" />
<xs:attribute name="textAlign" type="alignment" />
<xs:attribute name="type" use="required" /> <!-- Not tag-type because it is
case insensitive -->
<xs:attribute name="valueAlign" type="alignment" />
</xs:complexType>
</xs:element>

<xs:element name="value">
<xs:complexType>

```

```
<xs:attribute name="key" use="required"/>
<xs:attribute name="value" use="required" type="xs:decimal"/>
</xs:complexType>
</xs:element>

<xs:element name="wizard">
  <xs:complexType>
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="tags"/>
        <xs:element ref="milestones"/>
        <xs:element ref="repositories"/>
        <xs:element ref="tools"/>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="autoBaseline" type="xs:boolean"/>
    <xs:attribute name="group"/>
    <xs:attribute name="groups"/>
    <xs:attribute name="hideRulesEdition" type="xs:boolean"/>
    <xs:attribute name="img"/>
    <xs:attribute name="users"/>
    <xs:attribute name="versionPattern"/>
    <xs:attribute name="wizardId" use="required" type="id"/>
  </xs:complexType>
</xs:element>

<xs:element name="milestones">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="goals"/>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="milestone"/>
    </xs:sequence>
    <xs:attribute name="canCreateMilestone" type="xs:boolean"/>
    <xs:attribute name="canCreateGoal" type="xs:boolean"/>
    <xs:attribute name="hide" type="xs:boolean"/>
  </xs:complexType>
</xs:element>

<xs:element name="goals">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element ref="goal"/>
    </xs:sequence>
    <xs:attribute name="displayableFamilies" use="required" type="list-id"/>
  </xs:complexType>
</xs:element>

<xs:element name="goal">
  <xs:complexType>
    <xs:attribute name="mandatory" use="required" type="xs:boolean"/>
    <xs:attribute name="measureId" use="required" type="id"/>
  </xs:complexType>
</xs:element>

<xs:element name="milestone">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="defaultGoal"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```
<xs:attribute name="id" use="required" type="id"/>
<xs:attribute name="mandatory" type="xs:boolean"/>
</xs:complexType>
</xs:element>

<xs:element name="defaultGoal">
  <xs:complexType>
    <xs:attribute name="measureId" use="required" type="id"/>
    <xs:attribute name="value" use="required" type="xs:integer"/>
  </xs:complexType>
</xs:element>

<xs:element name="repositories">
  <xs:complexType>
    <xs:sequence>
      <xs:element maxOccurs="unbounded" ref="repository"/>
    </xs:sequence>
    <xs:attribute name="all" use="required" type="xs:boolean"/>
    <xs:attribute name="hide" use="required" type="xs:boolean"/>
  </xs:complexType>
</xs:element>

<xs:element name="repository">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" ref="param"/>
    </xs:sequence>
    <xs:attribute name="checkedInUI" type="xs:boolean"/>
    <xs:attribute name="name" use="required"/>
  </xs:complexType>
</xs:element>

<xs:element name="tools">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="tool"/>
    </xs:sequence>
    <xs:attribute name="all" type="xs:boolean"/>
    <xs:attribute name="expandedInUI" type="xs:boolean"/>
  </xs:complexType>
</xs:element>

<xs:element name="tool">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="param"/>
    </xs:sequence>
    <xs:attribute name="checkedInUI" type="xs:boolean"/>
    <xs:attribute name="expandedInUI" type="xs:boolean"/>
    <xs:attribute name="name" use="required"/>
    <xs:attribute name="optional" type="xs:boolean"/>
    <xs:attribute name="projectStatusOnFailure" type="project-status"/>
    <xs:attribute name="projectStatusOnWarning" type="project-status"/>
  </xs:complexType>
</xs:element>

<xs:element name="param">
  <xs:complexType>
    <xs:attribute name="availableChoices"/>
  </xs:complexType>
</xs:element>
```

```
<xs:attribute name="name" use="required" />  
<xs:attribute name="value" use="required" />  
</xs:complexType>  
</xs:element>  
</xs:schema>
```

Appendix C. Licences

Squoring Software

End-User License and Support Agreement

Please read this document carefully. This is a legal agreement by which Squoring Technologies SAS ("Squoring") permits use of its Software products ("Squoring Software"). The user ("Customer") accepts the terms of this Agreement by taking any or all of the following actions: (a) by signing an order form or purchase order referencing either this Agreement or a Technical and Financial proposal issued by Squoring (an "Order Form"), (b) by opening the package containing the Software, and/or (c) by installing the Software on a computer ("Target Hardware").

CUSTOMER CONSENTS TO BE LEGALLY BOUND BY THESE TERMS. IF CUSTOMER DOES NOT AGREE TO ALL OF THE TERMS OF THIS AGREEMENT, CUSTOMER MUST NOT USE THE SOFTWARE AND MUST RETURN IT, INCLUDING ANY PRINTED ASSOCIATED DOCUMENTATION, TO SQUORING WITHIN FOURTEEN (14) DAYS TO RECEIVE A FULL REFUND OF THE PURCHASE PRICE.

1) DEFINITION

- (a) "Squoring Software" includes (i) the Software identified in an Order Form or delivered with this Agreement; (ii) any authorized copies thereof; (iii) all related documentation ("Documentation") delivered with or included in that software; and (iv) any update to that software that Customer may receive from Squoring.
- (b) Squoring Software is based on a traditional 3-tier architecture consisting of:
- . a database and a data folder for storing project and user management data
 - . an application server running the "Squore Server Software" and the "License Server Software"
 - . a client front-end accessible through a Web Browser and a Command Line Interface to interact with the application server from a client machine
- (c) Target hardware ("Target Hardware") is uniquely identified by the hardware specification and the operating system running on it.

2) LICENSE

Squoring grants to Customer, and Customer accepts from Squoring, a non-exclusive and non-transferable right and license to use the Squoring Software that is specified in the Order Form and/or that accompanies this Agreement, but only (i) in accordance with the related Documentation, (ii) subject to Customer's payment of applicable license fees and (iii) subject to the terms and conditions specified below.

Customer agrees that Customer does not have, and does not hereby acquire, any title or rights of ownership in any Squoring Software or, except for the license rights hereby granted, any right to use, copy, transfer or disclose all or any portion of any Squoring Software. The Squoring Software is protected by copyright laws and international treaties.

3) FEES

The fees for the license under this Agreement are set forth in the applicable Order Form or, if no Order Form exists, in the applicable Technical and Financial proposal issued by Squoring, or, if no proposal exists, then in accordance with Squoring current list prices.

4) USE

- (a) Unless otherwise stated in a special agreement, all dissemination or commercial exploitation of Squoring Software results is strictly forbidden.
- (b) The license granted by this Agreement is a license under which a maximum number of active users and projects specified in the Order Form may use the Squoring Software. An "Active User" is a physical user registered in the Squoring Software database. Active Users are not shared among several Squoring software databases. A user is active if any activity has been recorded by the Squore Server Software in the past 6 months. Activities include remote project creation, viewing of analysis results, and e-mail notification.
- (c) The management and regulation of Active Users is managed by the License Server Software hosted on the Target hardware designated by the Customer.
- (d) Except for continuous integration purpose, it is strictly forbidden to share the same Squoring Software login between different physical users.
- (e) Customer will ensure that at least one of its employees has completed the two days on-site training course "Administrating Squoring software" given by a Squoring Software certified trainer, and that such trained employee(s) are the people within Customer's organization who are responsible for interactions with Squoring on maintenance and support matters.
- (f) Customer may make a reasonable number of back-up or archival copies of the Software. Customer will reproduce all confidentiality and proprietary notices on each of these copies and maintain an accurate record of the location of each of these copies.
- (g) Customer will not:
 - . Reverse compile, disassemble, or otherwise reverse engineer any Squoring Software, or allow anyone else to do so (except only to the extent such prohibition is contrary to applicable law).
 - . Remove or destroy any proprietary markings or legends or any encrypted license keys or similar security devices placed upon or contained in any Squoring Software.
 - . Modify or adapt the Squoring Software or create a derivative work based on or incorporate the Squoring Software into or with other software.
 - . Unless otherwise stated in a special agreement, distribute, sublicense, share, display, or in any manner make the Squoring Software available to any third party, with or without compensation.
 - . Use all or any part of the Squoring Software to create other software a principal purpose of which is to perform the same or similar functions as, or to replace any component of, the Squoring Software.

5. MAINTENANCE AND SUPPORT

- (a) Depending on the type of licenses bought by the Customer, Squoring will provide support and maintenance services according to the following schedule:
 - i. In case of perpetual licenses bought by Customer
 - . Maintenance fees will be charged in addition to the price of the purchase of the Squoring Software licenses. The annual initial amount of maintenance fees will be calculated on the basis of a 20% percentage of the net list price of the software licenses purchased by the Customer.
 - . After one year, Maintenance and Support services will be renewed by tacit agreement of the parties, for annual periods. Before the anniversary date of each license for which Support Services are in effect, Squoring shall advise the Customer of the applicable Maintenance and Support Service fees for the coming year.
 - . The termination of maintenance contract will be effective only by sending a registered letter with acknowledgement of receipt denouncing the contract at least 60 days before the end of the period of validity.
 - ii. In case of software licenses subscription

If the license to use the software is subject to a periodic subscription, the maintenance cost for these licenses is included in the subscription price. The maintenance is so provided during all the duration of the subscription, and stops automatically at the end of the subscription validity.

(b) Conditions. Maintenance and support services as defined hereafter in paragraph 5-(c) are applicable subject to the following conditions:

- . Squoring Software is covered by a valid maintenance contract for all acquired Squoring software licenses.
- . Squoring Software was not modified by the Customer.
- . The version of the installed Squoring Software corresponds to one of the two latest annual major updates distributed by Squoring Technologies.
- . Customer engages to comply with the normal use of the software, strictly comply with the instructions given by the Squoring Technologies and to respect all provisions in the present agreement.
- . Customer shall nominate from among its staff a technical coordinator and an alternate coordinator at the Customer Site(s), with up to date knowledge of Squoring licensed products usage and sufficient technical knowledge to interact with Squoring support staff. In case of change of the coordinators, the Customer will provide written notification to Squoring Technologies.

(c) Maintenance and support services include:

- . Assistance from support: the online support (hot line) is available during Squoring normal business hours from 9 am to 6 pm (Central European Time) to answer the questions of the technical coordinator when technical facts encountered in the use of the Squoring Software. Support will help to identify problems and provide, where appropriate, temporary fixing patches. Contact information for support is:
 - support site: <https://support.squoring.com>
 - email: support@squoring.com
- . Corrective maintenance: the corrective maintenance includes the development, to the extent commercially reasonable, of workarounds or program fixes for malfunctions submitted by Customer. Are considered as malfunctions recognized or reproducible defects resulting in distorted results compared to those defined in the software Manual and not coming from non-observance of the instructions of the said Manual.
- . Updates: updates include the delivery of successive versions of the software, being either due to bug fixes or to enhancements of performances or features (this delivery does not necessarily provide new additional features). Customer will destroy any prior version before installing a new update.
- . Rehost: any change of "Target hardware" implies a change of license keys and shall be subject to the prior written authorization of Squoring and to the signature by the Customer of a letter of destruction of all the license files already installed.

d) Maintenance and support services do not include:

- . Time spent, after request of the Customer by Squoring Technologies staff not directly attributable to maintenance services: search for non-reproducible anomalies, malfunction due to non-compliance with the Manual without the Software itself being an issue, unavailability of the system, operating activities prior to the intervention (such as preliminary backup ...).
- . Installation of the Software by Squoring Technologies.
- . Additional services which do not fall within the scope of maintenance services as defined above in paragraph 5-(c).
- . On-site support: Squoring may offer on-site support to Customer at additional charges

e) Procedure for the submission of requests:

- . To be taken into account, Customer requests shall be sent using the support site at <https://support.squoring.com>
- . Customer agrees to give, in support of a request for correction due to an anomaly, any information likely to facilitate the search for the causes of this anomaly, and to give for free to Squoring Technologies an open access to its

premises and development stations in the day and hours necessary to perform the contract, and to ensure the conservation, under the conditions of appropriate security and condition of the latest version, of the sources of programs in case of this is necessary to enable Squoring Technologies to perform its maintenance services.

- . Regarding correction of anomalies, Squoring is committed to act as soon as possible to correct the anomalies detected. Squoring will transfer to the Customer either a technique to bypass the anomaly or a patch of necessary corrections or a new version of the software.
 - . Any issues not resolved with the initial response will be investigated using the data provided. Below are the targeted response times for continued investigations.
 - Blocker: 1 day, daily update. A request is "blocking" when the incident has a significant impact with a risk of operating loss or when data are corrupted. The significant impact is appreciated by Squoring Technologies.
 - Serious: 2 days, weekly update
 - Major: 3 days, monthly update
 - Minor: 1 week, monthly update
- (f) Limitations
- . Squoring is expressly subject to an obligation of means.
 - . Squoring is not required to (i) develop and release any, or any particular type of enhancements or (ii) customize the enhancements to satisfy Customer's particular requirements.
 - . The Updates will not include any upgrade or new version of the Products that Squoring decides, in its sole discretion, to make generally available as a separately priced item.
 - . Squoring will be released from any responsibility in case of breach by the Customer of any provision of this maintenance terms and conditions.

6. WARRANTIES AND REMEDIES

- (a) Limited Warranty. Squoring warrants that it has the right to (i) enter into this Agreement and (ii) grant the licenses hereunder. Squoring also warrants that the Squoring Software will perform substantially as described in the Documentation during a 90 days Warranty Period. Customer acknowledges that (i) the Products may not satisfy all of Customer's requirements and (ii) the use of the Products may not be uninterrupted or error-free.
