



Squore 15-A-SP2

Getting Started Guide

Reference : SUM_Squore
Version : 15-A-SP2
Date : 07/10/2015

Getting Started Guide

Copyright © 2015 Squoring Technologies

Abstract

This edition of the Getting Started Guide applies to Square 15-A-SP2 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

Typographical Conventions	viii
Acronyms and Abbreviations	ix
1. Introduction	1
1.1. Foreword	1
1.2. About This Document	1
1.3. Contacting Squoring Technologies Product Support	1
1.4. Responsibilities	2
1.5. Getting the Latest Version of this Manual	2
2. The Tools at Your Disposal	3
2.1. Default Users and Sample Projects	3
2.2. Getting More Help	3
2.2.1. Online Help	3
2.2.2. User Guides and Support Wiki	4
2.2.3. Log Files and Debug info	4
3. Accessing Squore	6
3.1. Understanding Profiles and Roles	6
3.1.1. User Profiles	6
3.1.2. User Roles	7
3.2. How Do I log into Squore?	7
3.3. Where Do I Go From Here?	8
3.4. How Do I log out of Squore?	9
3.5. Can I Tweak the Squore Look and Feel?	9
3.5.1. Using a Different Theme	9
3.5.2. User Interface Language	9
4. Creating Projects and Versions	10
4.1. How Do I Create a Project in Squore?	10
4.2. Creating Version 2 of My Project	15
4.3. Working with Draft and Baseline Versions	17
4.3.1. Drafts vs. Baseline: The Basic Concepts	18
4.3.2. Baselining at Version Creation	18
4.3.3. Baselining After Review	18
4.4. Can I Make Changes to My Project?	19
4.5. Can I Create a Project Via the Command Line?	19
4.6. How Do I Connect Squore to My Continuous Integration System?	20
4.7. Can Squore Pull Source From My Version Control System?	20
4.8. Can I Create Projects with Sources From Multiple Locations?	20
4.9. Where Are My Analysis Results?	21
4.9.1. The Tree Pane	23
4.9.2. The Dashboards	26
5. Understanding Analysis Results	29
5.1. Has the Quality of My Project Decreased Since the Previous Analysis?	29
5.1.1. Finding Artefacts Using Filters and Search	31
5.1.2. Finding Artefacts Using Highlights	38
5.2. How Do I Find and Keep Track of Artefacts?	40
5.3. How can I Understand and Enhance My Model?	41
5.3.1. Viewer	41
5.3.2. Validator	42
5.3.3. Dashboard Editor	44
5.4. Reviewing Multiple Projects	45
6. Managing Your To-Do List With Squore	48

6.1. How do I understand and Improve My Ratings?	48
6.2. Relaxing Violations in Code	55
6.3. Relaxing Artefacts	57
6.4. Working with Forms and Checklists	61
6.5. What Does This Measure Mean Exactly?	63
6.6. How Do I Review And Manage Action Items Flagged by Squore?	64
6.7. Can I Perform Advanced Data Mining?	66
7. Track Your Favourite Indicators	70
7.1. Building a cross-project Dashboard in My Favourites	70
7.2. Managing Favourites	71
7.3. Squore Mobile	71
8. Communicating With Squore	73
8.1. Comments and Notifications	73
8.1.1. Commenting Charts	73
8.1.2. Commenting Action Items	74
8.1.3. Commenting From the Artefact Tree	75
8.1.4. Following Discussions	75
8.2. Adding and Removing Artefacts Manually	77
8.3. Reporting Project Status	79
8.4. Providing Access to Collaborators	79
8.5. E-mail Notifications	81
9. Keep it Tidy: Project Maintenance in Squore	83
9.1. Can I Delete a Version?	83
9.2. Can I Delete a Project?	83
9.3. Squore Server Administration	83
9.4. What About Server Maintenance?	84
10. Repository Connectors	85
10.1. Folder Path	85
10.1.1. Description	85
10.1.2. Usage	85
10.2. Zip Upload	85
10.2.1. Description	85
10.2.2. Usage	85
10.3. ClearCase	85
10.3.1. Description	85
10.3.2. Usage	86
10.4. TFS	86
10.4.1. Description	86
10.4.2. Usage	86
10.5. Git	87
10.5.1. Description	87
10.5.2. Usage	87
10.6. MKS	87
10.6.1. Description	87
10.6.2. Usage	87
10.7. Synergy	88
10.7.1. Description	88
10.7.2. Usage	88
10.8. SVN	89
10.8.1. Description	89
10.8.2. Usage	89
10.9. CVS	89
10.9.1. Description	89

10.9.2. Usage	89
10.10. Perforce	90
10.10.1. Description	90
10.10.2. Usage	90
10.11. Using Multiple Nodes	90
10.12. Using Data Provider Input Files From Version Control	91
11. Data Providers	92
11.1. AntiC	92
11.1.1. Description	92
11.1.2. Usage	92
11.2. BullseyeCoverage Code Coverage Analyzer	92
11.2.1. Description	92
11.2.2. Usage	92
11.3. CPD	92
11.3.1. Description	93
11.3.2. Usage	93
11.4. CPD (plugin)	93
11.4.1. Description	93
11.4.2. Usage	93
11.5. Cppcheck	93
11.5.1. Description	93
11.5.2. Usage	94
11.6. Cppcheck (plugin)	94
11.6.1. Description	94
11.6.2. Usage	94
11.7. CPPTest	94
11.7.1. Description	94
11.7.2. Usage	94
11.8. CheckStyle	95
11.8.1. Description	95
11.8.2. Usage	95
11.9. CheckStyle (plugin)	95
11.9.1. Description	95
11.9.2. Usage	95
11.10. CheckStyle for SQALE (plugin)	96
11.10.1. Description	96
11.10.2. Usage	96
11.11. Cobertura	96
11.11.1. Description	96
11.11.2. Usage	96
11.12. CodeSonar	97
11.12.1. Description	97
11.12.2. Usage	97
11.13. Coverity	97
11.13.1. Description	97
11.13.2. Usage	97
11.14. FindBugs	97
11.14.1. Description	97
11.14.2. Usage	97
11.15. FindBugs (plugin)	98
11.15.1. Description	98
11.15.2. Usage	98
11.16. FxCop	98

11.16.1. Description	98
11.16.2. Usage	99
11.17. GCov	99
11.17.1. Description	99
11.17.2. Usage	99
11.18. GNATcheck	99
11.18.1. Description	99
11.18.2. Usage	99
11.19. GNATCompiler	99
11.19.1. Description	100
11.19.2. Usage	100
11.20. JUnit	100
11.20.1. Description	100
11.20.2. Usage	100
11.21. JaCoCo	100
11.21.1. Description	100
11.21.2. Usage	100
11.22. Klocwork	101
11.22.1. Description	101
11.22.2. Usage	101
11.23. Rational Logiscope	101
11.23.1. Description	101
11.23.2. Usage	101
11.24. NCover	101
11.24.1. Description	102
11.24.2. Usage	102
11.25. Oracle PLSQL compiler Warning checker	102
11.25.1. Description	102
11.25.2. Usage	102
11.26. MISRA Rule Checking using PC-lint	102
11.26.1. Description	103
11.26.2. Usage	103
11.27. PMD	103
11.27.1. Description	103
11.27.2. Usage	103
11.28. PMD (plugin)	103
11.28.1. Description	103
11.28.2. Usage	104
11.29. Polyspace	104
11.29.1. Description	104
11.29.2. Usage	104
11.30. MISRA Rule Checking using Polyspace	104
11.30.1. Description	104
11.30.2. Usage	104
11.31. Polyspace (plugin)	105
11.31.1. Description	105
11.31.2. Usage	105
11.32. MISRA Rule Checking with QAC	105
11.32.1. Description	105
11.32.2. Usage	105
11.33. Unit Test Code Coverage from Rational Test RealTime	106
11.33.1. Description	106
11.33.2. Usage	106

11.34. ReqIF	106
11.34.1. Description	106
11.34.2. Usage	106
11.35. SQL Code Guard	107
11.35.1. Description	107
11.35.2. Usage	107
11.36. Squan Sources	107
11.36.1. Description	107
11.36.2. Usage	107
11.37. Squore Import	109
11.37.1. Description	109
11.37.2. Usage	109
11.38. Squore Virtual Project	109
11.38.1. Description	109
11.38.2. Usage	110
11.39. StyleCop	110
11.39.1. Description	110
11.39.2. Usage	110
11.40. StyleCop (plugin)	110
11.40.1. Description	110
11.40.2. Usage	110
11.41. Tessy	111
11.41.1. Description	111
11.41.2. Usage	111
11.42. OSLC	111
11.42.1. Description	111
11.42.2. Usage	111
11.43. pep8	112
11.43.1. Description	112
11.43.2. Usage	112
11.44. pylint	112
11.44.1. Description	112
11.44.2. Usage	112
Index	113

Typographical Conventions

The following conventions are used in this manual.

Typeface or Symbol	Meaning
Bold	Book titles, important items, or items that can be selected including buttons and menu choices. For example: Click the Next > button to continue
<i>Italic</i>	A name of a user defined textual element. For example: Username : <i>admin</i>
Courier New	Files and directories; file extensions, computer output. For example: Edit the <code>config.xml</code> file
Courier Bold	Commands, screen messages requiring user action. For example: Username : <i>admin</i>
>	Menu choices. For example: Select File > Open . This means select the File menu, then select the Open command from it.
<...>	Generic terms. For example: <INSTALLDIR> refers to the Squore installation directory.

Notes

Screenshots displayed in this manual may differ slightly from the ones in the actual product.

Acronyms and Abbreviations

The following acronyms and abbreviations are used in this manual.

CI	Continuous Integration
CLI	Command Line Interface
DP	Data Provider, a Squore module capable of handling input from various other systems and import information into Squore
RC	Repository Connector, a Squore module capable of extracting source code from source code management systems.

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

This document is the Getting Started Guide for Squore.

It is indented as a follow up to the Squore Installation and Administration Guide and will help you understand how to use the Squore user interface to create and update projects. It is divided into several chapters, as detailed below:

- Chapter 2, *The Tools at Your Disposal* provides details on where to find the sample Squore projects.
- Chapter 3, *Accessing Squore* will guide you through your first access to Squore as a user.
- Chapter 4, *Creating Projects and Versions* covers ways of creating new projects and versions.
- Chapter 5, *Understanding Analysis Results* describes the user interface and functionality you will use in Squore on a daily basis.
- Chapter 6, *Managing Your To-Do List With Squore* helps you integrate action items suggested by Squore into your workflow.
- Chapter 7, *Track Your Favourite Indicators* shows how you can track your favourite items and consult Squore results on mobile devices.
- Chapter 8, *Communicating With Squore* covers all reporting features of Squore.
- Chapter 9, *Keep it Tidy: Project Maintenance in Squore* helps you maintain a Squore installation.

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 15-A-SP2)** or **(new in 2015-A)** throughout this manual. A summary of the new features described in this manual is available in the entry * **What's New in Squore 2015-A?** of this manual's Getting Started Guide.

For information on how to use and configure Squore, the following manuals are also available:

- Squore Installation and Administration Guide,
- Squore Command Line Interface,
- Squore Eclipse Plugin Guide,
- Squore Configuration 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: <http://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 <http://support.squoring.com/documentation/15-A-SP2>. Manuals are constantly updated and published as soon as they are available.

2. The Tools at Your Disposal

2.1. Default Users and Sample Projects

Squore ships with a collection of sample projects that we will refer to throughout this guide. Each project consists of one or several versions of the source code of an application. The code can be found in Squore Server and Squore CLI in the folder `<INSTALLDIR>/samples`. If you do not have access to the sample projects, contact your Squore administrator to obtain a copy of the code.

Squore ships with a database that contains two sample users that you can use to familiarise yourself with all the functionality available:

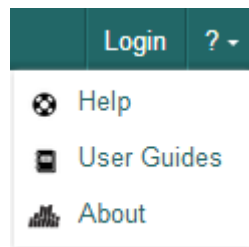
- **admin/admin** is the default user that can manage the server installation, reload the server configuration after changes and perform access management tasks for the Squore installation.
- **demo/demo** is the default Squore power user that can create, review and manage projects, as well as give team members visibility or management privileges on the projects he himself manages.

You can use these two default users, but we recommend that you change their passwords after your first connection. The privileges and permissions assigned to these default users can be modified as needed. You can familiarise yourself with Squore permissions and privileges by referring to Section 3.1, “Understanding Profiles and Roles”.

Tip

You may choose to read this manual from beginning to end, or jump straight to a specific topic. Logging in as the **demo** user, gives you access to a **Tools** menu that allows to reproduce the examples shown in this manual. Click **Tools > Create Demo** and select the **ISO9126 -- C** to get started.

2.2. Getting More Help



The Help menu

If at any moment you have doubts about how a feature works, Squore offers help in HTML and PDF formats. A Wiki and support site are also available.

2.2.1. Online Help

The Squore online help can be accessed from anywhere in Squore by clicking on the **? > Help** menu entry.

The online help is contextual and provides information in a popup window about the page that you are currently viewing in Squore.

2.2.2. User Guides and Support Wiki







The Squore user guides are available in PDF and HTML format by clicking the ? > **User Guides** menu entry in Squore. You can download a copy for offline use.

The Squoring Technologies Support Wiki provides release notes, known issues and hints and tips for current and past Squore versions. Visit <http://openwiki.squoring.com> for more information.

2.2.3. Log Files and Debug info

Every **owner** or **Project Manager** of a project can retrieve the analysis log files for their projects without the need to consult an administrator. This is done by accessing the **Manage** page for a particular project and viewing the **Versions** tab (**My Projects page** > **Manage icon** > **Versions tab**) as shown below:

Manage Project - Earth

Manage Project - Earth									
Project Properties			Versions				Team		
Id	Version	Creation Time	Owner	Status	Baseline	Log	Debug Info		
<input type="checkbox"/>	6	V1	Sep 8, 2014 11:27:16 AM	demo	Successful	Yes		Download	
<input type="checkbox"/>	7	V2	Sep 8, 2014 11:27:32 AM	demo	Successful	Yes		Download	
<input type="checkbox"/>	8	V3	Sep 8, 2014 11:28:08 AM	demo	Successful	Yes		Download	
<input type="checkbox"/>	9	V4	Sep 8, 2014 11:28:18 AM	demo	Successful	Yes		Download	
<input type="checkbox"/>	10	V5	Sep 8, 2014 11:28:27 AM	demo	Successful	Yes		Download	
<input type="checkbox"/>	11		Sep 8, 2014 11:28:36 AM	demo	Successful	No		Download	
<input type="checkbox"/>	12				Work in progress				

The Versions tab provides access to log files and Debug info

Clicking the **Log** icon opens a page showing the project's client and server logs, as well as configuration and output files will open in a new browser tab.

Clicking the **Download** link in the **Debug info** column downloads a zip file of the logs and project data that can be further analysed to understand problems during problem creation.