- (b) Remedies. Squoring or its representative will correct or replace any defective Software. Customer acknowledges that this paragraph sets forth Customer's exclusive remedy, and Squoring exclusive liability, for any breach of warranty or other duty related to the quality of the Products.
- (c) Disclaimer. EXCEPT AS EXPRESSLY PROVIDED IN THIS AGREEMENT OR REQUIRED BY APPLICABLE LAW, ALL WARRANTIES, CONDITIONS, REPRESENTATIONS, INDEMNITIES AND GUARANTEES WITH RESPECT TO THE PRODUCTS, WHETHER EXPRESS OR IMPLIED, ARISING BY LAW, CUSTOM, PRIOR ORAL OR WRITTEN STATEMENTS BY SQUORING, ITS SQUORINGS OR REPRESENTATIVES OR OTHERWISE (INCLUDING, BUT NOT LIMITED TO ANY WARRANTY OF MERCHANTABILITY, SATISFACTION AND FITNESS FOR A PARTICULAR PURPOSE) ARE HEREBY OVERRIDDEN, EXCLUDED AND DISCLAIMED.
- IN NO EVENT SHALL THE AGGREGATE LIABILITY OF SQUORING TO CUSTOMER ON ACCOUNT OF ANY MATTER ARISING WITH RESPECT TO THE SQUORING SOFTWARE EXCEED THE LICENSE FEES PAID BY CUSTOMER UNDER THIS AGREEMENT.
- (d) Infringement Indemnity. If an action is brought against Customer claiming that the Product infringes a patent, trade secret or copyright, Squoring will defend Customer at Squoring expense and, subject to this Section, pay the damages and costs finally awarded against Customer in the infringement action, but only if (i) Customer notifies Squoring promptly upon learning that the claim might be asserted, (ii) Squoring has sole control over the defense of the claim and any negotiation for its settlement or compromise, and (iii) Customer takes no action that is contrary to Squoring interest. If a claim described above may

be or has been asserted, Customer will permit Squoring, at Squoring option and expense, to (A) procure the right to continue using the Product, (B) replace or modify the Product to eliminate the infringement while providing functionally equivalent performance, or (C) accept the return of the Product and refund to Customer the License Fee actually paid to Squoring for such Product, less depreciation based on a 5-year straight-line-depreciation schedule.

Squoring shall have no indemnity obligation to Customer under this Section if the patent or copyright infringement claim results from (i) a correction or modification of the Product not provided by Squoring, (ii) the failure to promptly install an Update or Enhancement at Squoring direction with knowledge that installation thereof would have avoided the infringement or (iii) the combination of the Product with other non-Squoring software or (iv) any unauthorized use of the Squoring Software, or (v) any version of the Software other than the latest update offered by Squoring to Customer at no additional charge.

7. LIMITATION OF LIABILITY

UNDER NO CIRCUMSTANCES WILL SQUORING OR ITS SQUORINGS BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, SPECIAL, PUNITIVE OR INCIDENTAL DAMAGES OR LOST PROFITS, WHETHER FORESEEABLE OR UNFORESEEABLE, BASED ON CUSTOMER'S CLAIMS OR THOSE OF ITS CUSTOMERS (INCLUDING, BUT NOT LIMITED TO, CLAIMS FOR LOSS OF DATA, GOODWILL, USE OF MONEY OR USE OF THE PRODUCTS, INTERRUPTION IN USE OR AVAILABILITY OF DATA, STOPPAGE OF OTHER WORK OR IMPAIRMENT OF OTHER ASSETS), ARISING OUT OF BREACH OR FAILURE OF EXPRESS OR IMPLIED WARRANTY, BREACH OF CONTRACT, MISREPRESENTATION, NEGLIGENCE, STRICT LIABILITY IN TORT OR OTHERWISE. IN NO EVENT WILL THE AGGREGATE LIABILITY WHICH SQUORING OR ITS SQUORINGS MAY INCUR IN ANY ACTION OR PROCEEDING EXCEED THE LICENSE FEES ACTUALLY PAID BY CUSTOMER FOR THE SPECIFIC PRODUCT THAT DIRECTLY CAUSED THE DAMAGE. THIS SECTION WILL NOT APPLY ONLY WHEN AND TO THE EXTENT THAT APPLICABLE LAW SPECIFICALLY REQUIRES LIABILITY, DESPITE THE FOREGOING EXCLUSION AND LIMITATION.

8. OWNERSHIP

All trademarks, service marks, patents, copyrights, trade secrets and other proprietary rights in or related to the Products are and will remain the exclusive property of Squoring, whether or not specifically recognized or perfected under local applicable law. Customer will not take any action that jeopardizes Squoring proprietary rights or acquire any right in the Products, except the limited use rights specified in Section 4. Squoring will own all rights in any copy, translation, modification, adaptation or derivation of the Products, including any improvement or development thereof.

9. CONFIDENTIALITY

(a) Confidentiality. Customer acknowledges that the Products constitute and incorporate confidential and proprietary information developed or acquired by or licensed to Squoring. Customer will take all reasonable precautions necessary to safeguard the confidentiality of the Products, including at a minimum those taken by Customer to protect Customer's own confidential information. Customer will not allow the removal or defacement of any confidentiality or proprietary notice placed on the Products. The placement of copyright notices on these items will not constitute publication or otherwise impair their confidential nature.

(b) Disclosure. Customer will not disclose, in whole or in part, the Products or any portion thereof or other information that has been designated as confidential to any individual, entity or other person, except to those of Customer's employees or consultants who require access for Customer's authorized use of the Products, provided such consultants agree in writing to comply with the use and non-disclosure restrictions applicable to the Products under this

Agreement. Customer acknowledges that any unauthorized use or disclosure of the Products may cause irreparable damage to Squoring. If an unauthorized use or disclosure occurs, Customer will immediately notify Squoring and take, at Customer's expense, all steps which may be available to recover the Products and to prevent their subsequent unauthorized use or dissemination. Squoring agrees to take the same action regarding any information designated in writing as proprietary which it receives from Customer ("Customer Information").

(c) Limitation. Neither Squoring nor Customer will have any confidentiality obligation with respect to any portion of the Products or Customer Information that (i) the receiving party knew or independently developed before receiving such Products or Customer Information under this Agreement, (ii) the receiving party lawfully obtained from a third party under no confidentiality obligation, or (iii) became available to the public other than as a result of any act or omission by the receiving party or any of receiving party's employees or consultants.

10. TERMINATION

Customer may terminate this Agreement or any Order Form, without right to refund, by notifying Squoring of such termination and returning the Product and copies thereof to Squoring. Squoring may terminate this Agreement, upon reasonable notice and without judicial or administrative resolution, if Customer or any of Customer's employees or consultants breach any material term or condition hereof. This Agreement will terminate automatically if Customer becomes insolvent or enters into bankruptcy, suspension of payments, moratorium, or any other proceeding that relates to insolvency or protection or creditors' rights.

Upon the termination of this Agreement for any reason, all rights granted to Customer hereunder will cease, and Customer will stop using Squoring Software, return or destroy all copies and so certify to Squoring in writing. The provisions of Sections 6-8 will survive the termination of this Agreement.

11. INSPECTION

During the term of this Agreement, Squoring or its representative, if in receipt of credible evidence of non-compliance, may, upon prior notice to Customer, inspect the files, computer processors, equipment and facilities of Customer during normal working hours to verify Customer's compliance with this Agreement. While conducting such inspection, Squoring or its representative will be entitled to copy any item that Customer may possess in violation of this Agreement, without disruption of Squoring business and violation of Laws and Regulations.

12. ASSIGNMENT

Customer shall not assign, delegate or otherwise transfer this Agreement or any of its rights or obligations hereunder without Squoring prior approval which shall not be unreasonably withheld.

13. MISCELLANEOUS

. Any terms and conditions of any unilateral letter, memorandum, purchase order or other writing issued by Customer shall not be binding on Squoring. Any waiver or modification of this Agreement will not be effective unless executed in writing and signed by an authorized representative of Squoring and Customer. This Agreement will bind Customer's successors-in-interest.

. This Agreement will be governed by and interpreted in accordance with the laws of France. If any provision of this Agreement is held to be unenforceable, in whole or in part, such holding will not affect the validity of the other

provisions of this Agreement, unless the Parties in good faith deem the unenforceable provision to be essential, in which case either Party may terminate this Agreement effective immediately upon notice to the other Party. This Agreement constitutes the complete and entire statement of all conditions and representations of the agreement between Squoring and Customer with respect to its subject matter and supersedes all prior writings or understandings.

Redistributed Software	Version	License File

Wildfly	10.1.0.Final	lgpl-2.1.txt
PostgreSQL	8.4	postgresql-bsd.txt
Perl	5.12.3	al.txt
TCL	8.5	tcltkl.txt,tcllib.txt
PhantomJS	2.1.1	bsd3.txt
Components	Sub component	Version License File

Richfaces	4.5.17	lgpl-2.1.txt
PostgreSQL JDBC Driver	42.0.0	bsd2.txt
Oracle JDBC driver	12.1.02	otn.txt
Omnifaces	2.6	apache-2.0.txt
ANTLR	3.1	antlr3.txt
JasperReports Library	4.8	lgpl-2.1.txt
Checkstyle	5.6	lgpl-2.1.txt
JTcl	2.8	jtcl.txt, amd.txt, itcl.txt, janino.txt, tcllib.txt, tcltk.txt, ucb.txt
log4j	1.2.17	apache-2.0.txt
Apache Commons	commons-lang3 3.1	apache-2.0.txt
Apache HttpComponents	httpclient 4.1.2	apache-2.0.txt
Apache HttpComponents	httpmime 4.1.2	apache-2.0.txt
Apache XML Graphics	batik-transcoder 1.7	apache-2.0.txt
Bouncy Castle	bcmail-jdk16 1.46	bouncy-castle.txt
com.beust	jcommander 1.48	apache-2.0.txt
com.google.collections	google-collections 1.0	apache-2.0.txt
com.googlecode.juniversalchardet	juniversalchardet 1.0.3	mpl1.1.txt
com.sun.mail	javax.mail 1.5.3	gf.txt
commons-cli	commons-cli 1.2	apache-2.0.txt
commons-collections	commons-collections 3.2.2	apache-2.0.txt
javax.enterprise	cdi-api 1.2	apache-2.0.txt
javax.validation	validation-api 1.1.0.Final	apache-2.0.txt
net.java.dev.jna	jna 4.1.0	lgpl-2.1.txt
net.sf.jsci	jsci 1.2	lgpl-2.1.txt
net.sf.saxon	saxon-xom 8.7	mpl1.0.txt
nux	nux 1.6	nux.txt
org.glassfish	javax.json 1.0.4	gf.txt
org.jdom	jdom2 2.0.5	jdom.txt
Resources	Version	License File

CodeMirror	4.4.0	mit.txt
font-awesome	4.7.0	mit.txt, ofl-1.1.txt
JavaScript InfoVis Toolkit	2.0.1	mit.txt
jquery	1.12.3	jquery.txt
jquery-mobile	1.4.2	jquery-mobile.txt
jquery-ui-resizable	1.11.4	mit.txt, jquery.txt
notify.js	0.4.2	mit.txt
splitter.js	1.51	mit.txt
gridstack.js	1.0.0	mit.txt
spectrum.js	1.8.0	mit.txt

al

The "Artistic License"

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions:

"Package" refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.

"Standard Version" refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.

"Copyright Holder" is whoever is named in the copyright or copyrights for the package.

"You" is you, if you're thinking about copying or distributing this Package.

"Reasonable copying fee" is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

"Freely Available" means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.

2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.

3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:

a) place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.

b) use the modified Package only within your corporation or organization.

c) rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.

d) make other distribution arrangements with the Copyright Holder.

4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:

a) distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.

b) accompany the distribution with the machine-readable source of the Package with your modifications.

c) give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.

d) make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.

6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whoever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package via the so-called "undump" or "unexec" methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.

7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.

8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible

to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.

9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.

10. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

amd

license.amd

The following license terms apply to the TJC compiler source and test files located in the src/tjc, src/tests/tjc, and tests/tjc directories.

Copyright 2005 Advanced Micro Devices, Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that existing copyright notices are retained in all copies, this notice is included verbatim in any distributions, and the terms and conditions hererin are met.

Use of the this software manifests acceptance of the terms of this license by performance.

The name of Advanced Micro Devices, Inc. may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY ADVANCED MICRO DEVICES, INC. "AS IS" AND ANY EXPRESS, IMPLIED OR STATUTORY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT, OR THOSE ARISING FROM CUSTOM OF TRADE OR COURSE OF USAGE ARE DISCLAIMED.

IN NO EVENT SHALL ADVANCED MICRO DEVICES, INC. BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED, AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE ITS DOCUMENTATION OR ANY DERIVATIVES THEREOF, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. BY USING THIS SOFTWARE WITHOUT CHARGE, YOU ACCEPT THIS ALLOCATION OF RISK. THIS DISCLAIMER OF LIABILITY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. ADVANCED MICRO DEVICES, INC. HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS OF THIS SOFTWARE.

In the redistribution and use of this software, each party shall at all times comply with all applicable governmental laws, statutes, ordinances, rules, regulations, orders, and other requirements, including without limitation such governmental requirements applicable to environmental protection, health, safety, wages, hours, equal employment opportunity, nondiscrimination, working conditions, import or export control, and

transportation. Without limiting the foregoing, each party shall adhere to the U.S. Export Administration Regulations (EAR), currently found at 15 C.F.R. Sections 730 through 744, and, unless properly authorized by the U.S. Government, shall not (1) export, re-export or release restricted technology, software, or source code to a national of a country in Country Groups D:1 or E:1, or (2) export to Country Groups D:1 or E:1 the direct product of such technology or software, if such foreign produced direct product is subject to national security controls as identified on the Commerce Control List (currently found in Supplement 1 to Section 774 of EAR). These export requirements shall survive any expiration or termination of this agreement.

antlr3

ANTLR 3 License

[The BSD License]
Copyright (c) 2010 Terence Parr
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the author nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS AS IS AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

apache-2.0

Apache License
Version 2.0, January 2004
<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction,

and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable

copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
 - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
 - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
 - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
 - (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with

the conditions stated in this License.

5. **Submission of Contributions.** Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. **Trademarks.** This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. **Disclaimer of Warranty.** Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. **Limitation of Liability.** In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. **Accepting Warranty or Additional Liability.** While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier

```
identification within third-party archives.
```

```
Copyright [yyyy] [name of copyright owner]
```

```
Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.
```

bouncy-castle

```
Copyright (c) 2000 - 2012 The Legion Of The Bouncy Castle  
(http://www.bouncycastle.org)
```

```
Permission is hereby granted, free of charge, to any person obtaining a copy of  
this software and associated documentation files (the "Software"), to deal in  
the Software without restriction, including without limitation the rights to  
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies  
of the Software, and to permit persons to whom the Software is furnished to do  
so, subject to the following conditions:
```

```
The above copyright notice and this permission notice shall be included in all  
copies or substantial portions of the Software.
```

```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
SOFTWARE.
```

bsd2

```
Copyright (c) 1997, PostgreSQL Global Development Group  
All rights reserved.
```

```
Redistribution and use in source and binary forms, with or without  
modification, are permitted provided that the following conditions are met:
```

1. Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation
and/or other materials provided with the distribution.

```
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"  
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE  
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE  
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE  
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR  
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
```

SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

bsd3

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the <organization> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL <COPYRIGHT HOLDER> BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

gf

GlassFish

COMMON DEVELOPMENT AND DISTRIBUTION LICENSE (CDDL) Version 1.1

1. Definitions.

- 1.1. "Contributor" means each individual or entity that creates or contributes to the creation of Modifications.
- 1.2. "Contributor Version" means the combination of the Original Software, prior Modifications used by a Contributor (if any), and the Modifications made by that particular Contributor.
- 1.3. "Covered Software" means (a) the Original Software, or (b) Modifications, or (c) the combination of files containing Original Software with files containing Modifications, in each case including portions thereof.
- 1.4. "Executable" means the Covered Software in any form other than Source Code.
- 1.5. "Initial Developer" means the individual or entity that first makes Original Software available under this License.
- 1.6. "Larger Work" means a work which combines Covered Software or portions thereof with code not governed by the terms of this License.

1.7. "License" means this document.

1.8. "Licensable" means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently acquired, any and all of the rights conveyed herein.

1.9. "Modifications" means the Source Code and Executable form of any of the following:

A. Any file that results from an addition to, deletion from or modification of the contents of a file containing Original Software or previous Modifications;

B. Any new file that contains any part of the Original Software or previous Modification; or

C. Any new file that is contributed or otherwise made available under the terms of this License.

1.10. "Original Software" means the Source Code and Executable form of computer software code that is originally released under this License.

1.11. "Patent Claims" means any patent claim(s), now owned or hereafter acquired, including without limitation, method, process, and apparatus claims, in any patent Licensable by grantor.

1.12. "Source Code" means (a) the common form of computer software code in which modifications are made and (b) associated documentation included in or with such code.

1.13. "You" (or "Your") means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License. For legal entities, "You" includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, "control" means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

2. License Grants.

2.1. The Initial Developer Grant.

Conditioned upon Your compliance with Section 3.1 below and subject to third party intellectual property claims, the Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license:

(a) under intellectual property rights (other than patent or trademark) Licensable by Initial Developer, to use, reproduce, modify, display, perform, sublicense and distribute the Original Software (or portions thereof), with or without Modifications, and/or as part of a Larger Work; and

(b) under Patent Claims infringed by the making, using or selling of Original Software, to make, have made, use, practice, sell, and offer for sale, and/or otherwise dispose of the Original Software

(or portions thereof).

(c) The licenses granted in Sections 2.1(a) and (b) are effective on the date Initial Developer first distributes or otherwise makes the Original Software available to a third party under the terms of this License.

(d) Notwithstanding Section 2.1(b) above, no patent license is granted: (1) for code that You delete from the Original Software, or (2) for infringements caused by: (i) the modification of the Original Software, or (ii) the combination of the Original Software with other software or devices.

2.2. Contributor Grant.

Conditioned upon Your compliance with Section 3.1 below and subject to third party intellectual property claims, each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license:

(a) under intellectual property rights (other than patent or trademark) Licensable by Contributor to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof), either on an unmodified basis, with other Modifications, as Covered Software and/or as part of a Larger Work; and

(b) under Patent Claims infringed by the making, using, or selling of Modifications made by that Contributor either alone and/or in combination with its Contributor Version (or portions of such combination), to make, use, sell, offer for sale, have made, and/or otherwise dispose of: (1) Modifications made by that Contributor (or portions thereof); and (2) the combination of Modifications made by that Contributor with its Contributor Version (or portions of such combination).

(c) The licenses granted in Sections 2.2(a) and 2.2(b) are effective on the date Contributor first distributes or otherwise makes the Modifications available to a third party.

(d) Notwithstanding Section 2.2(b) above, no patent license is granted: (1) for any code that Contributor has deleted from the Contributor Version; (2) for infringements caused by: (i) third party modifications of Contributor Version, or (ii) the combination of Modifications made by that Contributor with other software (except as part of the Contributor Version) or other devices; or (3) under Patent Claims infringed by Covered Software in the absence of Modifications made by that Contributor.

3. Distribution Obligations.

3.1. Availability of Source Code.

Any Covered Software that You distribute or otherwise make available in Executable form must also be made available in Source Code form and that Source Code form must be distributed only under the terms of this License. You must include a copy of this License with every copy of the Source Code form of the Covered Software You distribute or otherwise make available. You must inform recipients of any such Covered Software in Executable form as to how they can obtain such

Covered Software in Source Code form in a reasonable manner on or through a medium customarily used for software exchange.

3.2. Modifications.

The Modifications that You create or to which You contribute are governed by the terms of this License. You represent that You believe Your Modifications are Your original creation(s) and/or You have sufficient rights to grant the rights conveyed by this License.

3.3. Required Notices.

You must include a notice in each of Your Modifications that identifies You as the Contributor of the Modification. You may not remove or alter any copyright, patent or trademark notices contained within the Covered Software, or any notices of licensing or any descriptive text giving attribution to any Contributor or the Initial Developer.

3.4. Application of Additional Terms.

You may not offer or impose any terms on any Covered Software in Source Code form that alters or restricts the applicable version of this License or the recipients' rights hereunder. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Software. However, you may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor. You must make it absolutely clear that any such warranty, support, indemnity or liability obligation is offered by You alone, and You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of warranty, support, indemnity or liability terms You offer.

3.5. Distribution of Executable Versions.

You may distribute the Executable form of the Covered Software under the terms of this License or under the terms of a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable form does not attempt to limit or alter the recipient's rights in the Source Code form from the rights set forth in this License. If You distribute the Covered Software in Executable form under a different license, You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or Contributor. You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of any such terms You offer.

3.6. Larger Works.

You may create a Larger Work by combining Covered Software with other code not governed by the terms of this License and distribute the Larger Work as a single product. In such a case, You must make sure the requirements of this License are fulfilled for the Covered Software.

4. Versions of the License.

4.1. New Versions.

Oracle is the initial license steward and may publish revised and/or new versions of this License from time to time. Each version will be given a distinguishing version number. Except as provided in Section 4.3, no one other than the license steward has the right to modify this License.

4.2. Effect of New Versions.

You may always continue to use, distribute or otherwise make the Covered Software available under the terms of the version of the License under which You originally received the Covered Software. If the Initial Developer includes a notice in the Original Software prohibiting it from being distributed or otherwise made available under any subsequent version of the License, You must distribute and make the Covered Software available under the terms of the version of the License under which You originally received the Covered Software. Otherwise, You may also choose to use, distribute or otherwise make the Covered Software available under the terms of any subsequent version of the License published by the license steward.

4.3. Modified Versions.

When You are an Initial Developer and You want to create a new license for Your Original Software, You may create and use a modified version of this License if You: (a) rename the license and remove any references to the name of the license steward (except to note that the license differs from this License); and (b) otherwise make it clear that the license contains terms which differ from this License.

5. DISCLAIMER OF WARRANTY.

COVERED SOFTWARE IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE COVERED SOFTWARE IS FREE OF DEFECTS, MERCHANTABILITY, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE COVERED SOFTWARE IS WITH YOU. SHOULD ANY COVERED SOFTWARE PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL DEVELOPER OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED SOFTWARE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

6. TERMINATION.

6.1. This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

6.2. If You assert a patent infringement claim (excluding declaratory judgment actions) against Initial Developer or a Contributor (the Initial Developer or Contributor against whom You assert such claim is referred to as "Participant") alleging that the

Participant Software (meaning the Contributor Version where the Participant is a Contributor or the Original Software where the Participant is the Initial Developer) directly or indirectly infringes any patent, then any and all rights granted directly or indirectly to You by such Participant, the Initial Developer (if the Initial Developer is not the Participant) and all Contributors under Sections 2.1 and/or 2.2 of this License shall, upon 60 days notice from Participant terminate prospectively and automatically at the expiration of such 60 day notice period, unless if within such 60 day period You withdraw Your claim with respect to the Participant Software against such Participant either unilaterally or pursuant to a written agreement with Participant.

6.3. If You assert a patent infringement claim against Participant alleging that the Participant Software directly or indirectly infringes any patent where such claim is resolved (such as by license or settlement) prior to the initiation of patent infringement litigation, then the reasonable value of the licenses granted by such Participant under Sections 2.1 or 2.2 shall be taken into account in determining the amount or value of any payment or license.

6.4. In the event of termination under Sections 6.1 or 6.2 above, all end user licenses that have been validly granted by You or any distributor hereunder prior to termination (excluding licenses granted to You by any distributor) shall survive termination.

7. LIMITATION OF LIABILITY.

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED SOFTWARE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

8. U.S. GOVERNMENT END USERS.

The Covered Software is a "commercial item," as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of "commercial computer software" (as that term is defined at 48 C.F.R. § 252.227-7014(a)(1)) and "commercial computer software documentation" as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire Covered Software with only those rights set forth herein. This U.S. Government Rights clause is in lieu of, and supersedes, any other FAR, DFAR, or other clause or provision that addresses Government rights in computer software under this License.

9. MISCELLANEOUS.

This License represents the complete agreement concerning subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. This License shall be governed by the law of the jurisdiction specified in a notice contained within the Original Software (except to the extent applicable law, if any, provides otherwise), excluding such jurisdiction's conflict-of-law provisions. Any litigation relating to this License shall be subject to the jurisdiction of the courts located in the jurisdiction and venue specified in a notice contained within the Original Software, with the losing party responsible for costs, including, without limitation, court costs and reasonable attorneys' fees and expenses. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not apply to this License. You agree that You alone are responsible for compliance with the United States export administration regulations (and the export control laws and regulation of any other countries) when You use, distribute or otherwise make available any Covered Software.

10. RESPONSIBILITY FOR CLAIMS.

As between Initial Developer and the Contributors, each party is responsible for claims and damages arising, directly or indirectly, out of its utilization of rights under this License and You agree to work with Initial Developer and Contributors to distribute such responsibility on an equitable basis. Nothing herein is intended or shall be deemed to constitute any admission of liability.

NOTICE PURSUANT TO SECTION 9 OF THE COMMON DEVELOPMENT AND DISTRIBUTION LICENSE (CDDL)

|The code released under the CDDL shall be governed by the laws of the State of California (excluding conflict-of-law provisions). Any litigation relating to this License shall be subject to the jurisdiction of the Federal Courts of the Northern District of California and the state courts of the State of California, with venue lying in Santa Clara County, California.

The GNU General Public License (GPL) Version 2, June 1991

|Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor
Boston, MA 02110-1335
USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the

scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the

original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the

author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

One line to give the program's name and a brief idea of what it does.
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1335 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type
`show w'. This is free software, and you are welcome to redistribute
it under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the
program `Gnomovision' (which makes passes at compilers) written by
James Hacker.
```

```
signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

|

```
|Certain source files distributed by Oracle America, Inc. and/or its
|affiliates are subject to the following clarification and special
|exception to the GPLv2, based on the GNU Project exception for its
|Classpath libraries, known as the GNU Classpath Exception, but only
|where Oracle has expressly included in the particular source file's
|header the words "Oracle designates this particular file as subject to
|the "Classpath" exception as provided by Oracle in the LICENSE file
|that accompanied this code."
```

You should also note that Oracle includes multiple, independent programs in this software package. Some of those programs are provided under licenses deemed incompatible with the GPLv2 by the Free Software Foundation and others. For example, the package includes programs licensed under the Apache License, Version 2.0. Such programs are licensed to you under their original licenses.

Oracle facilitates your further distribution of this package by adding the Classpath Exception to the necessary parts of its GPLv2 code, which permits you to use that code in combination with other independent modules not licensed under the GPLv2. However, note that this would not permit you to commingle code under an incompatible license with Oracle's GPLv2 licensed code by, for example, cutting and pasting such code into a file also containing Oracle's GPLv2 licensed code and then distributing the result. Additionally, if you were to remove the Classpath Exception from any of the files to which it applies and

distribute the result, you would likely be required to license some or all of the other code in that distribution under the GPLv2 as well, and since the GPLv2 is incompatible with the license terms of some items included in the distribution by Oracle, removing the Classpath Exception could therefore effectively compromise your ability to further distribute the package.

Proceed with caution and we recommend that you obtain the advice of a lawyer skilled in open source matters before removing the Classpath Exception or making modifications to this package which may subsequently be redistributed and/or involve the use of third party software.

CLASSPATH EXCEPTION

Linking this library statically or dynamically with other modules is making a combined work based on this library. Thus, the terms and conditions of the GNU General Public License version 2 cover the whole combination.

As a special exception, the copyright holders of this library give you permission to link this library with independent modules to produce an executable, regardless of the license terms of these independent modules, and to copy and distribute the resulting executable under terms of your choice, provided that you also meet, for each linked independent module, the terms and conditions of the license of that module. An independent module is a module which is not derived from or based on this library. If you modify this library, you may extend this exception to your version of the library, but you are not obligated to do so. If you do not wish to do so, delete this exception statement from your version.

gpl-2.0

GNU GENERAL PUBLIC LICENSE Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete

machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you

may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES

PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>  
Copyright (C) <year> <name of author>
```

```
This program is free software; you can redistribute it and/or modify  
it under the terms of the GNU General Public License as published by  
the Free Software Foundation; either version 2 of the License, or  
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,  
but WITHOUT ANY WARRANTY; without even the implied warranty of  
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License along  
with this program; if not, write to the Free Software Foundation, Inc.,  
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author  
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may

be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.
```

```
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

gpl

```
Copyright (C) <year> <name of author>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License
along with this program. If not, see <https://www.gnu.org/licenses/>.
```

itcl

```
license.itcl
```

The following license terms apply to the Itcl source and test files located in the src/itcl and tests/itcl directories.

This software is copyrighted by Cadence Design Systems, Inc., and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

```
IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY
```


FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

GOVERNMENT USE: If you are acquiring this software on behalf of the U.S. government, the Government shall have only "Restricted Rights" in the software and related documentation as defined in the Federal Acquisition Regulations (FARs) in Clause 52.227.19 (c) (2). If you are acquiring the software on behalf of the Department of Defense, the software shall be classified as "Commercial Computer Software" and the Government shall have only "Restricted Rights" as defined in Clause 252.227-7013 (c) (1) of DFARS. Notwithstanding the foregoing, the authors grant the U.S. Government and others acting in its behalf permission to use and distribute the software in accordance with the terms specified in this license.

janino

license.janino

Janino - An embedded Java[™] compiler

Copyright (c) 2001-2010, Arno Unkrig
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

jdom

```
/*--  
  
Copyright (C) 2000-2012 Jason Hunter & Brett McLaughlin.  
All rights reserved.  
  
Redistribution and use in source and binary forms, with or without  
modification, are permitted provided that the following conditions  
are met:  
  
1. Redistributions of source code must retain the above copyright  
notice, this list of conditions, and the following disclaimer.  
  
2. Redistributions in binary form must reproduce the above copyright  
notice, this list of conditions, and the disclaimer that follows  
these conditions in the documentation and/or other materials  
provided with the distribution.  
  
3. The name "JDOM" must not be used to endorse or promote products  
derived from this software without prior written permission. For  
written permission, please contact <request_AT_jdom_DOT_org>.  
  
4. Products derived from this software may not be called "JDOM", nor  
may "JDOM" appear in their name, without prior written permission  
from the JDOM Project Management <request_AT_jdom_DOT_org>.  
  
In addition, we request (but do not require) that you include in the  
end-user documentation provided with the redistribution and/or in the  
software itself an acknowledgement equivalent to the following:  
    "This product includes software developed by the  
    JDOM Project (http://www.jdom.org/)."  
Alternatively, the acknowledgment may be graphical using the logos  
available at http://www.jdom.org/images/logos.  
  
THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED  
WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES  
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE  
DISCLAIMED. IN NO EVENT SHALL THE JDOM AUTHORS OR THE PROJECT  
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,  
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT  
LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF  
USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND  
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,  
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT  
OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF  
SUCH DAMAGE.  
  
This software consists of voluntary contributions made by many  
individuals on behalf of the JDOM Project and was originally  
created by Jason Hunter <jhunter_AT_jdom_DOT_org> and  
Brett McLaughlin <brett_AT_jdom_DOT_org>. For more information  
on the JDOM Project, please see <http://www.jdom.org/>.  
  
*/
```

jdom2

```
/*--  
  
Copyright (C) 2000-2012 Jason Hunter & Brett McLaughlin.  
All rights reserved.  
  
Redistribution and use in source and binary forms, with or without  
modification, are permitted provided that the following conditions  
are met:  
  
1. Redistributions of source code must retain the above copyright  
notice, this list of conditions, and the following disclaimer.  
  
2. Redistributions in binary form must reproduce the above copyright  
notice, this list of conditions, and the disclaimer that follows  
these conditions in the documentation and/or other materials  
provided with the distribution.  
  
3. The name "JDOM" must not be used to endorse or promote products  
derived from this software without prior written permission. For  
written permission, please contact <request_AT_jdom_DOT_org>.  
  
4. Products derived from this software may not be called "JDOM", nor  
may "JDOM" appear in their name, without prior written permission  
from the JDOM Project Management <request_AT_jdom_DOT_org>.  
  
In addition, we request (but do not require) that you include in the  
end-user documentation provided with the redistribution and/or in the  
software itself an acknowledgement equivalent to the following:  
    "This product includes software developed by the  
    JDOM Project (http://www.jdom.org/)."  
Alternatively, the acknowledgment may be graphical using the logos  
available at http://www.jdom.org/images/logos.  
  
THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED  
WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES  
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE  
DISCLAIMED. IN NO EVENT SHALL THE JDOM AUTHORS OR THE PROJECT  
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,  
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT  
LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF  
USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND  
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,  
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT  
OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF  
SUCH DAMAGE.  
  
This software consists of voluntary contributions made by many  
individuals on behalf of the JDOM Project and was originally  
created by Jason Hunter <jhunter_AT_jdom_DOT_org> and  
Brett McLaughlin <brett_AT_jdom_DOT_org>. For more information  
on the JDOM Project, please see <http://www.jdom.org/>.  
  
*/
```

jquery-mobile

Copyright jQuery Foundation and other contributors,
<https://jquery.org/>

This software consists of voluntary contributions made by many individuals. For exact contribution history, see the revision history available at <https://github.com/jquery/jquery-mobile>

The following license applies to all parts of this software except as documented below:

====

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

====

Copyright and related rights for sample code are waived via CC0. Sample code is defined as all source code contained within the demos directory.

CC0: <http://creativecommons.org/publicdomain/zero/1.0/>

====

All files located in the node_modules and external directories are externally maintained libraries used by this software which have their own licenses; we recommend you read them, as their terms may differ from the terms above.

jquery

Copyright JS Foundation and other contributors, <https://js.foundation/>

This software consists of voluntary contributions made by many individuals. For exact contribution history, see the revision history available at <https://github.com/jquery/jquery>

The following license applies to all parts of this software except as documented below:

====

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

====

All files located in the node_modules and external directories are externally maintained libraries used by this software which have their own licenses; we recommend you read them, as their terms may differ from the terms above.

jtcl

license.terms

SUN MICROSYSTEMS, INC. THROUGH ITS SUN MICROSYSTEMS LABORATORIES DIVISION ("SUN") WILL LICENSE THIS SOFTWARE AND THE ACCOMPANYING DOCUMENTATION TO YOU (a "Licensee") ONLY ON YOUR ACCEPTANCE OF ALL THE TERMS SET FORTH BELOW.

Sun grants Licensee a non-exclusive, royalty-free right to download, install, compile, use, copy and distribute the Software, modify or otherwise create derivative works from the Software (each, a "Modification") and distribute any Modification in source code and/or binary code form to its customers with a license agreement containing these terms and noting that the Software has been modified. The Software is copyrighted by Sun and other third parties and Licensee shall retain and reproduce all copyright and other notices presently on the Software. As between Sun and Licensee, Sun is the sole owner of all rights in and to the Software other than the limited rights granted to Licensee herein; Licensee will own its Modifications, expressly subject to Sun's continuing ownership of the Software. Licensee will, at its expense, defend and indemnify Sun and its licensors from and against any third party claims, including costs and reasonable attorneys' fees, and be wholly responsible for any liabilities arising out of or related to Licensee's development, use or distribution of the Software or Modifications. Any distribution of the Software and Modifications must comply with all applicable United States export control laws.

THE SOFTWARE IS BEING PROVIDED TO LICENSEE "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT,

ARE DISCLAIMED. IN NO EVENT WILL SUN BE LIABLE HEREUNDER FOR ANY DIRECT DAMAGES OR ANY INDIRECT, PUNITIVE, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES OF ANY KIND.

lgpl-2.1

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that

there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The

former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany

it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under

Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the

original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the

ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the library's name and a brief idea of what it does.>  
Copyright (C) <year> <name of author>
```

```
This library is free software; you can redistribute it and/or  
modify it under the terms of the GNU Lesser General Public  
License as published by the Free Software Foundation; either  
version 2.1 of the License, or (at your option) any later version.
```

```
This library is distributed in the hope that it will be useful,  
but WITHOUT ANY WARRANTY; without even the implied warranty of  
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU  
Lesser General Public License for more details.
```

```
You should have received a copy of the GNU Lesser General Public  
License along with this library; if not, write to the Free Software  
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301  
USA
```

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the  
library `Frob' (a library for tweaking knobs) written by James Random Hacker.
```

```
<signature of Ty Coon>, 1 April 1990  
Ty Coon, President of Vice
```

That's all there is to it!

mit

```
Copyright <YEAR> <COPYRIGHT HOLDER>
```

```
Permission is hereby granted, free of charge, to any person obtaining a copy of  
this software and associated documentation files (the "Software"), to deal in  
the Software without restriction, including without limitation the rights to  
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of  
the Software, and to permit persons to whom the Software is furnished to do so,  
subject to the following conditions:
```

```
The above copyright notice and this permission notice shall be included in all  
copies or substantial portions of the Software.
```

```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS  
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS  
OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,  
WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN  
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

mpl1.0

```
*MOZILLA PUBLIC LICENSE*
*Version 1.0*
```

```
-----
*1. Definitions.*
```

```
*1.1. ``Contributor''* means each entity that creates or contributes
to the creation of Modifications.
```

```
*1.2. ``Contributor Version''* means the combination of the Original
Code, prior Modifications used by a Contributor, and the
Modifications made by that particular Contributor.
```

```
*1.3. ``Covered Code''* means the Original Code or Modifications or
the combination of the Original Code and Modifications, in each case
including portions thereof*.*
```

```
*1.4. ``Electronic Distribution Mechanism''* means a mechanism
generally accepted in the software development community for the
electronic transfer of data.
```

```
*1.5. ``Executable''* means Covered Code in any form other than
Source Code.
```

```
*1.6. ``Initial Developer''* means the individual or entity
identified as the Initial Developer in the Source Code notice
required by *Exhibit A*.
```

```
*1.7. ``Larger Work''* means a work which combines Covered Code or
portions thereof with code not governed by the terms of this License.
```

```
*1.8. ``License''* means this document.
```

```
*1.9. ``Modifications''* means any addition to or deletion from the
substance or structure of either the Original Code or any previous
Modifications. When Covered Code is released as a series of files, a
Modification is:
```

```
  *A.* Any addition to or deletion from the contents of a file
  containing Original Code or previous Modifications.
```

```
  *B.* Any new file that contains any part of the Original Code or
  previous Modifications.
```

```
*1.10. ``Original Code''* means Source Code of computer software
code which is described in the Source Code notice required by
*Exhibit A* as Original Code, and which, at the time of its release
under this License is not already Covered Code governed by this
License.
```

```
*1.11. ``Source Code''* means the preferred form of the Covered Code
for making modifications to it, including all modules it contains,
plus any associated interface definition files, scripts used to
control compilation and installation of an Executable, or a list of
source code differential comparisons against either the Original
Code or another well known, available Covered Code of the
```

Contributor's choice. The Source Code can be in a compressed or archival form, provided the appropriate decompression or de-archiving software is widely available for no charge.