In order to investigate application failures (rather than project analysis errors), Squore administrators have the possibility to extract the latest log file created by the application. You can access the log if you have administrator privileges by clicking **Administration** > **Server Log** in the toolbar after logging in. The log file opens in a new browser window or tab.

Administrators can also get debug information and manage any project created on the server by clicking **Administration** > **Projects**, which provides a detailed view of all projects created on Squore Server, on a summary page shown below.

Projects
Number of Projects: 8

Number of Versions: 13

Database Size: 33 MiB

Id	Project	Owner	Creation Time	Versions	Last Version	Last Build Time	Status		
2	Earth	demo	Sep 8, 2014 11:27:16 AM	6	Current	Sep 8, 2014 11:28:36 AM	Successful		
3	Mars	demo	Sep 8, 2014 11:28:46 AM	1	V3.2.6	Sep 8, 2014 11:28:46 AM	Successful		
4	Neptune	demo	Sep 8, 2014 11:28:53 AM	1	W25	Sep 8, 2014 11:28:53 AM	Successful		
5	Venus	demo	Sep 8, 2014 11:29:00 AM	1	Beta	Sep 8, 2014 11:29:00 AM	Successful		
6	Pluto	demo	Sep 8, 2014 11:29:07 AM	1	R9	Sep 8, 2014 11:29:07 AM	Successful		
7	Uranus	demo	Sep 8, 2014 11:29:14 AM	1	B625	Sep 8, 2014 11:29:14 AM	Successful		
8	Mercury	demo	Sep 8, 2014 11:29:28 AM	1	V2010B	Sep 8, 2014 11:29:28 AM	Successful		
9	Saturn	demo	Sep 8, 2014 11:29:38 AM	1	Prel	Sep 8, 2014 11:29:38 AM	Successful		

The project administration page for administrators

A debug info package contains the following items:

- A `DataProviders` folder containing the output files generated by each Data Provider run during the analysis.
- A `[DataProviderName].log` file for each Data Provider included in the analysis.
- A `[projectId]_conf.xml` file summarising the project parameters used for the analysis.
- A `[projectId]_output.xml` file containing the output information requested with the `--filter` parameter during the analysis.
- A `build.log` file containing the information relative to actions carried out on the server during the analysis.
- A `build_client.log` file containing the information relative to actions carried out on the client during the analysis.
- A `excluded.log` file containing the list of all files not included in the analysis and the reason for their exclusion. Note that this file is only generated if some files were excluded.
- A `table.md5` file containing state information about the analysed source code, if any.
- A `storage` folder containing information about the analysed source code, if any.

Tip

If you do not want to download the entire debug package, note that the main log files can also be downloaded individually from the Projects page by clicking on the project status label.

3. Accessing Squore

This chapter walks you through your first access to Squore and covers the web interface and some ways to customise it to your liking.

3.1. Understanding Profiles and Roles

Before you start working with Squore, it is essential to understand how access management works. The various permissions and privileges that can be assigned to Squore users are grouped in profiles and roles respectively. A set of default roles and profiles is available when you first start the server. You can edit them, or create more as needed.

Use this simple trick to remember the different between a profile and a role:

- A **Profile** is a set of permissions granting access to certain Squore features to a user
- A **Role** is a set of privileges for a user within a Squore project.

A Squore user with the Administrator profile can manage users, their roles and profiles. A Squore user with the Project Manager role for a project can create a new version of this project or give access to another user to this project's analysis results.

3.1.1. User Profiles

You can use profiles to grant or deny access to the following Squore features:

- **Manage Server:** Configure the server, access server logs, manage all projects.
- **Manage Users, Groups and Roles:** Complete access to user management on the server.
- **Use Capitalisation Base:** Provides access to the Capitalisation Base feature to learn from past data in order to improve your model.
- **Create Projects:** Allows users to run analyses.
- **View Models:** Allows users to use the Viewer and the Validator.
- **Modify Models:** Allows users to use the Dashboard Editor and the Analysis Model Editor (beta).
- **Use External Tools:** View and use external tools configured by your Squore Administrator. To learn more about this feature, consult the Configuration Guide.
- **Manage Configuration:** Allows users to reload the server configuration from disk.

Three profiles are available by default, with permissions set as shown below:

	Manage Server	Manage Users, Groups and Roles	Use Capitalisation Base	Create Projects	View Models	Modify Models	Use External Tools	Manage Configuration
ADMINISTRATOR	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ADVANCED_USER	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
STANDARD_USER	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Discard

Default profiles for administrators, advanced user and standard user

Note that a profile can be assigned to a user or a group of users. It is therefore possible for a user be a member of more than one profile. In this case, the user's profile is the combination of all permissions from all the profiles they are a member of.

3.1.2. User Roles

A role is the set of privileges that a user enjoys in the context of a project. You can use roles to allow users to undertake these actions within the scope of a project:

- **View Projects:** Allows a user to see a project in their project list and to browse this project's analysis results.
- **Manage Projects:** Allows a user to manage a project: rename it, create or delete versions, access project creation log files and add other user to the project team.
- **Baseline Projects:** Allows a user to create a baseline version of a project that will not be overwritten by a subsequent analysis. For more information about baselining, see Section 4.3, "Working with Draft and Baseline Versions".
- **View Drafts of Projects:** Allows a user to view the current draft version of a project. Without this privilege, only baseline versions of a project are visible in the project portfolio. For more information about baselining, see Section 4.3, "Working with Draft and Baseline Versions".
- **Modify Action Items:** Allows updating the status of Action Items from `TODO` to `Relaxed` for example. Without this privilege, the status is displayed as a read-only field.
- **Modify Artefacts Attributes:** Allows user to modify the value of attributes displayed in the Forms tab of the Explorer. Without this privilege, attributes are read-only.
- **View Source Code:** Allows user to click to view the source code of an artefact from any tab in the Explorer.
- **Modify Artefacts:** Allows user to add, delete, relax, exclude artefacts from the artefact tree. Users without this privilege, can still view artefacts created by others.

Five roles are available by default, with privileges assigned as shown below:

Role ↕	View Projects	Manage Projects	Baseline Projects	View Drafts of Projects	Modify Action Items	Modify Artefacts Attributes	View Source Code	Modify Artefacts
PROJECT_MANAGER	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
QUALITY_ENGINEER	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
DEVELOPER	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
TESTER	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
GUEST	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Default roles available for users in Squore

Note that a user can have multiple roles in a project. This allows a user to view the dashboard in the Explorer as a user from another role would. A **View As** option in the option menu of the Explorer allows to you to switch between the various dashboards available to you. When you have multiple roles in a project, you combine privileges from all the roles that you are a member of.

3.2. How Do I log into Squore?

Your Squore installation runs on `http://localhost:8180/SquORE_Server` by default. By accessing this page in your browser, you will be redirected to the Squore login page, as shown below:

Please login

Username:

Password:

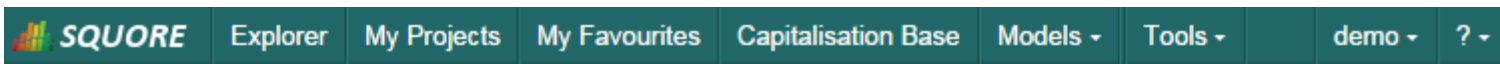
Remember me:

The Squore login page

Type in a username and a password and check the **Remember Me** box if you want your username to be filled-in automatically the next time you view the login page.

3.3. Where Do I Go From Here?

To begin using Squore, log in as the demo user with demo as username and password on the login page. Click the **Login** button and wait for the Welcome page to open.



Welcome, demo!