1.12. ``You'' means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License or a future version of this License issued under Section 6.1. For legal entities, ``You'' includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, ``control'' means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of fifty percent (50%) or more of the outstanding shares or beneficial ownership of such entity.

2. Source Code License.

2.1. The Initial Developer Grant.

The Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license, subject to third party intellectual property claims:

(a) to use, reproduce, modify, display, perform, sublicense and distribute the Original Code (or portions thereof) with or without Modifications, or as part of a Larger Work; and

(b) under patents now or hereafter owned or controlled by Initial Developer, to make, have made, use and sell (``Utilize'') the Original Code (or portions thereof), but solely to the extent that any such patent is reasonably necessary to enable You to Utilize the Original Code (or portions thereof) and not to any greater extent that may be necessary to Utilize further Modifications or combinations.

2.2. Contributor Grant.

Each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license, subject to third party intellectual property claims:

(a) to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof) either on an unmodified basis, with other Modifications, as Covered Code or as part of a Larger Work; and

(b) under patents now or hereafter owned or controlled by Contributor, to Utilize the Contributor Version (or portions thereof), but solely to the extent that any such patent is reasonably necessary to enable You to Utilize the Contributor Version (or portions thereof), and not to any greater extent that may be necessary to Utilize further Modifications or combinations.

3. Distribution Obligations.

3.1. Application of License.

The Modifications which You create or to which You contribute are governed by the terms of this License, including without limitation

Section *2.2*. The Source Code version of Covered Code may be distributed only under the terms of this License or a future version of this License released under Section *6.1*, and You must include a copy of this License with every copy of the Source Code You distribute. You may not offer or impose any terms on any Source Code version that alters or restricts the applicable version of this License or the recipients' rights hereunder. However, You may include an additional document offering the additional rights described in Section *3.5*.

3.2. Availability of Source Code.

Any Modification which You create or to which You contribute must be made available in Source Code form under the terms of this License either on the same media as an Executable version or via an accepted Electronic Distribution Mechanism to anyone to whom you made an Executable version available; and if made available via Electronic Distribution Mechanism, must remain available for at least twelve (12) months after the date it initially became available, or at least six (6) months after a subsequent version of that particular Modification has been made available to such recipients. You are responsible for ensuring that the Source Code version remains available even if the Electronic Distribution Mechanism is maintained by a third party.

3.3. Description of Modifications.

You must cause all Covered Code to which you contribute to contain a file documenting the changes You made to create that Covered Code and the date of any change. You must include a prominent statement that the Modification is derived, directly or indirectly, from Original Code provided by the Initial Developer and including the name of the Initial Developer in (a) the Source Code, and (b) in any notice in an Executable version or related documentation in which You describe the origin or ownership of the Covered Code.

3.4. Intellectual Property Matters

(a) Third Party Claims.

If You have knowledge that a party claims an intellectual property right in particular functionality or code (or its utilization under this License), you must include a text file with the source code distribution titled ``LEGAL'' which describes the claim and the party making the claim in sufficient detail that a recipient will know whom to contact. If you obtain such knowledge after You make Your Modification available as described in Section *3.2*, You shall promptly modify the LEGAL file in all copies You make available thereafter and shall take other steps (such as notifying appropriate mailing lists or newsgroups) reasonably calculated to inform those who received the Covered Code that new knowledge has been obtained.

(b) Contributor APIs.

If Your Modification is an application programming interface and You own or control patents which are reasonably necessary to implement that API, you must also include this information in the LEGAL file.

3.5. Required Notices.

You must duplicate the notice in *Exhibit A* in each file of the

Source Code, and this License in any documentation for the Source Code, where You describe recipients' rights relating to Covered Code. If You created one or more Modification(s), You may add your name as a Contributor to the notice described in *Exhibit A*. If it is not possible to put such notice in a particular Source Code file due to its structure, then you must include such notice in a location (such as a relevant directory file) where a user would be likely to look for such a notice. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Code. However, You may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor. You must make it absolutely clear than any such warranty, support, indemnity or liability obligation is offered by You alone, and You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of warranty, support, indemnity or liability terms You offer.

3.6. Distribution of Executable Versions.

You may distribute Covered Code in Executable form only if the requirements of Section *3.1-3.5* have been met for that Covered Code, and if You include a notice stating that the Source Code version of the Covered Code is available under the terms of this License, including a description of how and where You have fulfilled the obligations of Section *3.2*. The notice must be conspicuously included in any notice in an Executable version, related documentation or collateral in which You describe recipients' rights relating to the Covered Code. You may distribute the Executable version of Covered Code under a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable version does not attempt to limit or alter the recipient's rights in the Source Code version from the rights set forth in this License. If You distribute the Executable version under a different license You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or any Contributor. You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of any such terms You offer.

3.7. Larger Works.

You may create a Larger Work by combining Covered Code with other code not governed by the terms of this License and distribute the Larger Work as a single product. In such a case, You must make sure the requirements of this License are fulfilled for the Covered Code.

4. Inability to Comply Due to Statute or Regulation.

If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Code due to statute or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be

included in the LEGAL file described in Section *3.4* and must be included with all distributions of the Source Code. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

5. Application of this License.

This License applies to code to which the Initial Developer has attached the notice in *Exhibit A*, and to related Covered Code.

6. Versions of the License.

6.1. New Versions.

Netscape Communications Corporation (``Netscape'') may publish revised and/or new versions of the License from time to time. Each version will be given a distinguishing version number.

6.2. Effect of New Versions.

Once Covered Code has been published under a particular version of the License, You may always continue to use it under the terms of that version. You may also choose to use such Covered Code under the terms of any subsequent version of the License published by Netscape. No one other than Netscape has the right to modify the terms applicable to Covered Code created under this License.

6.3. Derivative Works.

If you create or use a modified version of this License (which you may only do in order to apply it to code which is not already Covered Code governed by this License), you must (a) rename Your license so that the phrases ``Mozilla'', ``MOZILLAPL'', ``MOZPL'', ``Netscape'', ``NPL'' or any confusingly similar phrase do not appear anywhere in your license and (b) otherwise make it clear that your version of the license contains terms which differ from the Mozilla Public License and Netscape Public License. (Filling in the name of the Initial Developer, Original Code or Contributor in the notice described in *Exhibit A* shall not of themselves be deemed to be modifications of this License.)

7. DISCLAIMER OF WARRANTY.

COVERED CODE IS PROVIDED UNDER THIS LICENSE ON AN ``AS IS'' BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE COVERED CODE IS FREE OF DEFECTS, MERCHANTABILITY, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE COVERED CODE IS WITH YOU. SHOULD ANY COVERED CODE PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL DEVELOPER OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED CODE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

8. TERMINATION.

This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to

cure such breach within 30 days of becoming aware of the breach. All sublicenses to the Covered Code which are properly granted shall survive any termination of this License. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

9. LIMITATION OF LIABILITY.

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED CODE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO YOU OR ANY OTHER PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THAT EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

10. U.S. GOVERNMENT END USERS.

The Covered Code is a ``commercial item,'' as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of ``commercial computer software'' and ``commercial computer software documentation,'' as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire Covered Code with only those rights set forth herein.

11. MISCELLANEOUS.

This License represents the complete agreement concerning subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. This License shall be governed by California law provisions (except to the extent applicable law, if any, provides otherwise), excluding its conflict-of-law provisions. With respect to disputes in which at least one party is a citizen of, or an entity chartered or registered to do business in, the United States of America: (a) unless otherwise agreed in writing, all disputes relating to this License (excepting any dispute relating to intellectual property rights) shall be subject to final and binding arbitration, with the losing party paying all costs of arbitration; (b) any arbitration relating to this Agreement shall be held in Santa Clara County, California, under the auspices of JAMS/EndDispute; and (c) any litigation relating to this Agreement shall be subject to the jurisdiction of the Federal Courts of the Northern District of California, with venue lying in Santa Clara County, California, with the losing party responsible for costs, including without limitation, court costs and reasonable attorneys fees and expenses. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not apply to

this License.

12. RESPONSIBILITY FOR CLAIMS.

Except in cases where another Contributor has failed to comply with Section *3.4*, You are responsible for damages arising, directly or indirectly, out of Your utilization of rights under this License, based on the number of copies of Covered Code you made available, the revenues you received from utilizing such rights, and other relevant factors. You agree to work with affected parties to distribute responsibility on an equitable basis.

EXHIBIT A.

``The contents of this file are subject to the Mozilla Public License Version 1.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.mozilla.org/MPL/>

Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License.

The Original Code is _____.

The Initial Developer of the Original Code is _____.
Portions created by _____
are Copyright (C) _____ . All Rights Reserved.

Contributor(s): _____.''

mpl1.1

MOZILLA PUBLIC LICENSE

Version 1.1

1. Definitions.

1.0.1. "Commercial Use" means distribution or otherwise making the Covered Code available to a third party.

1.1. "Contributor" means each entity that creates or contributes to the creation of Modifications.

1.2. "Contributor Version" means the combination of the Original Code, prior Modifications used by a Contributor, and the Modifications made by that particular Contributor.

1.3. "Covered Code" means the Original Code or Modifications or the combination of the Original Code and Modifications, in each case including portions thereof.

1.4. "Electronic Distribution Mechanism" means a mechanism generally accepted in the software development community for the electronic transfer of data.

1.5. "Executable" means Covered Code in any form other than Source Code.

1.6. "Initial Developer" means the individual or entity identified as the Initial Developer in the Source Code notice required by Exhibit A.

1.7. "Larger Work" means a work which combines Covered Code or portions thereof with code not governed by the terms of this License.

1.8. "License" means this document.

1.8.1. "Licensable" means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently acquired, any and all of the rights conveyed herein.

1.9. "Modifications" means any addition to or deletion from the substance or structure of either the Original Code or any previous Modifications. When Covered Code is released as a series of files, a Modification is:

A. Any addition to or deletion from the contents of a file containing Original Code or previous Modifications.

B. Any new file that contains any part of the Original Code or previous Modifications.

1.10. "Original Code" means Source Code of computer software code which is described in the Source Code notice required by Exhibit A as Original Code, and which, at the time of its release under this License is not already Covered Code governed by this License.

1.10.1. "Patent Claims" means any patent claim(s), now owned or hereafter acquired, including without limitation, method, process, and apparatus claims, in any patent Licensable by grantor.

1.11. "Source Code" means the preferred form of the Covered Code for making modifications to it, including all modules it contains, plus any associated interface definition files, scripts used to control compilation and installation of an Executable, or source code differential comparisons against either the Original Code or another well known, available Covered Code of the Contributor's choice. The Source Code can be in a compressed or archival form, provided the appropriate decompression or de-archiving software is widely available for no charge.

1.12. "You" (or "Your") means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License or a future version of this License issued under Section 6.1. For legal entities, "You" includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, "control" means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

2. Source Code License.

2.1. The Initial Developer Grant.

The Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license, subject to third party intellectual property claims:

(a) under intellectual property rights (other than patent or trademark) Licensable by Initial Developer to use, reproduce, modify, display, perform, sublicense and distribute the Original Code (or portions thereof) with or without Modifications, and/or as part of a Larger Work; and

(b) under Patents Claims infringed by the making, using or selling of Original Code, to make, have made, use, practice, sell, and offer for sale, and/or otherwise dispose of the Original Code (or portions thereof).

(c) the licenses granted in this Section 2.1(a) and (b) are effective on the date Initial Developer first distributes Original Code under the terms of this License.

(d) Notwithstanding Section 2.1(b) above, no patent license is granted: 1) for code that You delete from the Original Code; 2) separate from the Original Code; or 3) for infringements caused by: i) the modification of the Original Code or ii) the combination of the Original Code with other software or devices.

2.2. Contributor Grant.

Subject to third party intellectual property claims, each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license

(a) under intellectual property rights (other than patent or trademark) Licensable by Contributor, to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof) either on an unmodified basis, with other Modifications, as Covered Code and/or as part of a Larger Work; and

(b) under Patent Claims infringed by the making, using, or selling of Modifications made by that Contributor either alone and/or in combination with its Contributor Version (or portions of such combination), to make, use, sell, offer for sale, have made, and/or otherwise dispose of: 1) Modifications made by that Contributor (or portions thereof); and 2) the combination of Modifications made by that Contributor with its Contributor Version (or portions of such combination).

(c) the licenses granted in Sections 2.2(a) and 2.2(b) are effective on the date Contributor first makes Commercial Use of the Covered Code.

(d) Notwithstanding Section 2.2(b) above, no patent license is granted: 1) for any code that Contributor has deleted from the Contributor Version; 2) separate from the Contributor Version; 3) for infringements caused by: i) third party modifications of Contributor Version or ii) the combination of Modifications made by that Contributor with other software (except as part of the Contributor Version) or other devices; or 4) under Patent Claims infringed by Covered Code in the absence of Modifications made by that Contributor.

3. Distribution Obligations.

3.1. Application of License.

The Modifications which You create or to which You contribute are governed by the terms of this License, including without limitation Section 2.2. The Source Code version of Covered Code may be distributed only under the terms of this License or a future version of this License released under Section 6.1, and You must include a copy of this License with every copy of the Source Code You distribute. You may not offer or impose any terms on any Source Code version that alters or restricts the applicable version of this License or the recipients' rights hereunder. However, You may include an additional document offering the additional rights described in Section 3.5.

3.2. Availability of Source Code.

Any Modification which You create or to which You contribute must be made available in Source Code form under the terms of this License either on the same media as an Executable version or via an accepted Electronic Distribution Mechanism to anyone to whom you made an Executable version available; and if made available via Electronic Distribution Mechanism, must remain available for at least twelve (12) months after the date it initially became available, or at least six (6) months after a subsequent version of that particular Modification has been made available to such recipients. You are responsible for ensuring that the Source Code version remains available even if the Electronic Distribution Mechanism is maintained by a third party.

3.3. Description of Modifications.

You must cause all Covered Code to which You contribute to contain a file documenting the changes You made to create that Covered Code and the date of any change. You must include a prominent statement that the Modification is derived, directly or indirectly, from Original Code provided by the Initial Developer and including the name of the Initial Developer in (a) the Source Code, and (b) in any notice in an Executable version or related documentation in which You describe the origin or ownership of the Covered Code.

3.4. Intellectual Property Matters

(a) Third Party Claims.

If Contributor has knowledge that a license under a third party's intellectual property rights is required to exercise the rights granted by such Contributor under Sections 2.1 or 2.2, Contributor must include a text file with the Source Code distribution titled "LEGAL" which describes the claim and the party making the claim in sufficient detail that a recipient will know whom to contact. If Contributor obtains such knowledge after the Modification is made available as described in Section 3.2, Contributor shall promptly modify the LEGAL file in all copies Contributor makes available thereafter and shall take other steps (such as notifying appropriate mailing lists or newsgroups) reasonably calculated to inform those who received the Covered Code that new knowledge has been obtained.

(b) Contributor APIs.

If Contributor's Modifications include an application programming interface and Contributor has knowledge of patent licenses which are reasonably necessary to implement that API, Contributor must also include this information in the LEGAL file.

(c) Representations.

Contributor represents that, except as disclosed pursuant to Section 3.4(a) above, Contributor believes that Contributor's Modifications are Contributor's original creation(s) and/or Contributor has sufficient rights to grant the rights conveyed by this License.

3.5. Required Notices.

You must duplicate the notice in Exhibit A in each file of the Source Code. If it is not possible to put such notice in a particular Source Code file due to its structure, then You must include such notice in a location (such as a relevant directory) where a user would be likely to look for such a notice. If You created one or more Modification(s) You may add your name as a Contributor to the notice described in Exhibit A. You must also duplicate this License in any documentation for the Source Code where You describe recipients' rights or ownership rights relating to Covered Code. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Code. However, You may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor. You must make it absolutely clear than any such warranty, support, indemnity or liability obligation is offered by You alone, and You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of warranty, support, indemnity or liability terms You offer.

3.6. Distribution of Executable Versions.

You may distribute Covered Code in Executable form only if the requirements of Section 3.1-3.5 have been met for that Covered Code, and if You include a notice stating that the Source Code version of the Covered Code is available under the terms of this License, including a description of how and where You have fulfilled the obligations of Section 3.2. The notice must be conspicuously included in any notice in an Executable version, related documentation or collateral in which You describe recipients' rights relating to the Covered Code. You may distribute the Executable version of Covered Code or ownership rights under a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable version does not attempt to limit or alter the recipient's rights in the Source Code version from the rights set forth in this License. If You distribute the Executable version under a different license You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or any Contributor. You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of any such terms You offer.

3.7. Larger Works.

You may create a Larger Work by combining Covered Code with other code not governed by the terms of this License and distribute the Larger Work as a single product. In such a case, You must make sure the requirements of this License are fulfilled for the Covered Code.

4. Inability to Comply Due to Statute or Regulation.

If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Code due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be included in the LEGAL file described in Section 3.4 and must be included with all distributions of the Source Code. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

5. Application of this License.

This License applies to code to which the Initial Developer has attached the notice in Exhibit A and to related Covered Code.

6. Versions of the License.

6.1. New Versions.

Netscape Communications Corporation ("Netscape") may publish revised and/or new versions of the License from time to time. Each version will be given a distinguishing version number.

6.2. Effect of New Versions.

Once Covered Code has been published under a particular version of the License, You may always continue to use it under the terms of that version. You may also choose to use such Covered Code under the terms of any subsequent version of the License published by Netscape. No one other than Netscape has the right to modify the terms applicable to Covered Code created under this License.

6.3. Derivative Works.

If You create or use a modified version of this License (which you may only do in order to apply it to code which is not already Covered Code governed by this License), You must (a) rename Your license so that the phrases "Mozilla", "MOZILLAPL", "MOZPL", "Netscape", "MPL", "NPL" or any confusingly similar phrase do not appear in your license (except to note that your license differs from this License) and (b) otherwise make it clear that Your version of the license contains terms which differ from the Mozilla Public License and Netscape Public License. (Filling in the name of the Initial Developer, Original Code or Contributor in the notice described in Exhibit A shall not of themselves be deemed to be modifications of this License.)

7. DISCLAIMER OF WARRANTY.

COVERED CODE IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE COVERED CODE IS FREE OF DEFECTS, MERCHANTABILITY, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE COVERED CODE IS WITH YOU. SHOULD ANY COVERED CODE PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL DEVELOPER OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED CODE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

8. TERMINATION.

8.1. This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. All sublicenses to the Covered Code which are properly granted shall survive any termination of this License. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

8.2. If You initiate litigation by asserting a patent infringement claim (excluding declaratory judgment actions) against Initial Developer or a Contributor (the Initial Developer or Contributor against whom You file such action is referred to as "Participant") alleging that:

(a) such Participant's Contributor Version directly or indirectly infringes any patent, then any and all rights granted by such Participant to You under Sections 2.1 and/or 2.2 of this License shall, upon 60 days notice from Participant terminate prospectively, unless if within 60 days after receipt of notice You either: (i) agree in writing to pay Participant a mutually agreeable reasonable royalty for Your past and future use of Modifications made by such Participant, or (ii) withdraw Your litigation claim with respect to the Contributor Version against such Participant. If within 60 days of notice, a reasonable royalty and payment arrangement are not mutually agreed upon in writing by the parties or the litigation claim is not withdrawn, the rights granted by Participant to You under Sections 2.1 and/or 2.2 automatically terminate at the expiration of the 60 day notice period specified above.

(b) any software, hardware, or device, other than such Participant's Contributor Version, directly or indirectly infringes any patent, then any rights granted to You by such Participant under Sections 2.1(b) and 2.2(b) are revoked effective as of the date You first made, used, sold, distributed, or had made, Modifications made by that Participant.

8.3. If You assert a patent infringement claim against Participant alleging that such Participant's Contributor Version directly or indirectly infringes any patent where such claim is resolved (such as by license or settlement) prior to the initiation of patent infringement litigation, then the reasonable value of the licenses granted by such Participant under Sections 2.1 or 2.2 shall be taken into account in determining the amount or value of any payment or license.

8.4. In the event of termination under Sections 8.1 or 8.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or any distributor hereunder prior to termination shall survive termination.

9. LIMITATION OF LIABILITY.

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED CODE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER

COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

10. U.S. GOVERNMENT END USERS.

The Covered Code is a "commercial item," as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of "commercial computer software" and "commercial computer software documentation," as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire Covered Code with only those rights set forth herein.

11. MISCELLANEOUS.

This License represents the complete agreement concerning subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. This License shall be governed by California law provisions (except to the extent applicable law, if any, provides otherwise), excluding its conflict-of-law provisions. With respect to disputes in which at least one party is a citizen of, or an entity chartered or registered to do business in the United States of America, any litigation relating to this License shall be subject to the jurisdiction of the Federal Courts of the Northern District of California, with venue lying in Santa Clara County, California, with the losing party responsible for costs, including without limitation, court costs and reasonable attorneys' fees and expenses. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not apply to this License.

12. RESPONSIBILITY FOR CLAIMS.

As between Initial Developer and the Contributors, each party is responsible for claims and damages arising, directly or indirectly, out of its utilization of rights under this License and You agree to work with Initial Developer and Contributors to distribute such responsibility on an equitable basis. Nothing herein is intended or shall be deemed to constitute any admission of liability.

13. MULTIPLE-LICENSED CODE.

Initial Developer may designate portions of the Covered Code as "Multiple-Licensed". "Multiple-Licensed" means that the Initial Developer permits you to utilize portions of the Covered Code under Your choice of the MPL or the alternative licenses, if any, specified by the Initial Developer in the file described in Exhibit A.

EXHIBIT A -Mozilla Public License.

``The contents of this file are subject to the Mozilla Public License

Version 1.1 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at <https://www.mozilla.org/MPL/>

Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License.

The Original Code is _____.

The Initial Developer of the Original Code is _____.
Portions created by _____ are Copyright (C) _____
_____. All Rights Reserved.

Contributor(s): _____.

Alternatively, the contents of this file may be used under the terms of the _____ license (the "[____] License"), in which case the provisions of [_____] License are applicable instead of those above. If you wish to allow use of your version of this file only under the terms of the [____] License and not to allow others to use your version of this file under the MPL, indicate your decision by deleting the provisions above and replace them with the notice and other provisions required by the [____] License. If you do not delete the provisions above, a recipient may use your version of this file under either the MPL or the [____] License."

[NOTE: The text of this Exhibit A may differ slightly from the text of the notices in the Source Code files of the Original Code. You should use the text of this Exhibit A rather than the text found in the Original Code Source Code for Your Modifications.]

nux

Nux License Agreement

Copyright (c) 2005, The Regents of the University of California, through Lawrence Berkeley National Laboratory (subject to receipt of any required approvals from the U.S. Dept. of Energy). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

(1) Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

(2) Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

(3) Neither the name of the University of California, Lawrence Berkeley National Laboratory, U.S. Dept. of Energy nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,

THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

You are under no obligation whatsoever to provide any bug fixes, patches, or upgrades to the features, functionality or performance of the source code ("Enhancements") to anyone; however, if you choose to make your Enhancements available either publicly, or directly to Lawrence Berkeley National Laboratory, without imposing a separate written license agreement for such Enhancements, then you hereby grant the following license: a non-exclusive, royalty-free perpetual license to install, use, modify, prepare derivative works, incorporate into other computer software, distribute, and sublicense such enhancements or derivative works thereof, in binary and source code form.

ofl-1.1

Copyright (c) <dates>, <Copyright Holder> (<URL|email>),
with Reserved Font Name <Reserved Font Name>.
Copyright (c) <dates>, <additional Copyright Holder> (<URL|email>),
with Reserved Font Name <additional Reserved Font Name>.
Copyright (c) <dates>, <additional Copyright Holder> (<URL|email>).

This Font Software is licensed under the SIL Open Font License, Version 1.1.
This license is copied below, and is also available with a FAQ at:
<http://scripts.sil.org/OFL>

SIL OPEN FONT LICENSE Version 1.1 - 26 February 2007

PREAMBLE

The goals of the Open Font License (OFL) are to stimulate worldwide development of collaborative font projects, to support the font creation efforts of academic and linguistic communities, and to provide a free and open framework in which fonts may be shared and improved in partnership with others.

The OFL allows the licensed fonts to be used, studied, modified and redistributed freely as long as they are not sold by themselves. The fonts, including any derivative works, can be bundled, embedded, redistributed and/or sold with any software provided that any reserved names are not used by derivative works. The fonts and derivatives, however, cannot be released under any other type of license. The requirement for fonts to remain under this license does not apply to any document created using the fonts or their derivatives.

DEFINITIONS

"Font Software" refers to the set of files released by the Copyright Holder(s) under this license and clearly marked as such. This may include source files, build scripts and documentation.

"Reserved Font Name" refers to any names specified as such after the copyright statement(s).

"Original Version" refers to the collection of Font Software components as distributed by the Copyright Holder(s).

"Modified Version" refers to any derivative made by adding to, deleting, or substituting -- in part or in whole -- any of the components of the Original Version, by changing formats or by porting the Font Software to a new environment.

"Author" refers to any designer, engineer, programmer, technical writer or other person who contributed to the Font Software.

PERMISSION & CONDITIONS

Permission is hereby granted, free of charge, to any person obtaining a copy of the Font Software, to use, study, copy, merge, embed, modify, redistribute, and sell modified and unmodified copies of the Font Software, subject to the following conditions:

- 1) Neither the Font Software nor any of its individual components, in Original or Modified Versions, may be sold by itself.
- 2) Original or Modified Versions of the Font Software may be bundled, redistributed and/or sold with any software, provided that each copy contains the above copyright notice and this license. These can be included either as stand-alone text files, human-readable headers or in the appropriate machine-readable metadata fields within text or binary files as long as those fields can be easily viewed by the user.
- 3) No Modified Version of the Font Software may use the Reserved Font Name(s) unless explicit written permission is granted by the corresponding Copyright Holder. This restriction only applies to the primary font name as presented to the users.
- 4) The name(s) of the Copyright Holder(s) or the Author(s) of the Font Software shall not be used to promote, endorse or advertise any Modified Version, except to acknowledge the contribution(s) of the Copyright Holder(s) and the Author(s) or with their explicit written permission.
- 5) The Font Software, modified or unmodified, in part or in whole, must be distributed entirely under this license, and must not be distributed under any other license. The requirement for fonts to remain under this license does not apply to any document created using the Font Software.

TERMINATION

This license becomes null and void if any of the above conditions are not met.

DISCLAIMER

THE FONT SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF COPYRIGHT, PATENT, TRADEMARK, OR OTHER RIGHT. IN NO EVENT SHALL THE COPYRIGHT HOLDER BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, INCLUDING ANY GENERAL, SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING

FROM, OUT OF THE USE OR INABILITY TO USE THE FONT SOFTWARE OR FROM OTHER DEALINGS IN THE FONT SOFTWARE.

otn

Oracle Technology Network Development and Distribution License Terms

Export Controls on the Programs

Selecting the "Accept License Agreement" button is a confirmation of your agreement that you comply, now and during the trial term, with each of the following statements:

-You are not a citizen, national, or resident of, and are not under control of, the government of Cuba, Iran, Sudan, Libya, North Korea, Syria, nor any country to which the United States has prohibited export.

-You will not download or otherwise export or re-export the Programs, directly or indirectly, to the above mentioned countries nor to citizens, nationals or residents of those countries.

-You are not listed on the United States Department of Treasury lists of Specially Designated Nationals, Specially Designated Terrorists, and Specially Designated Narcotic Traffickers, nor are you listed on the United States Department of Commerce Table of Denial Orders.

You will not download or otherwise export or re-export the Programs, directly or indirectly, to persons on the above mentioned lists.

You will not use the Programs for, and will not allow the Programs to be used for, any purposes prohibited by United States law, including, without limitation, for the development, design, manufacture or production of nuclear, chemical or biological weapons of mass destruction.

EXPORT RESTRICTIONS

You agree that U.S. export control laws and other applicable export and import laws govern your use of the programs, including technical data; additional information can be found on Oracle®'s Global Trade Compliance web site (<http://www.oracle.com/products/export>).

You agree that neither the programs nor any direct product thereof will be exported, directly, or indirectly, in violation of these laws, or will be used for any purpose prohibited by these laws including, without limitation, nuclear, chemical, or biological weapons proliferation.

Oracle Employees: Under no circumstances are Oracle Employees authorized to download software for the purpose of distributing it to customers. Oracle products are available to employees for internal use or demonstration purposes only. In keeping with Oracle's trade compliance obligations under U.S. and applicable multilateral law, failure to comply with this policy could result in disciplinary action up to and including termination.

Note: You are bound by the Oracle Technology Network ("OTN") License Agreement terms. The OTN License Agreement terms also apply to all updates you receive

under your Technology Track subscription.

The OTN License Agreement terms below supercede any shrinkwrap license on the OTN Technology Track software CDs and previous OTN License terms (including the Oracle Program License as modified by the OTN Program Use Certificate).

Oracle Technology Network Development and Distribution License Agreement

"We," "us," and "our" refers to Oracle America, Inc., for and on behalf of itself and its subsidiaries and affiliates under common control. "You" and "your" refers to the individual or entity that wishes to use the programs from Oracle. "Programs" refers to the software product you wish to download and use and program documentation. "License" refers to your right to use the programs under the terms of this agreement. This agreement is governed by the substantive and procedural laws of California. You and Oracle agree to submit to the exclusive jurisdiction of, and venue in, the courts of San Francisco, San Mateo, or Santa Clara counties in California in any dispute arising out of or relating to this agreement.

We are willing to license the programs to you only upon the condition that you accept all of the terms contained in this agreement. Read the terms carefully and select the "Accept" button at the bottom of the page to confirm your acceptance. If you are not willing to be bound by these terms, select the "Do Not Accept" button and the registration process will not continue.

License Rights

We grant you a nonexclusive, nontransferable limited license to use the programs: (a) for purposes of developing, testing, prototyping and running applications you have developed for your own internal data processing operations; (b) to distribute the programs with applications you have developed to your customers provided that each such licensee agrees to license terms consistent with the terms of this Agreement, you do not charge your end users any additional fees for the use of the programs, and your end users may only use the programs to run your applications for their own business operations; and (c) to use the programs to provide third party demonstrations and training. You are not permitted to use the programs for any purpose other than as permitted under this Agreement. If you want to use the programs for any purpose other than as expressly permitted under this agreement you must contact us, or an Oracle reseller, to obtain the appropriate license. We may audit your use and distribution of the programs. Program documentation is either shipped with the programs, or documentation may accessed online at <http://www.oracle.com/technetwork/indexes/documentation/index.html>.

Ownership and Restrictions

We retain all ownership and intellectual property rights in the programs. You may make a sufficient number of copies of the programs for the licensed use and one copy of the programs for backup purposes.

You may not:

- use the programs for any purpose other than as provided above;
- distribute the programs unless accompanied with your applications;
- charge your end users for use of the programs;
- remove or modify any program markings or any notice of our proprietary rights;
- use the programs to provide third party training on the content and/or functionality of the programs, except for training your licensed users;
- assign this agreement or give the programs, program access or an interest in

the programs to any individual or entity except as provided under this agreement;

- cause or permit reverse engineering (unless required by law for interoperability), disassembly or decompilation of the programs;
- disclose results of any program benchmark tests without our prior consent.