About Me

- Username: demo
- Number of Connections: 1
- Account Settings

Ready to Go?

- Track Performance Levels
- Manage Projects
- Learn from Past Data

Need Some Help?

- Click ? > Help to learn to interact with a page
- Read User Guides
- Request Squore Support

Pinned Artefacts

- None

Last Visited Projects

- Earth

Copyright © 2014 Squoring Technologies - All rights reserved - <http://www.squoring.com/>

The Squore Welcome page

From the Welcome page, you can automatically return to the last projects or favourite artefacts that you had opened in the Explorer before logging out. You can also get links to the help and other features available for your account.

As the demo user, you are an advanced user of Squore and have access to the following functionality from the toolbar:

- **Explorer**, where you can review and drill-down data for your projects.
- **My Projects**, where projects are created and managed.
- **My Favourites**, where you can view and manage your favourite charts across projects.
- **Capitalisation Base**, where aggregated statistical data can be found.
- **Models**, under which you can examine all characteristics of your model and edit your dashboards.
- **Tools**, which contains shortcuts to scripts that recreate demo projects. Note that only the demo user has access to this menu by default.
- **<username>**, where you can set your preferences and log out from Squore.
- **?**, where online help, user manuals and application information can be found.

Note: If you log in as an administrator of Squore using `admin` as the username and password, you will gain access to the **Administration** menu where you can configure access management and administer the server.

3.4. How Do I log out of Squore?

You can log out of Squore by clicking your user name in the menu bar and selecting the **Log Out** option. Note that if you close your browser without logging out, your session will automatically time out after two hours.

3.5. Can I Tweak the Squore Look and Feel?

3.5.1. Using a Different Theme

The Squore look and feel can be adapted to your liking, with three provided themes, accessible from the **<username>** menu option. Select one of the available colour schemes to change the color of the interface. Your changes are saved using a browser cookie.

3.5.2. User Interface Language

You can use Squore in various languages. English and French are provided by default, and your Squore administrator can add more as needed. If you want to change the language of the Squore user interface, click the **<username>** menu option and click one of the flags available. The changes are applied immediately and your preferences are saved even after you log out.

4. Creating Projects and Versions

In this chapter, you will learn about the various ways to create a project in Squore: using the UI, using a command line tool or triggering analyses in a continuous integration environment.

4.1. How Do I Create a Project in Squore?
















Creating a project in Squore is as easy as following a wizard that will prompt you for information about the source material to analyse, and the external Data Providers to add to the analysis results.

The example below assumes that the source code for the sample project used is available on a network share. The path to the source files to analyse is relative to the server.

In order to create a project for the sample application Neptune2, follow these steps:

1. Access `http://localhost:8180/SQuORE_Server` in your browser. The log-in page appears.
2. Log in as the demo user with the login/password combination `demo/demo`.
3. Click the **Login** button. You are presented with the Squore home page.
4. Click **My Projects** to switch to the projects view and click **New Project...** to create the Neptune2 project. The list of available project creation wizards appears:

Wizard Selection

- | | | |
|----------------------------------|---|---|
| <input type="radio"/> |  Check C/C++ Code ECSS Compliance | C/C++ Code ECSS Compliance |
| <input type="radio"/> |  Check C/C++ Code Compliance to HIS Metrics | HIS C Code Metrics Compliance |
| <input type="radio"/> |  Determine COBOL C | |
| <input checked="" type="radio"/> |  Determine C Code Maintainability Level | C Code ISO-9126 Maintainability Level |
| <input type="radio"/> |  Determine Java Code Maintainability Level | Java Code ISO-9126 Maintainability Level |
| <input type="radio"/> |  Determine Object-Oriented Code Maintainability Level | Object-Oriented Code ISO-9126 Maintainability Level |
| <input type="radio"/> |  Determine XAML Code Maintainability Level | XAML Code ISO-9126 Maintainability Level |
| <input type="radio"/> |  Calculate COBOL Programs SQuALE Quality Index | COBOL Programs SQuALE Quality Index |
| <input type="radio"/> |  Calculate Java Code SQuALE Quality Index | Java Code SQuALE Quality Index |
| <input type="radio"/> |  Assess C Code SQuORE Risk Index | C Code SQuORE Risk Index |
| <input type="radio"/> |  Assess Java Code SQuORE Risk Index | Java Code SQuORE Risk Index |
| <input type="radio"/> |  Assess Object-Oriented Code SQuORE Risk Index | Object-Oriented Code SQuORE Risk Index |
| <input type="radio"/> |  Assess PL/SQL Code SQuORE Risk Index | PL/SQL Code SQuORE Risk Index |
| <input type="radio"/> |  Assess Ada Code SQuORE Risk Level | Ada Code SQuORE Risk Index |
| <input type="radio"/> |  Java Technical Debt | Java Technical Debt Management |

Description: Determine C code level of Maintainability according to the ISO-9126 Quality Model using only base measures from SQuORE. No additional data providers are needed.

Previous


Next

Cancel

The SQuore wizard selection screen

- Click the **Determine C Code Maintainability Level** wizard to start creating the project.
- On the Project Identification screen, enter the information relative to your project as shown below:

Project Identification

Project Name:	<input type="text" value="Project1"/>				
Group:	<input type="text"/>				?
Version Pattern:	<input type="text" value="V#N1#"/>				?
Version Name:	<input type="text" value="V1"/>				
Version Date:	<input type="text"/>				
Color:	<input type="color" value="#800080"/>				?
Automatic Baselining:	<input checked="" type="checkbox"/>				?
Keep old versions data files:	<input type="checkbox"/>				?
E-mail the creator of a version:	<input type="checkbox"/> On draft	<input type="checkbox"/> On baseline	<input type="checkbox"/> On error		?
E-mail team members:	<input type="checkbox"/> On draft	<input type="checkbox"/> On baseline			?

Application attributes

Project Business Value:	<input type="text" value="50.0"/>	FP			
Project Cost:	<input type="text" value="0.0"/>	M/M			
Project Safety Integrity Level:	<input checked="" type="radio"/> SIL0 <input type="radio"/> SIL1 <input type="radio"/> SIL2 <input type="radio"/> SIL3 <input type="radio"/> SIL4				

The Project Identification screen

Tip

The **Version Date** field allows specifying a custom date for the analysis, so that different analyses can be placed correctly on a timeline later for certain charts in the dashboard. If you leave it empty, then the actual time at which you are running the analysis is used.

7. Click the **Next >** button. The Data Providers options screen is shown:

Specify Repository Locations

Folder
 Zip Upload
 ClearCase
 CVS
 Git
 MKS
 Perforce
 SVN
 Synergy
 TFS

Datapath:

Select Data Providers

<input type="checkbox"/> AntiC	<input type="checkbox"/> FxCop	<input type="checkbox"/> Polyspace
<input type="checkbox"/> Bullseye Code Coverage	<input type="checkbox"/> GCov	<input type="checkbox"/> Polyspace (plugin)
<input type="checkbox"/> CPD	<input type="checkbox"/> GnatCheck	<input type="checkbox"/> Rational Logiscope
<input type="checkbox"/> CPD (plugin)	<input type="checkbox"/> GnatCompiler	<input type="checkbox"/> SQLCodeGuard
<input type="checkbox"/> CPPCheck	<input type="checkbox"/> JUnit	<input checked="" type="checkbox"/> SQuAN Sources
<input type="checkbox"/> CPPCheck (plugin)	<input type="checkbox"/> Jacoco	<input type="checkbox"/> SQuORE Import
<input type="checkbox"/> CPPTest	<input type="checkbox"/> Klocwork	<input type="checkbox"/> SQuORE Virtual Project
<input type="checkbox"/> CheckStyle	<input type="checkbox"/> MISRA Rule Checking from QAC	<input type="checkbox"/> StyleCop
<input type="checkbox"/> CheckStyle (plugin)	<input type="checkbox"/> MISRA Rule Checking using PC_Lint	<input type="checkbox"/> StyleCop (plugin)
<input type="checkbox"/> CheckStyle for SQuALE (plugin)	<input type="checkbox"/> MISRA Rule Checking using Polyspace	<input type="checkbox"/> Tessy
<input type="checkbox"/> CodeSonar	<input type="checkbox"/> NCover	<input type="checkbox"/> Unit Test Code Coverage from Rational Test RealTime
<input type="checkbox"/> Coverity	<input type="checkbox"/> Oracle PLSQL compiler Warning checker	<input type="checkbox"/> oslc_cm
<input type="checkbox"/> FindBugs	<input type="checkbox"/> PMD	<input type="checkbox"/> pep8
<input type="checkbox"/> FindBugs (plugin)	<input type="checkbox"/> PMD (plugin)	<input type="checkbox"/> pylint

Note: Data Providers listed as plugins above require a one-time download of binary components before they can be executed for the first time. Consult the

The Data Providers options screen

8. This screen allows configuring the repository locations and tools that will be used in your analysis. Set the source code files option to **Folder**. In the **Datapath** text box, type the path to the Neptune2 source code: `\\server\share\samples\c\Neptune\W25`. The only Data Provider used in our analysis is SQuORE, the source code analyser, so you can leave all the other tools unchecked.

Tip

If you want to learn more about the available Data Providers and Repository Connectors, consult Chapter 11, *Data Providers* and Chapter 10, *Repository Connectors*.

In the SQuORE parameters, ensure that the programming language selected is **C**, as shown below:

Specify Repository Locations

Select Data Providers

SQuORE


SQuAN Sources

 Languages: C
 Generate control graphs

 Use qualified names

 Limit the analysis depth to file only

 Add a 'Source Code' node

 Name of the 'Source Code' node
 Compact folders having only one son

Exclude files whose contents match with PERL regexp:

 files whose name matches with Exclude Include

 Shell-like wildcards e.g.

 directories whose name matches with Exclude Include

 Shell-like wildcards e.g.
 Detect functional cloning

 Detect text cloning





























The SQuORE Data Provider parameters

- Click the **Next >** button to view a summary of your selections and **Run** to confirm the project creation.

Tip

The summary page lists all the options you specified for the project creation and also allows outputting them in various formats so that you can repeat the project creation in command line. For more information about reusing the project parameters in a different context, consult the online help or Section 4.5, "Can I Create a Project Via the Command Line?".

When the project analysis completes, Squore shows you the list of projects. Neptune2 appears in the list, together with information about the current version and its computed rating:

Name ↕	Version ↕	Rating	Analysis Model ↕	Color ↕	Owner ↕	Manage	Build	Base
	Current	E	C Code ISO-9126 Maintainability Level		demo			
	V3.2.6	D	C Code ISO-9126 Maintainability Level		demo			
	V2010B	E	C Code ISO-9126 Maintainability Level		demo			
	W25	C	C Code ISO-9126 Maintainability Level		demo			
	V1	C	C Code ISO-9126 Maintainability Level		demo			
	R9	C	C Code ISO-9126 Maintainability Level		demo			
	Prel	E	C Code ISO-9126 Maintainability Level		demo			
	B625	D	C Code ISO-9126 Maintainability Level		demo			
	Beta	B	C Code ISO-9126 Maintainability Level		demo			


The projects list

To consult the results of the analysis, click on the project name to view the Squore Dashboard. More information on how to read the Dashboard is available in Section 4.9, “Where Are My Analysis Results?”.

4.2. Creating Version 2 of My Project


Adding a version to an already-existing project is a simple procedure that is carried out from the **My Projects** page.

Follow these steps to create version 2 of your project:

1. After logging into Squore, click on **My Projects**.
2. Click the **Build** icon () for the Neptune2 project in order to access the source code file options.
3. The first screen of the wizard enables you to specify the version name and to modify some of the project attributes if necessary.

General Information
Data Providers
Confirmation

Project Identification

Project Name:	Neptune2			
Group:	<input type="text"/>			?
Version Pattern:	V#N1#			?
Version Name:	V2			?
Version Date:	<input type="text"/>			?
Color:				?
Automatic Baselining:	<input checked="" type="checkbox"/>			?
Keep old versions data files:	<input type="checkbox"/>			?
E-mail the creator of a version:	<input type="checkbox"/> On draft	<input type="checkbox"/> On baseline	<input type="checkbox"/> On error	?
E-mail team members:	<input type="checkbox"/> On draft	<input type="checkbox"/> On baseline		?

▼ Application attributes

Project Business Value:	<input type="text" value="50.0"/>	FP		
Project Cost:	<input type="text" value="0.0"/>	M/M		
Project Safety Integrity Level:	<input checked="" type="radio"/> SIL0	<input type="radio"/> SIL1	<input type="radio"/> SIL2	<input type="radio"/> SIL3
	<input type="radio"/> SIL4			

Previous
Next
Finish
Cancel

Parameters For the New Version of Neptune2

4. Click the **Next** > button to reach the project language and source settings screen. On this screen, you can modify the path to the source code and point to the newer version. Note that by default, Squore displays the path used when analysing the last version. Leave the path as it was for version 1. We are going to create a version that analyses the same code in this example. If you scroll down to the code analysis option, you will notice that most of them are now disabled. This is because the project configuration was set in version 1 and is not allowed to be modified in subsequent analyses. This ensures that your project is scored using the same criteria every time you analyse new code.

Sources

QuAN Sources

C

Generate control graphs

Qualified names

Limit the analysis depth to file only

Add a 'Source Code' node

Compact folders having only one son

Filter files whose contents match with PERL regexp:

Exclude Include

Filter files whose name matches with Perl-like wildcards e.g. *-test.c*:

Exclude Include

Filter files whose name matches with Perl-like wildcards e.g. Test_*.

Select functional cloning

Select text cloning

Select Open Source cloning

Generate Textual stability

Generate 'Algorithm' stability

Parameters:

Unavailable options when creating version 2 of a project

Note: You can add new sources to the project at this stage if needed. Read more about projects using sources spread over multiple locations in Section 4.8, "Can I Create Projects with Sources From Multiple Locations?".

5. Click the **Next >** button and **Run** to launch the analysis of Neptune2 V2. When the analysis finishes, Neptune2 V2 will be listed in the list of projects on the Projects page.

4.3. Working with Draft and Baseline Versions

This section covers an essential workflow feature of Squore: baselining. While it is possible to keep every version of a project created in Squore, you may want to permanently keep analysis results only for particular milestones and work with an always updating draft version.

You can decide whether a version is a draft or a baseline when you create it, or after the analysis is finished.

4.3.1. Drafts vs. Baseline: The Basic Concepts

The most important thing to remember about a draft version is that it is a snapshot of your data at a given time. You can use it to compare the evolution of your project against the last baseline created. There is therefore only one draft version available per project (the latest version), which Squore creates automatically if your previous version was a baseline. A baseline version, on the other hand, is permanently saved and will not be overwritten the next time an analysis is launched.