Program Distribution

We grant you a nonexclusive, nontransferable right to copy and distribute the programs to your end users provided that you do not charge your end users for use of the programs and provided your end users may only use the programs to run your applications for their business operations. Prior to distributing the programs you shall require your end users to execute an agreement binding them to terms consistent with those contained in this section and the sections of this agreement entitled "License Rights," "Ownership and Restrictions," "Export," "Disclaimer of Warranties and Exclusive Remedies," "No Technical Support," "End of Agreement," "Relationship Between the Parties," and "Open Source." You must also include a provision stating that your end users shall have no right to distribute the programs, and a provision specifying us as a third party beneficiary of the agreement. You are responsible for obtaining these agreements with your end users.

You agree to: (a) defend and indemnify us against all claims and damages caused by your distribution of the programs in breach of this agreements and/or failure to include the required contractual provisions in your end user agreement as stated above; (b) keep executed end user agreements and records of end user information including name, address, date of distribution and identity of programs distributed; (c) allow us to inspect your end user agreements and records upon request; and, (d) enforce the terms of your end user agreements so as to effect a timely cure of any end user breach, and to notify us of any breach of the terms.

Export

You agree that U.S. export control laws and other applicable export and import laws govern your use of the programs, including technical data; additional information can be found on Oracle's Global Trade Compliance web site located at <http://www.oracle.com/products/export/index.html?content.html>. You agree that neither the programs nor any direct product thereof will be exported, directly, or indirectly, in violation of these laws, or will be used for any purpose prohibited by these laws including, without limitation, nuclear, chemical, or biological weapons proliferation.

Disclaimer of Warranty and Exclusive Remedies

THE PROGRAMS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. WE FURTHER DISCLAIM ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT.

IN NO EVENT SHALL WE BE LIABLE FOR ANY INDIRECT, INCIDENTAL, SPECIAL, PUNITIVE OR CONSEQUENTIAL DAMAGES, OR DAMAGES FOR LOSS OF PROFITS, REVENUE, DATA OR DATA USE, INCURRED BY YOU OR ANY THIRD PARTY, WHETHER IN AN ACTION IN CONTRACT OR TORT, EVEN IF WE HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. OUR ENTIRE LIABILITY FOR DAMAGES HEREUNDER SHALL IN NO EVENT EXCEED ONE THOUSAND DOLLARS (U.S. \$1,000).

No Technical Support

Our technical support organization will not provide technical support, phone support, or updates to you for the programs licensed under this agreement.

Restricted Rights

If you distribute a license to the United States government, the programs, including documentation, shall be considered commercial computer software and you will place a legend, in addition to applicable copyright notices, on the documentation, and on the media label, substantially similar to the following:

NOTICE OF RESTRICTED RIGHTS

"Programs delivered subject to the DOD FAR Supplement are 'commercial computer software' and use, duplication, and disclosure of the programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, programs delivered subject to the Federal Acquisition Regulations are 'restricted computer software' and use, duplication, and disclosure of the programs, including documentation, shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software-Restricted Rights (June 1987). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065."

End of Agreement

You may terminate this agreement by destroying all copies of the programs. We have the right to terminate your right to use the programs if you fail to comply with any of the terms of this agreement, in which case you shall destroy all copies of the programs.

Relationship Between the Parties

The relationship between you and us is that of licensee/licensor. Neither party will represent that it has any authority to assume or create any obligation, express or implied, on behalf of the other party, nor to represent the other party as agent, employee, franchisee, or in any other capacity. Nothing in this agreement shall be construed to limit either party's right to independently develop or distribute software that is functionally similar to the other party's products, so long as proprietary information of the other party is not included in such software.

Open Source

"Open Source" software - software available without charge for use, modification and distribution - is often licensed under terms that require the user to make the user's modifications to the Open Source software or any software that the user 'combines' with the Open Source software freely available in source code form. If you use Open Source software in conjunction with the programs, you must ensure that your use does not: (i) create, or purport to create, obligations of us with respect to the Oracle programs; or (ii) grant, or purport to grant, to any third party any rights to or immunities under our intellectual property or proprietary rights in the Oracle programs. For example, you may not develop a software program using an Oracle program and an Open Source program where such use results in a program file(s) that contains code from both the Oracle program and the Open Source program (including without limitation libraries) if the Open Source program is licensed under a license that requires any "modifications" be made freely available. You also may not combine the Oracle program with programs licensed under the GNU General Public License ("GPL") in any manner that could cause, or could be interpreted or asserted to cause, the Oracle program or any modifications thereto to become subject to the terms of the GPL.

Entire Agreement

You agree that this agreement is the complete agreement for the programs and licenses, and this agreement supersedes all prior or contemporaneous agreements or representations. If any term of this agreement is found to be invalid or unenforceable, the remaining provisions will remain effective.

Last updated: 01/24/09

Should you have any questions concerning this License Agreement, or if you desire to contact Oracle for any reason, please write:

Oracle America, Inc.
500 Oracle Parkway,
Redwood City, CA 94065

Oracle may contact you to ask if you had a satisfactory experience installing and using this OTN software download.

postgresql-bsd

Copyright (c) 1997-2011, PostgreSQL Global Development Group
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the PostgreSQL Global Development Group nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

tcllib

license.tcllib

This software is copyrighted by Ajuba Solutions and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where

they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

GOVERNMENT USE: If you are acquiring this software on behalf of the U.S. government, the Government shall have only "Restricted Rights" in the software and related documentation as defined in the Federal Acquisition Regulations (FARs) in Clause 52.227.19 (c) (2). If you are acquiring the software on behalf of the Department of Defense, the software shall be classified as "Commercial Computer Software" and the Government shall have only "Restricted Rights" as defined in Clause 252.227-7013 (c) (1) of DFARS. Notwithstanding the foregoing, the authors grant the U.S. Government and others acting in its behalf permission to use and distribute the software in accordance with the terms specified in this license.

tktk

Following is the original agreement for the Tcl/Tk software from Sun Microsystems.

This software is copyrighted by the Regents of the University of California, Sun Microsystems, Inc., and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE

IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

GOVERNMENT USE: If you are acquiring this software on behalf of the U.S. government, the Government shall have only "Restricted Rights" in the software and related documentation as defined in the Federal Acquisition Regulations (FARs) in Clause 52.227.19 (c) (2). If you are acquiring the software on behalf of the Department of Defense, the software shall be classified as "Commercial Computer Software" and the Government shall have only "Restricted Rights" as defined in Clause 252.227-7013 (c) (1) of DFARS. Notwithstanding the foregoing, the authors grant the U.S. Government and others acting in its behalf permission to use and distribute the software in accordance with the terms specified in this license.

ucb

license.ucb

Portions of Jacl and Tcl Blend are
Copyright (c) 1997-1999 The Regents of the University of California.
All rights reserved.

Permission is hereby granted, without written agreement and without license or royalty fees, to use, copy, modify, and distribute this software and its documentation for any purpose, provided that the above copyright notice and the following two paragraphs appear in all copies of this software.

IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

Index

Symbols

- * What's New in Squore 17.1?
 - Handling of offline XSL Transformations, 208

A

- ABAP, 3
- Acceptance Testing, 221
- Accessibility, 221
- Accuracy, 221
- Accuracy of Measurement, 222
- Acquirer, 222
- Action, 223
- Activity, 223
- Actor, 224
- ADA, 13
- Adaptability, 224
- additional_param, 199
- Agreement, 225
- Analysability, 225
- Analysis Model, 225
- Architecture, 225
- arg, 215
- Attractiveness, 226
- Attribute, 226
- auxiliarypath, 188
- Availability, 227

B

- Baseline, 228
- Base Measure, 227
- baseName, 213
- branch, 174
- Branch, 228
- Branch Coverage, 229
- Branch Testing, 229
- Budget, 230
- Build, 230

C

- C, 23
- Call Graph, 230
- Capability Maturity Model, 231
- Certification, 231
- Certification Criteria, 231
- Changeability, 233
- changeable, 214
- Change Control Board, 232
- Change Control System, 232
- Change Management, 232
- clAlg, 199
- class_dir, 188

- clBw, 199
- clFR, 199
- clIOS, 199
- clRen, 199
- clTxt, 199
- CMMi, 382
- COBOL, 35, 207
- Code, 233
- Code Coverage, 234
- Code Freeze, 234
- Code Review, 234
- Code Verification, 234
- Coding, 235
- Co-existence, 233
- Cohesion, 235
- Commercial-Off-The-Shelf (COTS), 236
- commit, 176
- Commit, 236
- Commitment, 236
- compact_folder, 198
- Compatibility, 237
- Complexity, 237
- Component, 238
- Conciseness, 238
- Condition, 239
- configFile, 185, 185, 194
- Configuration, 239
- Configuration Control, 239
- Configuration Item, 240
- Configuration Management, 241
- Configuration Management System, 241
- Conflict, 242
- Conformance, 242
- Connectivity, 243
- Consistency, 243
- Constraint, 243
- Content Coupling, 244
- Context of Use, 244
- Contract, 244
- Control Coupling, 245
- Control Flow, 245
- Control Flow Diagram, 245
- Convention, 246
- Correctability, 246
- Correctness, 246
- Coupling, 247
- Coverage, 247
- CPP, 48
- Criteria, 248
- Criticality, 248
- CSHARP, 60

csv, 181, 182, 188, 191, 203, 203, 203, 204, 205, 206
Customer, 249
Custom Software, 248

D

Data, 249
Database, 253
Data Coupling, 250
Data Flow, 250
Data Flow Diagram, 251
Data Management, 251
Data Model, 251
Data Processing, 252
Data Provider, 252
Data Providers
 AntiC, 181
 Automotive Coverage Import, 181
 Automotive Tag Import, 181
 BullseyeCoverage Code Coverage Analyzer, 182
 Cantata, 184
 CheckStyle, 184
 CheckStyle (plugin), 184
 CheckStyle for SQALE (plugin), 185
 Cobertura, 186
 CodeSniffer, 202
 CodeSonar, 186
 Compiler, 186
 Configuration Checker, 202
 Coverity, 187
 CPD, 182
 Cppcheck, 182
 Cppcheck (plugin), 183
 CPPTest, 183
 Csv, 209, 423
 csv_findings, 209, 426
 Csv Coverage Import, 203
 CSV Findings, 203
 CsvPerl, 209, 427
 Csv Tag Import, 203
 Csv Test Results Import, 204
 ExcelMetrics, 209, 440
 FindBugs, 187
 FindBugs (plugin), 187
 FindingsPerl, 209, 436
 Frameworks, 209
 Function Relaxer, 188
 FxCop, 188
 GCov, 189
 Generic, 209, 429
 GenericPerl, 209, 433
 GNATcheck, 189
 GNATCompiler, 190
 JaCoCo, 190
 JUnit, 190
 Klocwork, 191
 MemUsage, 191
 MISRA Rule Checking using PC-lint, 193
 MISRA Rule Checking with QAC, 195
 NCover, 192
 Oracle PLSQL compiler Warning checker, 192
 OSLC, 204
 pep8, 204
 PHP Code Coverage, 205
 PMD, 193
 PMD (plugin), 193
 Polyspace, 194
 Polyspace (plugin), 195
 Polyspace MISRA, 194
 pycodestyle / pep8 (plugin), 205
 pylint, 206
 pylint (plugin), 206
 Qac_8_2, 206
 Rational Logiscope, 191
 ReqIF, 196
 SQL Code Guard, 197
 Squan Sources, 197
 Advanced COBOL parsing, 207
 Squore Import, 199
 Squore Virtual Project, 200
 StyleCop, 200
 StyleCop (plugin), 201
 Tessy, 201
 Unit Test Code Coverage from Rational Test
 RealTime, 196
 VectorCAST, 201
Data Store, 253
Data Type, 253
db, 179
Decision Criteria, 254
Decoupling, 254
defaultValue, 214
Defect, 254
Degree of Confidence, 255
Deliverable, 255
Delivery, 255
Dependability, 256
Deployment, 256
depot, 175
depth, 198
Derived Measure, 256
Design, 257
Design Pattern, 257
Developer, 258
Development, 258
Development Testing, 259
dir, 181, 183, 189, 197, 205, 206

dir_choice, 199
Direct Measure, 259
Direct Metric, 260
displayType, 214
Document, 260
Documentation, 261
DOD-STD-2167A, 382
Dynamic Analysis, 262

E

Earned Value, 262
Effectiveness, 262
Efficiency, 262
Efficiency Compliance, 263
Effort, 263
Encapsulation, 263
End User, 264
Entity, 264
Entry Point, 265
env, 215
Environment, 265
error, 215
Error, 266
Error Tolerance, 266
Evaluation, 266
Evaluation Activity, 267
Evaluation Group, 267
Evaluation Method, 268
Evaluation Module, 268
Evaluation Technology, 269
Evaluation Tool, 269
excel, 192
excludedDirectoryPattern, 185
excludedExtensions, 193, 196
exec, 215, 215
exec-phase, 212, 214, 215
executable, 215
Execute, 269
Execution Efficiency, 270
Execution Time, 270
Exit, 270
Expandability, 270
ext, 189, 196
Extendability, 271
External Attribute, 271
External Measure, 271
External Quality, 272
externals, 180
External Software Quality, 272

F

Facility, 273
failOnError, 215

failOnStdErr, 215
Failure, 273
Failure Rate, 274
fatal, 215
Fault, 275
Fault Tolerance, 275
Feasibility, 276
Feature, 276
Feature Freeze, 276
files_choice, 198
Finite State Machine, 277
Flexibility, 277
FORTRAN, 72
Frozen Branch, 277
Function, 277
Functional Analysis, 278
Functionality, 279
Functionality Compliance, 280
Functional Requirement, 278
Functional Size, 279
Functional Testing, 279
Functional Unit, 279

G

genAs, 199
genCG, 198
Generality, 280
Generic Practice, 280
genTs, 199
Glossary, 281
Goal, 281
Granularity, 281

H

hide, 214
Historical Information, 281
hostname, 177
html, 182
html_report, 202, 205
Hybrid Coupling, 281

I

id="add-data", 214
IEC 61508, 383
IEC 61508-3, 383
IEC 61508-7, 383
IEEE 1012, 384
IEEE 1058, 384
IEEE 1061, 384
IEEE 1074, 384
IEEE 1220, 385
IEEE 1233, 385
IEEE 1320, 385

IEEE 1362, 385
IEEE 1490, 386
IEEE 610.12, 386
IEEE 829, 386
IEEE 830, 386
IEEE 982, 387
image, 214
Impact Analysis, 282
Implementation, 282
Implied Needs, 283
Incremental Development, 283
Indicator, 283
Indicator Value, 284
Indirect Measure, 284
Indirect Metric, 285
Information, 285
Information Analysis, 285
Information Management, 285
Information Need, 286
Information Product, 286
inputDir, 200
Inspection, 286
Installability, 287
Installation Manual, 287
Integration, 287
Integration Test, 288
Integrity, 288
Interface Testing, 288
Intermediate Software Product, 288
Internal Attribute, 289
Internal Measure, 289
Internal Quality, 290
Internal Software Quality, 290
Interoperability, 291
Interoperability Testing, 291
Interval Scale, 291
ISO/IEC/IEEE 15289, 420
ISO/IEC/IEEE 24765, 421
ISO/IEC 12119, 391
ISO/IEC 12207, 391
ISO/IEC 14143, 391
ISO/IEC 14143-1, 392
ISO/IEC 14143-3, 392
ISO/IEC 14598, 393
ISO/IEC 14598-1, 393
ISO/IEC 14598-2, 394
ISO/IEC 14598-3, 394
ISO/IEC 14598-4, 395
ISO/IEC 14598-5, 395
ISO/IEC 14598-6, 396
ISO/IEC 14756, 396
ISO/IEC 14764, 396
ISO/IEC 15026, 397
ISO/IEC 15026-1, 397
ISO/IEC 15026-2, 397
ISO/IEC 15288, 398
ISO/IEC 15289, 398
ISO/IEC 15414, 399
ISO/IEC 15474, 399
ISO/IEC 15474-1, 399
ISO/IEC 15474-2, 400
ISO/IEC 15504, 400
ISO/IEC 15504-1, 401
ISO/IEC 15504-2, 401
ISO/IEC 15504-3, 402
ISO/IEC 15504-4, 402
ISO/IEC 15504-5, 403
ISO/IEC 15504-6, 404
ISO/IEC 15504-7, 404
ISO/IEC 15846, 405
ISO/IEC 15910, 405
ISO/IEC 15939, 406
ISO/IEC 19759, 406
ISO/IEC 19770, 406
ISO/IEC 19770-1, 407
ISO/IEC 19770-2, 407
ISO/IEC 20000, 408
ISO/IEC 2382, 408
ISO/IEC 2382-1, 409
ISO/IEC 25000, 410
ISO/IEC 25001, 410
ISO/IEC 25010, 411
ISO/IEC 25012, 411
ISO/IEC 25020, 411
ISO/IEC 25021, 412
ISO/IEC 25030, 412
ISO/IEC 25040, 413
ISO/IEC 25045, 413
ISO/IEC 25051, 413
ISO/IEC 25060, 414
ISO/IEC 25062, 414
ISO/IEC 26514, 415
ISO/IEC 29881, 415
ISO/IEC 90003, 415
ISO/IEC 9126, 415
ISO/IEC 9126-1, 416
ISO/IEC 9126-2, 417
ISO/IEC 9126-3, 417
ISO/IEC 9126-4, 418
ISO/IEC 9294, 418
ISO/IEC 99, 419
ISO/IEC SQuaRE, 419
ISO 5806, 387
ISO 8402, 387
ISO 9001, 387
ISO 9127, 387

ISO 9241, 388
ISO 9241-10, 389
ISO 9241-11, 390
Item, 292
Iteration, 292

J

JAVA, 80
JAVASCRIPT, 89

K

key, 214, 214, 215, 215
Key Practices, 292
Key Process Area, 293
Knowledge Base, 293

L

label, 175
Languages
 ABAP, 3
 ADA, 13
 C, 23
 COBOL, 35, 207
 CPP, 48
 CSHARP, 60
 FORTRAN, 72
 JAVA, 80
 JAVASCRIPT, 89
 MINDC, 97
 OBJECTIVEC, 109
 PHP, 120
 PYTHON, 130
 SQL, 138
 TSQL, 146
 VBNET, 152
 XAML, 164
languages, 198
Learnability, 293
Lessons Learned, 294
Level of Performance, 294
Licences, 475
Life Cycle, 294
Life Cycle Model, 295
log, 190, 192
logDir, 193, 196, 196
login, 204

M

Maintainability, 295
Maintainability Compliance, 296
Maintainer, 296
Maintenance, 296
Maintenance Manual, 297

Maturity, 298
Measurable Concept, 298
Measurand, 298
Measure, 298
Measurement, 299
Measurement Analyst, 300
Measurement Experience Base, 300
Measurement Function, 300
Measurement Method, 301
Measurement Procedure, 301
Measurement Process, 302
Measurement Process Owner, 302
Measurement Sponsor, 302
Measurement User, 302
Metric, 303

Metrics

 % of parsed tokens, 76, 134, 159
 AND operators, 17
 Andthen Operators, 3, 13, 23, 49, 60, 72, 80, 89,
 97, 109, 121, 138, 152, 164
 Arithmetic Operators, 35
 Assignment Operators, 23, 49, 60, 80, 97, 110
 Blank Lines, 3, 14, 23, 35, 49, 61, 73, 81, 90, 98,
 110, 121, 130, 138, 146, 153, 165
 Brace Lines, 3, 14, 24, 49, 61, 73, 81, 90, 98, 110,
 121, 130, 138, 146, 153
 Break in Loop, 24, 50, 61, 73, 81, 90, 98, 110, 131,
 138, 146, 153, 165
 Break in Switch, 24, 50, 61, 81, 90, 98, 110, 154,
 165
 Called Depth, 25, 99
 Called External Functions, 24, 98
 Called Functions, 24, 98
 Call Graph Depth, 4, 14, 25, 36, 50, 62, 73, 82, 90,
 99, 111, 122, 131, 139, 147, 154, 165
 Calling Depth, 25, 99
 Calling Functions, 24, 98
 Calls From, 24, 98
 CALL Statements, 35
 Calls To, 24, 98
 Call to exit, 5, 37, 123, 132, 156
 Case Blocks, 3, 14, 24, 50, 61, 73, 81, 90, 98, 111,
 139, 154, 165
 Case Labels, 3, 14, 24, 50, 61, 73, 81, 90, 98, 111,
 122, 139, 154, 165
 Catch Statements, 3, 50, 61, 81, 90, 111, 122, 131,
 139, 146, 154, 165
 Cloned Code, 6, 16, 27, 38, 52, 64, 75, 83, 92, 101,
 113, 124, 133, 141, 148, 157, 167
 Cloned Control Flow Tokens, 6, 16, 27, 38, 52, 64,
 75, 83, 92, 101, 113, 124, 133, 141, 148, 157, 167
 Commented Statements, 4, 14, 25, 36, 51, 62, 73,
 82, 91, 99, 111, 122, 131, 139, 147, 154, 166

Comment Lines, 4, 14, 25, 36, 50, 62, 73, 82, 91, 99, 111, 122, 131, 139, 147, 154, 165
Comment lines with code, 36
Comment lines without alphabetic characters, 36
Comparison Operators, 25, 50, 62, 82, 99, 111
Compiler FLAG Nested Level, 29, 55, 67, 103, 116, 142
Conditions, 36
Constant Data, 48, 60, 80, 109, 120, 152
Constant Methods, 124
Constant Properties, 66
Constants, 17
Continue Statements, 4, 25, 50, 62, 73, 82, 91, 99, 111, 122, 131, 139, 147, 154, 165
Control Flow Token, 4, 14, 25, 35, 50, 62, 73, 81, 90, 99, 111, 122, 131, 139, 147, 154, 165
Cyclomatic Complexity, 3, 14, 24, 35, 50, 61, 73, 81, 90, 99, 111, 122, 131, 139, 146, 154, 165
Data Declarations, 39
Data Used, 39
Debug lines, 36
Declared functions, 17
Declare Members, 157
Declare operators, 15
Default Statement, 4, 15, 26, 51, 62, 74, 82, 91, 100, 112, 122, 140, 155, 166
Delegate Members, 157
Delete Statements, 147
Depth of Descendant Tree, 51, 62, 82, 112, 122, 131, 155
Depth of Inheritance Tree, 51, 62, 82, 112, 123, 132, 155
Derived types, 18
DISPLAY statements, 36
Distinct Operands, 5, 15, 26, 36, 51, 62, 74, 82, 91, 100, 112, 123, 132, 140, 147, 155, 166
Distinct Operands in Data Div., 37
Distinct Operands in Procedure Div., 37
Distinct Operators, 5, 15, 26, 37, 51, 63, 74, 82, 91, 100, 112, 123, 132, 140, 147, 155, 166
Distinct Operators in Data Div., 37
Distinct Operators in Procedure Div., 37
Do While Statements, 26, 51, 63, 74, 83, 91, 100, 112, 123, 155, 166
Else Statements, 5, 15, 26, 37, 51, 63, 74, 83, 91, 100, 112, 123, 132, 140, 147, 155, 166
End Statements, 157
Entry Statements, 15
EVALUATE Statements, 37
Events, 155
Exception handlers, 15
Exceptions, 17
Exception When blocks, 15
Executable Statements, 7, 20, 30, 39, 55, 68, 76, 86, 93, 104, 116, 126, 134, 143, 149, 160, 168
Fiend Attributes, 152
File Declarations, 37
Files Used, 37
File Type Count, 5
Foreach Statements, 63, 123
For Statements, 5, 15, 26, 51, 63, 74, 83, 91, 100, 112, 123, 132, 140, 156, 166
Friend Events, 155
Friend Members, 157
Friend Properties, 159
Generic object, 16
Goto Statements, 16, 26, 37, 52, 63, 75, 101, 113, 124, 140, 148, 156, 167
Header Blocks Of Comment, 3, 14, 23, 49, 61, 72, 81, 89, 97, 110, 121, 130, 138, 146, 153
Header Lines Of Code, 6, 16, 27, 52, 64, 75, 83, 92, 101, 113, 124, 133, 141, 148, 157
Header Lines Of Comment, 6, 16, 27, 52, 63, 75, 83, 92, 101, 113, 124, 133, 140, 148, 156
HTML Lines of Code, 124
IDMS calls for modification, 38
IDMS calls for reading/searching, 38
IDMS instructions called, 38
IDMS records called, 38
IDMS subschema definition, 38
If Statements, 6, 16, 27, 38, 52, 64, 75, 84, 92, 101, 113, 124, 133, 141, 148, 157, 167
Insert Statements, 148
Internal Data, 60
Internal Methods, 64
Internal Properties, 66
IO Functions, 30, 104
Is IDMS active, 38
Label Statements, 16, 148
Line Count, 6, 17, 27, 38, 52, 64, 75, 84, 92, 101, 113, 124, 133, 141, 149, 157, 167
Lines Added, 8, 20, 31, 40, 56, 69, 77, 86, 94, 105, 117, 127, 135, 143, 150, 161, 169
Lines Modified, 8, 20, 31, 40, 56, 69, 77, 86, 94, 105, 117, 127, 135, 144, 150, 161, 169
Lines Removed, 8, 20, 31, 40, 56, 69, 77, 86, 94, 105, 117, 127, 135, 144, 150, 161, 169
Loop Statements, 6, 17, 27, 52, 64, 75, 84, 92, 101, 113, 124, 133, 141, 157, 167
Maximum Nested Structures, 6, 19, 28, 38, 53, 65, 75, 85, 93, 102, 114, 125, 134, 141, 149, 159, 167
Max Nested Functions, 91
Memory Allocation, 27, 101
Memory Freeing, 27, 101
Methods without Accessibility, 53, 64, 84, 114, 125, 158

Minimum Number of Cycles, 4, 15, 25, 36, 51, 62, 74, 82, 91, 100, 111, 122, 131, 139, 147, 154, 166
Minimum Number of Indirect Cycles, 25, 99
Mixed Lines, 6, 17, 27, 53, 64, 75, 84, 93, 102, 113, 125, 133, 141, 149, 158
Multiple Inheritance Indicator, 53, 64, 84, 113, 125, 133, 157
Must Members, 158
Must Properties, 159
Non-Cyclic Paths, 7, 19, 28, 54, 65, 76, 85, 93, 102, 114, 126, 134, 141, 149, 159, 167
Number of #DEFINE, 28, 54, 67, 102, 115, 142
Number of #ELIF, 28, 54, 67, 102, 115, 142
Number of #ELSE, 28, 54, 67, 102, 115, 142
Number of #ENDIF, 28, 54, 67, 102, 115, 142
Number of #ENDREGION, 67
Number of #ERROR, 28, 54, 67, 102, 115, 142
Number of #IF, 28, 54, 67, 103, 115, 142
Number of #IFDEF, 29, 54, 67, 103, 115, 142
Number of #IFNDEF, 29, 54, 67, 103, 115, 142
Number of #PRAGMA, 29, 55, 67, 103, 116, 142
Number of #REGION, 68
Number of #UNDEF, 29, 55, 68, 103, 116, 143
Number of #WARNING, 29, 55, 68, 103, 116, 143
Number of Ancestors, 53, 65, 84, 114, 125, 133, 158
Number of arithmetic if, 72
Number of Attributes, 48, 60, 80, 109, 121, 152
Number of attributes, 164
Number of Check instruction, 4
Number Of Children, 53, 65, 85, 114, 126, 134, 159
Number of comment blocks, 3, 14, 23, 49, 61, 72, 81, 89, 97, 110, 121, 130, 138, 146, 153, 164
Number of data without accessibility, 49, 60, 80, 109, 121
Number of declarative statements, 74
Number of Descendants, 53, 65, 85, 114, 125, 133, 158
Number of DocString lines, 132
Number of Include, 29, 55, 103, 115, 142
Number of Methods, 159
Number of paragraphs, 39
Number of Parameters, 28, 54, 65, 85, 93, 102, 114, 141, 159, 167
Number of Sections, 39
Number of Structures, 6, 16, 26, 52, 63, 75, 83, 92, 100, 113, 124, 132, 140, 148, 156, 167
Number of text blocks, 168
Number of XML elements, 166
Operand Occurrences, 7, 20, 30, 39, 56, 69, 76, 86, 94, 105, 117, 127, 134, 143, 150, 161, 168
Operand Occurrences in Data Div., 40
Operand Occurrences in Procedure Div., 40
Operator Occurrences, 7, 20, 31, 40, 56, 69, 76, 86, 94, 105, 117, 127, 135, 143, 150, 161, 168
Operator Occurrences in Data Div., 40
Operator Occurrences in Procedure Div., 40
Or else operators, 7, 19, 28, 54, 65, 76, 85, 93, 102, 115, 126, 141, 159, 168
OR operators, 18
Partial Members, 158
PERFORM Statements, 39
PHP/HTML Mixed Lines, 125
PHP Lines of Code, 126
Private Constant, 4
Private constant, 17
Private Data, 5
Private data, 49, 60, 80, 110, 121, 153
Private Events, 156
Private exceptions, 18
Private functions/Procedures, 17
Private Methods, 53, 65, 84, 114, 125, 158
Private Properties, 66, 160
Private types, 19
Private variables, 19
Properties, 66, 115, 159
Properties with Get, 66
Properties without Accessibility, 66
Properties with Set, 66
Protected Constant, 4
Protected Data, 5, 49, 60, 80, 110, 121, 153
Protected Events, 155
Protected Internal Data, 60
Protected Internal Methods, 65
Protected Internal Properties, 66
Protected Methods, 53, 65, 84, 114, 125, 158
Protected objects, 18
Protected Properties, 66, 160
Public Constant, 4
Public constants, 17
Public Data, 5, 49, 60, 80, 109, 121, 152
Public Events, 155
Public exceptions, 18
Public functions, 17
Public Methods, 53, 64, 84, 114, 125, 158
Public Properties, 66, 159
Public types, 19
Public variables, 19
Raise statements, 19
Real comment lines with alphabetic characters, 36
Recursive Calls, 25, 99
Renamed objects, 18
Repeated Code Blocks, 7, 20, 29, 39, 55, 68, 76, 85, 93, 103, 116, 126, 134, 143, 149, 160, 168
Return Statements, 7, 19, 29, 55, 68, 76, 85, 93, 103, 116, 126, 134, 143, 149, 160, 168

- Select Statements, 149
 - Separate functions/procedures, 18
 - Separate packages, 18
 - Separate tasks, 18
 - Shadowed Attributes, 153
 - Shadowed Events, 156
 - Shadowed Members, 158
 - Shadowed Properties, 160
 - Shared Attributes, 153
 - Shared Events, 156
 - Shared Members, 158
 - Shared Properties, 160
 - Signal Functions, 29, 104
 - Skipped Lines of Comment code, 7, 20, 30, 55, 68, 76, 85, 93, 104, 116, 126, 134, 143, 149, 160
 - Source Lines Of Code, 7, 20, 30, 39, 55, 68, 76, 85, 93, 104, 116, 126, 134, 143, 149, 160, 168
 - Special Operators, 30, 55, 68, 85, 104, 116
 - Static Data, 49, 61, 80, 110, 121
 - Static Methods, 53, 65, 84, 114, 125
 - Static Properties, 66
 - STOP Statements, 39
 - Stop Statements, 153
 - String Conversions, 30, 104
 - Structures Added, 5, 15, 26, 51, 63, 74, 83, 92, 100, 112, 123, 132, 140, 147, 156, 166
 - Structures Modified, 5, 16, 26, 52, 63, 74, 83, 92, 100, 112, 123, 132, 140, 148, 156, 166
 - Structures Removed, 6, 16, 26, 52, 63, 74, 83, 92, 100, 112, 123, 132, 140, 148, 156, 167
 - Subtypes, 18
 - Switch Statements, 7, 20, 30, 56, 68, 76, 86, 93, 104, 116, 126, 143, 160, 168
 - System Functions, 30, 104
 - Ternary operators, 30, 56, 68, 86, 94, 104, 117, 126, 160, 168
 - Throw Statements, 7, 56, 69, 86, 94, 117, 127, 134, 149, 161, 168
 - Time Handling, 30, 104
 - TIMES Clauses, 39
 - Try Statements, 8, 56, 69, 86, 94, 117, 127, 135, 150, 161, 169
 - Types, 18
 - UNTIL Clauses, 40
 - Update Statements, 150
 - Use of longjump, 27, 101
 - Use of offsetof, 28, 102
 - Use of setjump, 29, 103
 - Variables, 19
 - VARYING Clauses, 40
 - Weighted Method per Class, 56, 117
 - WHEN Clauses, 40
 - While Statements, 8, 20, 31, 56, 69, 87, 94, 105, 117, 127, 135, 144, 150, 161, 169
 - With statements, 19
 - Milestone, 303
 - MINDC, 97
 - Mock Object, 304
 - Model, 304
 - Modifiability, 304
 - Modifiable, 305
 - Modularity, 305
 - Module, 305
 - Moke Object, 306
 - Multidimensional Analysis, 306
 - multipleChoice, 214
- ## N
- name, 179
 - needSources, 213
 - Network, 306
 - Nonfunctional Requirement, 307
 - Nontechnical Requirement, 307
- ## O
- Object, 307
 - OBJECTIVEC, 109
 - Object Model, 308
 - Object Oriented Design, 308
 - objType, 197
 - Observation, 308
 - Observation Period, 309
 - Operability, 309
 - Operand, 309
 - Operational Testing, 310
 - Operator, 310
 - Operator Manual, 311
 - option, 214
 - Optional Attribute, 311
 - Optional Requirement, 311
 - Organisational Unit, 312
 - output, 200
- ## P
- p, 202
 - p4port, 175
 - password, 175, 176, 177, 178, 179, 180, 204
 - path, 178
 - Path, 312
 - Path Analysis, 312
 - Pathological Coupling, 313
 - Path Testing, 313
 - pattern, 198
 - pattern_dir, 199
 - pattern_files, 198