When you create a draft version, it is always called Current and can be modified in several ways:

- Forms can be updated
- Attribute values can be modified so that a new value is taken into account in the next analysis
- Artefacts can be manually added, modified or deleted
- Components can be relaxed or removed from the project
- Action Items can have their status changed

Being able to view draft versions of a project is a user privilege that can be granted to users of a particular role, and so is the ability to baseline a project. For more information about roles, refer to Section 3.1, “Understanding Profiles and Roles”. This means that as a project manager, you can give access to every version to users within your team, but can restrict the project visibility to the rest of the company to show them only milestone versions (the ones you baselined). You can also decide which members of your team are allowed to change the status of a version from draft to baseline.


4.3.2. Baselining at Version Creation

Use the Automatic Baselining option on the General Information screen of the project creation wizard to create a draft or baseline as follows:

- When the Automatic Baselining box is unchecked, a draft version is created and all subsequent versions will be draft versions.
- When the Automatic Baselining box is checked, a baseline version is created and all subsequent versions will be baseline versions.

4.3.3. Baselining After Review

You can use the Baseline option on the My Projects page to create a baseline version of the current draft as follows:

1. Log into Squore and click on **My Projects**.
2. Click the Baseline icon () next to the project you want to baseline.
3. Click the **Baseline** button to confirm.

After confirming the baseline creation, you are redirected to the My Projects page and the last draft version becomes the new latest baseline. Note that baselining is only available for users whose role allows the **Baseline Projects** privilege. For more information about roles, consult Section 3.1, “Understanding Profiles and Roles”

Tip

No new analysis is run when you baseline manually. Baselining manually allows to save the modifications made to a draft, but only a full analysis with the Automatic Baselining checkbox

selected ensures that all the modifications made to the draft are fully taken into account in the final project rating.

4.4. Can I Make Changes to My Project?

There are three types of changes you can make to Squore projects:

- Changes to attribute values
- Changes to source code locations
- Changes to some of the Data Provider options

Project attributes are always editable when creating a new version of a project, except for the name of the project.

The location of the source code can always be modified. When editing a project, you can also add more source locations as needed, following the steps described in Section 4.8, “Can I Create Projects with Sources From Multiple Locations?”.

Whether you can edit the settings used in the Data Providers for the project depends on their ability to support edits. This ability is defined by a Squore administrator via the configuration of the Squore wizards. For more information, refer to the Squore Configuration Guide.

4.5. Can I Create a Project Via the Command Line?

Instead of creating a project from the Squore web interface, you can create a project directly from the command line using Squore CLI. Squore CLI is a client for Squore that enables you to create and analyse projects locally and send the results to Squore Server. Alternatively, you can use Squore CLI to instruct Squore Server to carry out the analysis.

If you have installed Squore CLI on your computer, you can call it using Java, passing the parameters you would have passed in the web interface to create projects. The following is an example of the command line you can use to create a project using Squore CLI on Windows:

```
@echo off
java -Dsquore.home.dir="%SQUORE_HOME%" ^
-jar %SQUORE_HOME%\lib\squore-engine.jar ^
--url=http://localhost:8180/SQuORE_Server ^
--commands=DELEGATE_CREATION ^
--name=Mars2 ^
--repository "type=FROMPATH,path=\\server\share\samples\c\Mars2\V3.2.6" ^
--color=rgb(103,25,237) ^
--tag BV=30 ^
--tag COST=50 ^
--version=1.0 ^
--login=demo ^
--password=demo ^
--filter=APPLICATION,MEASURE,LEVEL ^
--wizardId="ISO9126" ^
--dp "type=SQuORE,genAs=true,clAlg=true,languages=c:.c,.h;"
echo done
pause
```

The example above shows how to specify commands, parameters and project options to Squore CLI. This would create a project named **Mars2** in version **1.0**, analysing source code located in `\\server\share\samples\c\Mars2\V3.2.6` with the Data Provider **SQuORE** (the internal name for SQuORE).

You can find more information about using Squore CLI in the Command Line Interface manual, which explains how to install the client and create projects.

4.6. How Do I Connect Squore to My Continuous Integration System?

If you use a Continuous Integration tool like Jenkins or CruiseControl, you can add Squore to your build process and analyse projects every time your code is compiled. This requires the installation of Squore CLI on the continuous integration server, and is therefore described in greater details in the Command Line Interface Manual.

4.7. Can Squore Pull Source From My Version Control System?

The source code analysed by Squore does not have to be located on the same machine as Squore Server or Squore CLI. When you create a project, you get the option to choose from a range of Repository Connectors to pull source code from:

- Direct file system access (local drive, network share, mass storage media...)
- Zip upload
- A ClearCase view
- A CVS checkout
- Git cloning
- A MKS repository
- A Perforce depot
- A Subversion revision
- A Synergy database
- A TFS server

Each option requires different parameters, which can be specified from the project wizard, or via the command line. For more information, refer to Chapter 10, *Repository Connectors*.

4.8. Can I Create Projects with Sources From Multiple Locations?

Squore provides support for analysing projects whose sources are spread over several locations or version control systems. If your source code resides in `/products/common` and `/projects/myproject`, you can specify these two locations in the Squore project wizard by clicking the **Add Repository** button. Similarly, if some of your code is managed by a SVN repository and the rest is handled by a Git server, you can configure both locations as part of the same project, as shown below:

Specify Repository Locations

Folder
 Zip Upload
 ClearCase
 CVS
 Git
 SVN
 Synergy
 Remove

Artefact Name:

Uri:

Revision (optional):

No credentials
 Use my credentials
 Define credentials

Folder
 Zip Upload
 ClearCase
 CVS
 Git
 SVN
 Synergy
 Remove

Artefact Name:

Uri:

Revision (optional):

No credentials
 Use my credentials
 Define credentials

Folder
 Zip Upload
 ClearCase
 CVS
 Git
 SVN
 Synergy
 Remove

Artefact Name:

Datapath:

A project using sources from two SVN repositories and a network drive

4.9. Where Are My Analysis Results?

Now that you have created a project, you are ready to start reviewing the analysis results in the main section of Squore, the Explorer, which consists of a set of trees for browsing through project artefacts and various dashboards to display the information associated with these artefacts.

Dashboard
Action Items
Highlights
Findings
Forms
Comments
✕

Maintainability Level

Ex: Artefact, %fact

Key Performance Indicator

More efficient

Less efficient

← E

Decision Making		
Maintenance Performance		
Maintenance Performance	-3.5% ↑	D ↗
Technical Debt Variation	37.9% ↑	I
Project Size Variation	34.4% ↑	I
Code Stability Index	73.5% ↓	70/100 ↘

Rule Checking		
Non Conformity Count	222 ↑	I
Non Compliant Rules	15 ↑	G →
Adherence to Standards	42.3% ↓	40/100 ↘
Code Cloning	4.00 ↑	D ↘













Maintainability Breakdown

Function Level

Statements vs. Function Maintainability

The Squore Explorer

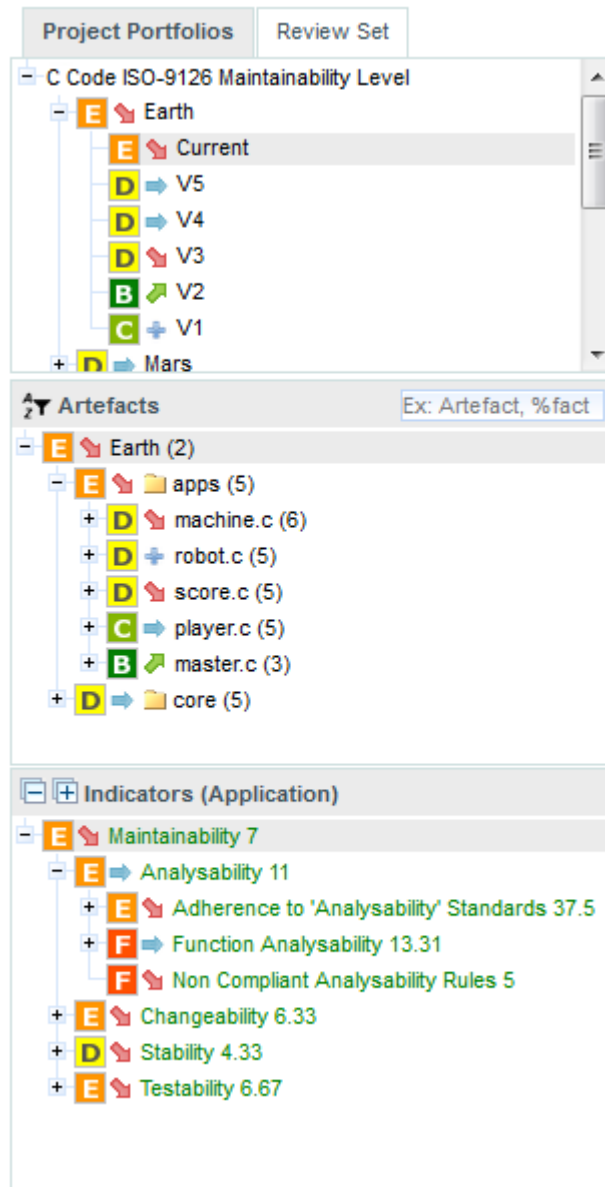
Common icons are used throughout the explorer to indicate the rating of a component and its evolution compared to the previous version. The image below shows the meaning of the different icons used:

Level	Evolution
 Unknown	 New
 Level A	 Deteriorated
 Level B	 Improved
 Level C	 Stable
 Level D	
 Level E	
 Level F	
 Level G	

The Squore Explorer icons

4.9.1. The Tree Pane

The left-hand part of the Explorer is a three-panel section containing expandable trees.

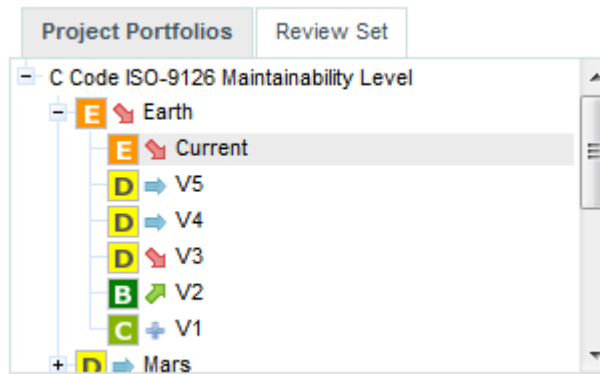


The Tree Pane

The top panel contains the **Project Portfolios** and the **Review Set**.

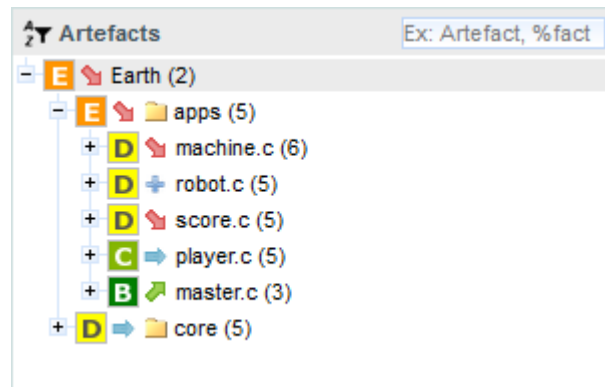
The Project Portfolios is a list of all the projects you have access to, grouped by analysis model. Each project is listed with its latest rating and evolution and can be expanded to show all versions of the project that were analysed with Squore.

The Review Set is a flat list of artefacts you collect from various projects in order to review them. This list is saved when you log out and log into Squore again.



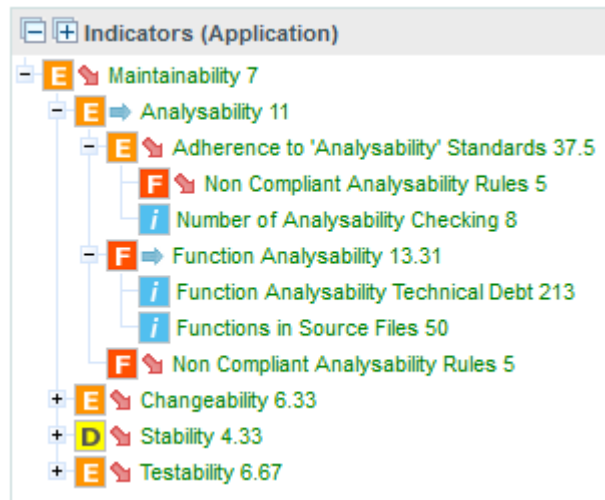
The expanded Earth project, rated E, and its 6 versions in the Project Portfolios

The tree in the middle panel is the **Artefact Tree**. When Squore analyses a project, it breaks it down into artefacts of various configurable types, for example APPLICATION, PROGRAM, FOLDER, FILE, CLASS or FUNCTION, according to the analysis model used. The artefacts in the tree are displayed for the version selected in the Project Portfolios. clicking a different version of a project refreshes the artefact tree with the ratings for the version just selected. Above the artefact tree are tools for filtering, pinning, sorting and searching artefacts. Each artefact is displayed with its current rating and can be expanded to reveal child artefacts if available. The number in brackets indicates the amount of child artefacts for the current artefact. You will learn about these tools later in Section 5.1, “Has the Quality of My Project Decreased Since the Previous Analysis?” and Section 5.2, “How Do I Find and Keep Track of Artefacts?”.



The Artefact Tree for version 6 of the Earth project

The bottom panel is the **Indicator Tree**, in which ratings for the indicators defined in the analysis model at the current level are displayed. Each indicator can be expanded to display the rating of each of its sub-indicators. The Indicator Tree displays statistics for the artefact currently selected in the Artefact Tree and refreshes when the selection is changed. The type of artefact selected is indicated in brackets. Two shortcut buttons can be found above the top node to quickly expand and collapse the entire tree.



The partly expanded Indicator Tree for version 6 of the Earth project at Application level

Clicking one of the tree nodes reveals more information about the indicator, including the formula used by Squore to compute its value and rating.

Comment Rate ✕

Mnemonic: COMR

Description: Ratio between the number of lines of comments and the number of source lines of code.