Peer Review, 313
Performance, 314
Performance Indicator, 314
Performance Testing, 315
PHP, 120
Pilot Project, 315
port, 177
Portability, 315
Portability Compliance, 315
Practice, 316
Precision, 316
Predictive Metric, 317
prefix, 192
Procedure, 317
Process, 317
Process Assessment, 318
Process Assessment Model, 318
Process Capability, 319
Process Capability Determination, 319
Process Capability Level, 319
Process Context, 319
Process Improvement, 320
Process Improvement Objective, 320
Process Improvement Program, 320
Process Improvement Project, 321
Process Metric, 321
Process Outcome, 321
Process Performance, 322
Process Purpose, 322
Product, 323
Productivity, 324
Product Line, 324
Product Metric, 324
Programmer Manual, 324
project, 174, 177
Project, 325
Project Management, 325
Project Phase, 326
projectSpec, 179
projectStatusOnFailure, 214
projectStatusOnWarning, 214
properties, 204
Prototype, 326
PYTHON, 130

Q

Qualification, 327
Qualification Testing, 327
qualified, 198
Quality, 327
Quality Assurance, 328
Quality Control, 329
Quality Evaluation, 329

Quality Factor, 330
Quality in Use, 331
Quality Management, 330
Quality Measure Element, 330
Quality Metric, 331
Quality Model, 331
query, 204

R

Rating, 332
Rating Level, 333
Readability, 333
rebuild_all, 198
Recoverability, 333
Recovery, 334
Redistributed Software, 475
Reengineering, 334
Regression Testing, 334
Release, 335
Reliability, 335
Reliability Compliance, 336
Repeatability of Results of Measurements, 336
Replaceability, 337
repository, 174
Repository Connectors
 ClearCase, 174
 CVS, 173
 Folder Path, 173
 Git, 176
 Multiple Source Nodes, 180
 Perforce, 175
 PTC Integrity, 176
 SVN, 179
 Synergy, 178
 TFS, 177
 Zip Upload, 173
Reproducibility of Results of Measurements, 337
Request For Change, 338
Request For Information, 338
Request For Proposal, 338
required, 214
Requirement, 338
Requirements Analysis, 339
Requirements Derivation, 340
Requirements Document, 340
Requirements Engineering, 340
Requirements Partitioning, 341
Requirements Review, 341
Requirements Specification, 341
Requirements Traceability, 342
Requirements Traceability Matrix, 342
Resource, 343
Resource Utilisation, 343

- Result, 343
- resultDir, 190, 195, 195, 201
- Retirement, 344
- rev, 180
- Reverse Engineering, 344
- revision, 177
- Risk, 345
- Risk Acceptance, 345
- Risk Analysis, 346
- Robustness, 346
- Role, 346
- Routine, 347
- RTCA/EUROCAE, 421
- Ruleset
 - 'abort, exit, getenv or system' shall not be used, 35, 109
 - 'atof, atoi or atol' shall not be used, 35, 109
 - 'cycle' shall not be used, 79
 - 'star' parameter shall not be used., 137
 - 'stop' shall not be used, 79
 - Abort shall not be used, 21
 - ALTER shall not be used, 45
 - Assignment in Boolean, 32, 57, 70, 87, 95, 106, 118, 128, 136, 170
 - Assignment without Comparison, 32, 57, 70, 87, 95, 106, 118, 128, 136, 170
 - Avoid accessing data by using the position and length, 47
 - Avoid calling a function module without handling exceptions, 11
 - Avoid Duplicated Blocks in Function, 13, 22, 34, 47, 59, 71, 79, 89, 96, 108, 120, 129, 137, 145, 152, 163, 171
 - Avoid GOTO jumps out of PERFORM range, 47
 - Avoid mixing paragraphs and sections, 47
 - Avoid obsolete DATA BEGIN OF OCCURS statement, 9
 - Avoid OPEN/CLOSE inside loops, 47
 - Avoid SELECT SQL statement with a WHERE clause containing the NOT EQUAL operator, 9
 - Avoid SELECT SQL statement without a WHERE clause, 9
 - Avoid UPDATE or DELETE SQL Statement without a WHERE clause, 10
 - Avoid using APPEND in SQL SELECT statements, 8
 - Avoid using APPEND statements in loops, 8
 - Avoid using BREAK-POINT, 8
 - Avoid using CHECK in SQL SELECT statements, 8
 - Avoid using COMMIT WORK statements in loops, 8
 - Avoid using GROUP BY in queries, 10
 - Avoid using inline PERFORM with too many lines of code, 43
 - Avoid using INSERT in SQL SELECT statements, 9
 - Avoid using INSERT statements in loops, 9
 - Avoid using LIKE in SQL queries, 10
 - Avoid using READ statement without AT END clause, 48
 - Avoid using SELECT *, 9
 - Avoid using SELECT DISTINCT Statement, 9
 - Avoid using SQL Aggregate Functions, 9
 - Avoid using SQL INTO statements in loops, 9
 - Avoid using SUBMIT statements in loops, 9
 - Avoid using the JOIN SQL clause, 10
 - Avoid using the SQL "BYPASSING BUFFER" clause, 9
 - Avoid using the WAIT statement, 10
 - Avoid using UPDATE, MODIFY, DELETE statements in loops, 10
 - Backward Goto shall not be used, 21, 31, 57, 69, 77, 105, 118, 127, 144, 150, 161, 169
 - Bad indentation of scope terminator, 41
 - Bad paragraph position used in PERFORM, 48
 - Bad statement indentation, 41
 - BLOCK Clause, 41
 - Cloned Algorithmic, 12, 22, 33, 46, 58, 71, 78, 88, 96, 107, 119, 129, 137, 145, 151, 163, 171
 - Cloned Classes, 12, 22, 33, 45, 58, 70, 78, 88, 96, 107, 119, 129, 136, 145, 151, 163, 171
 - Cloned Files, 12, 22, 33, 45, 58, 71, 78, 88, 96, 107, 119, 129, 136, 145, 151, 163, 171
 - Cloned Functions, 12, 22, 33, 45, 58, 71, 78, 88, 96, 107, 119, 129, 136, 145, 151, 163, 171
 - Close file once, 42
 - Close open file, 42
 - Column 7 for * and D Only, 41
 - Comment Before Paragraph, 169
 - Comment Division, 41
 - Commented-out Source Code is not allowed, 10, 21, 32, 57, 70, 87, 95, 106, 118, 128, 144, 150, 170
 - Comment FD, 41
 - Comment First Level, 41
 - Comment Variable 01 and 77, 41
 - Commit Used, 145
 - COMPUTE instead of ADD, 46
 - COMPUTE instead of DIVIDE, 46
 - COMPUTE instead of MULTIPLY, 46
 - COMPUTE instead of SUBTRACT, 46
 - Continue shall not be used, 12, 33, 58, 71, 78, 88, 96, 107, 119, 129, 137, 151, 163, 171
 - Delay shall not be used, 22
 - Do not use "Native SQL" instructions, 13
 - Dynamic Memory Allocation shall not be used, 32, 106
 - Each loop shall be named, 21
 - Empty line after EXIT, 41
 - Empty line after SECTION, 42
 - Empty lines around DIVISION, 41

- Exec shall not be used., 137
- Exit Label shall be named, 21
- EXIT PROGRAM shall not be used, 137
- Factorizable Classes, 12, 21, 32, 45, 58, 70, 78, 87, 95, 106, 118, 128, 136, 144, 151, 162, 170
- Factorizable Files, 12, 21, 32, 45, 58, 70, 78, 88, 95, 106, 118, 128, 136, 144, 151, 162, 170
- Factorizable Functions, 12, 21, 32, 45, 58, 70, 78, 88, 95, 107, 119, 128, 136, 144, 151, 162, 170
- Factorizable Packages, 12, 22, 33, 45, 58, 70, 78, 88, 96, 107, 119, 128, 136, 144, 151, 163, 170
- Fallthrough shall be avoided, 33, 58, 71, 88, 96, 107, 119, 129, 171
- FIXME shall not be committed in sources code, 13, 22, 33, 46, 59, 71, 79, 88, 96, 107, 119, 129, 137, 145, 151, 163, 171
- Forbid calls to dialog transactions, 11
- Forbid calls to GET RUN TIME., 11
- Forbid call to a system function, 11
- Forbid use of GENERATE REPORT / SUBROUTINE POOL / DYNPRO, 11
- Forbid use of INSERT/DELETE REPORT/TEXTPOOL, 11
- Forbid use of SYSTEM-CALL, 11
- Forbid uses of OFFSET in ASSIGN, 11
- GOTO shall not be used, 22, 33, 59, 71, 79, 107, 119, 129, 145, 151, 163, 171
- Homonymous variable shall not be used, 48
- IDMS FIND CURRENT, 42
- IDMS One modify by PERFORM, 42
- IDMS One same call, 43
- IDMS Ready Protected Update, 43
- IDMS Return Code, 43
- Incorrect Function Name, 77
- Incorrect Module Name, 77
- Incorrect Program Name, 77
- Incorrect Subroutine Name, 78
- IO Functions shall not be used, 35, 109
- Label out a switch, 33, 59, 71, 79, 96, 108, 120, 129, 137, 171
- Macro longjmp or setjmp shall not be used, 32, 106
- Macro offsetof shall not be used, 34, 108
- Method should have "self" as first argument, 136
- Method without parameter, 136
- Missing Break, 31, 57, 87, 94, 105, 117, 127, 169
- Missing Case Else clause, 161
- Missing case in switch, 13, 23, 34, 59, 72, 79, 89, 97, 108, 120, 130, 146, 172
- Missing compound if, 31, 57, 70, 87, 95, 105, 118, 128, 135, 169
- Missing compound statement, 31, 57, 69, 87, 95, 105, 118, 127, 135, 169
- Missing Default, 10, 32, 57, 70, 77, 87, 95, 106, 118, 128, 144, 170
- Missing END-ADD, 43
- Missing END-CALL, 43
- Missing END-COMPUTE, 43
- Missing END-DELETE, 43
- Missing END-DIVIDE, 43
- Missing END-EVALUATE, 42
- Missing END-IF, 43
- Missing END-MULTIPLY, 44
- Missing END-READ, 44
- Missing END-RETURN, 44
- Missing END-REWRITE, 44
- Missing END-SEARCH, 44
- Missing END-START, 44
- Missing END-STRING, 44
- Missing END-SUBTRACT, 44
- Missing END-UNSTRING, 44
- Missing END-WRITE, 44
- Missing FILLER, 45
- Missing final else, 10, 21, 32, 57, 70, 77, 87, 95, 106, 118, 128, 135, 144, 150, 161, 170
- Multiple break in loop are not allowed, 34, 59, 72, 89, 97, 108, 120, 130, 138, 152, 172
- Multiple exit, 80
- Multiple Exit (Function, Sub or Property) statement, 164
- Multiple Exit Do statement, 164
- Multiple Exit For statement, 164
- Multiple Exit in loop, 23
- Multiple exits are not allowed, 13, 23, 34, 59, 72, 79, 89, 97, 108, 120, 130, 138, 146, 152, 164, 172
- Multiple Exit While statement, 164
- Nested Program, 45
- Nesting Level of Preprocessing directives is too high, 32, 57, 106
- No case in Select, 164
- No Conditional GOTO, 46
- No DEBUG MODE, 46
- No INITIALIZE, 46
- No more than 3 nested IF, 45
- No MOVE CORRESPONDING, 46
- No procedural COPY, 46
- No RENAMES, 47
- No Resources, 170
- No Variables S9(9), 47
- Open file once, 42
- Paragraphs having exact same name, 42
- Parameters shall be ordered: 'IN', 'OUT', 'IN OUT', 23
- Perform with no THRU, 47
- Prevent use of EDITOR-CALLS, 13
- Print shall not be used., 137

- READ-WRITE Instruction, 48
Recursion are not allowed, 34, 108
Relaxed violation, 13, 23, 34, 48, 59, 72, 79, 89, 97, 108, 120, 130, 138, 146, 152, 164, 172
Resources Filename, 172
Resources Folder, 170
Risky Empty Statement, 34, 59, 72, 89, 97, 108, 120, 130, 138, 172
Rollback Used, 145
Signal or Raise shall not be used, 34, 108
Single GOBACK, 42
Standard Label, 43
Statement shall be in uppercase, 48
The class name should conform to the defined standard, 10
The form name should conform to the defined standard, 11
The function name should conform to the defined standard, 11
The macro name should conform to the defined standard, 12
The method name should conform to the defined standard, 12
The program or report name should conform to the defined standard, 13
There shall be a `__init__` method in the class., 135
There shall be a no code before first case, 33, 58, 71, 88, 96, 107, 119, 129, 171
There shall be no 'when others' in exception handler, 22
There shall be only one Statement per line, 137
Time Handling Functions shall not be used, 35, 109
TODO shall not be committed in sources code, 13, 22, 34, 47, 59, 72, 79, 89, 97, 108, 120, 130, 137, 145, 152, 163, 172
Use 'exit when' instead of if... exit syntax, 21
Use COMP for OCCURS, 47
Use FILE STATUS, 42
Use of continue is deprecated (Fortran), 77
Use of Exit Do statement, 162
Use of Exit For statement, 162
Use of Exit Function statement, 162
Use of Exit Property statement, 162
Use of Exit Select statement, 162
Use of Exit Sub statement, 162
Use of Exit Try statement, 162
Use of Exit While statement, 162
Use of SAVE and DATA, 79
Use SYNCHRONIZED, 48
Use WHEN OTHER, 48
Variable declaration format, 42
Run, 347
- S**
s, 202
Safety, 348
Satisfaction, 348
Scale, 348
scnode, 198
scnode_name, 198
scope, 177
Security, 349
server, 179, 204
server_display_view, 174
Service, 350
Service Level Agreement, 350
SIGIST, 421
Simplicity, 350
sln, 201
Software, 351
Software Asset Management, 351
Software Development Process, 352
Software Engineering, 352
Software Item, 352
Software Licence Agreement, 475
Software Life Cycle, 353
Software Product Evaluation, 353
Software Quality, 354
Software Quality Characteristic, 354
Software Quality Evaluation, 354
Software Quality Measure, 355
Software Repository, 355
Software Unit, 355
Source Code, 356
Specification, 356
SQL, 138
Stability, 356
Stage, 357
Stakeholder, 357
Standard, 358
Standard Process, 358
Statement, 358
Statement of Work, 359
Statement Testing, 359
Static Analysis, 359
Statistical Process Control, 360
Step, 361
Stress Testing, 361
Structural Testing, 361
Stub, 362
style, 214
sub_path, 174
subDir, 176
subFolder, 179
Suitability, 362
Supplier, 363

Support, 363
Support Manual, 364
System, 364
System Testing, 364

T

tag, 214
tags, 213
Task, 365
Team Software Process, 421
Technical Requirement, 365
Technique, 366
Test, 366
Testability, 370
Test Case, 367
Test Case Suite, 367
Test Coverage, 367
Test Documentation, 368
Test Environment, 368
Testing, 370
Testing Description, 371
Test Objective, 369
Test Plan, 369
Test Procedure, 369
Time Behaviour, 371
Tool, 372
Total Quality Management, 372
Traceability, 372
Traceable, 373
Trunk, 373
TSQL, 146
txt, 187, 189, 207
type, 214

U

Understandability, 373
unitByUnit, 195, 195
Unit of Measurement, 374
Unit Test, 374
url, 176, 180
URL, 178
Usability, 375
Usability Compliance, 375
useAccountCredentials, 175, 176, 177, 178, 179, 180
User, 375
User Documentation, 376
User Manual, 377
username, 175, 176, 177, 178, 180

V

Validation, 377
value, 214
Value, 379

VBNET, 152
Verification, 379
version, 178
Version, 380
view, 174
view_root_path, 174
vob_root_path, 174

W

warn, 215
Work Breakdown Structure, 380
Work Product, 381

X

XAML, 164
xml, 182, 183, 184, 184, 184, 186, 186, 187, 187, 189,
191, 191, 192, 193, 194, 197, 200, 202
XML Format Reference, 212, 214
XML Schema
 analysis.xsd, 451
 config-1.2.xsd, 449
 decision.xsd, 456
 description.xsd, 457
 exports.xsd, 458
 highlights.xsd, 459
 input-data-2.xsd, 446
 properties.xsd, 463
 properties-1.2.xsd, 448
 tutorials.xsd, 464
 wizards.xsd, 470
xmx, 185, 186, 188