Value: 16.67

Rating: C

Computation: $IF(ELOC=0,-1,(CLOC+MLOC)*100/(ELOC+CLOC))$

Scale: Scale COMR

☹]-∞; 0.0[
G	[0.0; 0.0]
F]0.0; 5.0]
E]5.0; 10.0]
D]10.0; 15.0]
C]15.0; 20.0]
B]20.0; 25.0]
A]25.0; +∞[

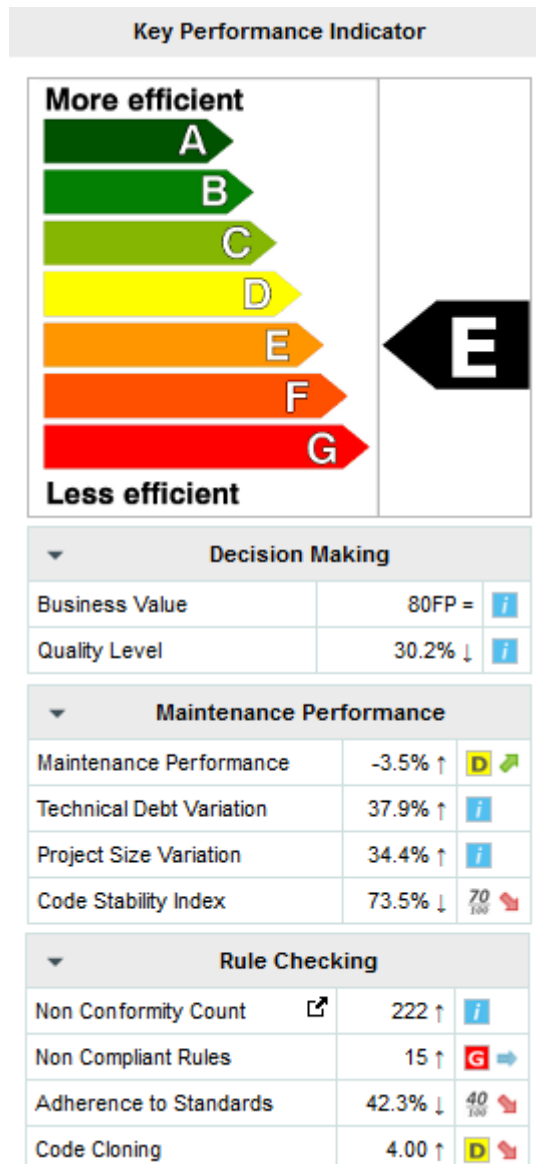
The popup displayed when clicking the Comment Rate indicator for a function

4.9.2. The Dashboards

The right-hand side of the Squore Explorer contains a series of tabs, the first of which is the Dashboard. The Dashboard is dynamic and always displays information about the artefact currently selected in the artefact tree. There is not one Dashboard, but a Dashboard per node in the tree. Additionally, the Dashboard can be customised by a Squore administrator so that users see a different Dashboard according to their role in a project, thus highlighting different information for project managers, quality engineers and developers for

example. Ask your Squore administrator about Dashboard customisation, or refer to the Squore Configuration Guide for more information.

The left-hand area of the Dashboard contains the score card , which consists of a graphical representation of the key performance indicator for the current artefact, and some tables highlighting key metrics about the project.



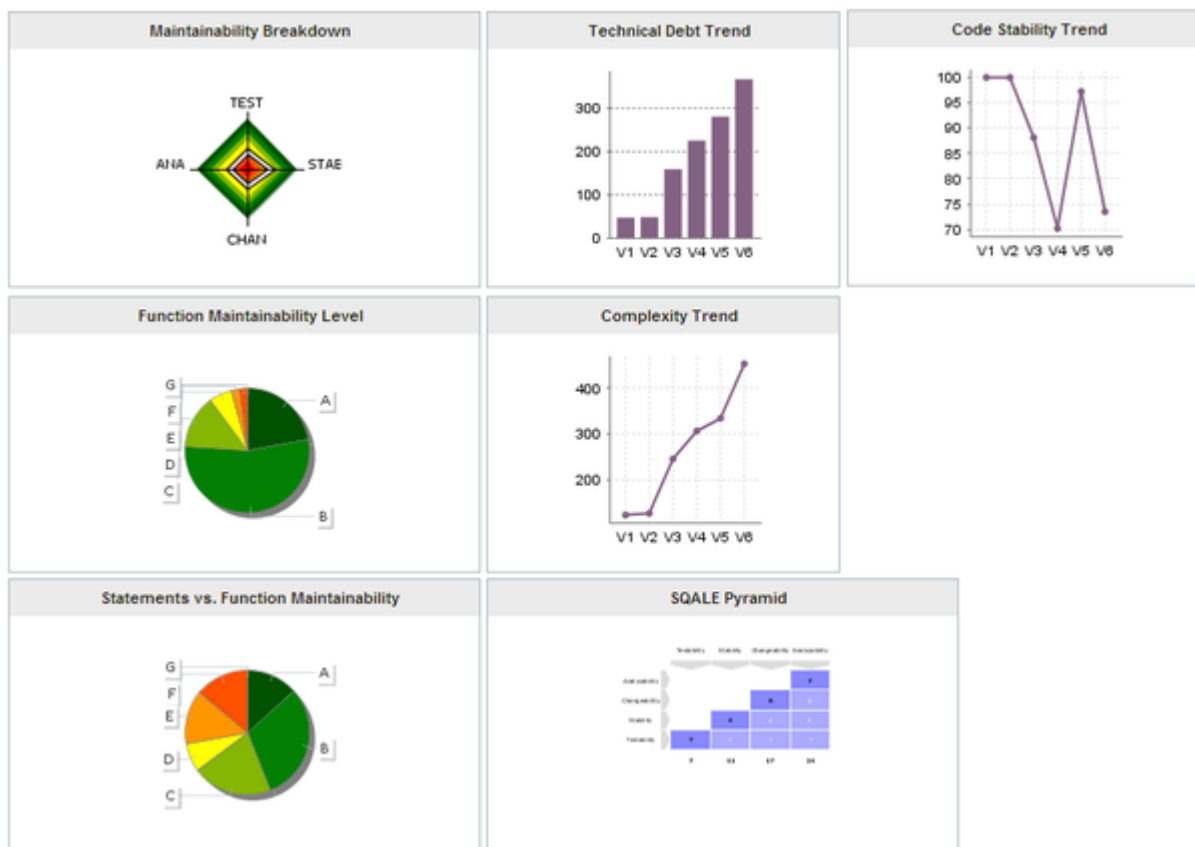
The score card area

Each table lines display a series of details about the key performance indicator:

- The name of the metric (e.g. **Maintenance Performance**). When clicked, a popup shows the way the metric is computed. Optionally, some metrics may allow an extra link to be displayed. This link shows the list of findings taken into account when calculating this metric (See **Non Conformity Count**).

- The raw value of the metric and its evolution according to the previous version (e.g. **-3.5% ↑**). Clicking a value in this column displays a chart of the history of the last 10 values recorded for this metric.
- If the metric displayed is an indicator, the rating of the indicator is displayed, along with its evolution (e.g. **Level D, improved**). If the metric is a measure, then an information icon is shown. In both cases, you can click the information in this column to display more details about how the metric is computed.

The right-hand area of the Dashboard contains a series of graphs representing key information about the current artefact. Clicking a graph opens a larger version of the image so you can analyse the data. Note that the available graphs will differ depending on the type of artefact selected in the tree. Files and functions for example include a **Source Code** chart (for users who have the privilege to browse source code), which does not appear in the Dashboard for folders and applications.



The graphs area

The Dashboard is only the first of a series of tabs in the Explorer. In the following chapter, you will find out more about the role of the **Action Items, Highlights, Findings, Forms, Reports, Indicators, Measures** and **Comments** tabs. Note however that like the Dashboard, the information displayed in each tab is always relative to the node currently selected in the Artefact Tree.

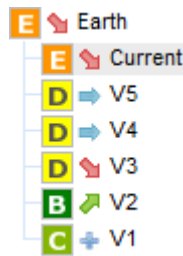
5. Understanding Analysis Results

This chapter covers a list of Squore use cases for various features of the Explorer. By the end of the chapter, you should be able to make the most of the information and decisions presented by Squore and start applying them to improve your development practices.

5.1. Has the Quality of My Project Decreased Since the Previous Analysis?

After completing the analysis of a new version of your project, you will probably want to investigate how it has evolved, more specifically for which artefacts the quality has decreased. Let's look at the current version of the Earth Project (which should be available if your Squore administrator has created the sample projects shipped with the Squore installation) to find out how to spot the worst-scored components in your project.

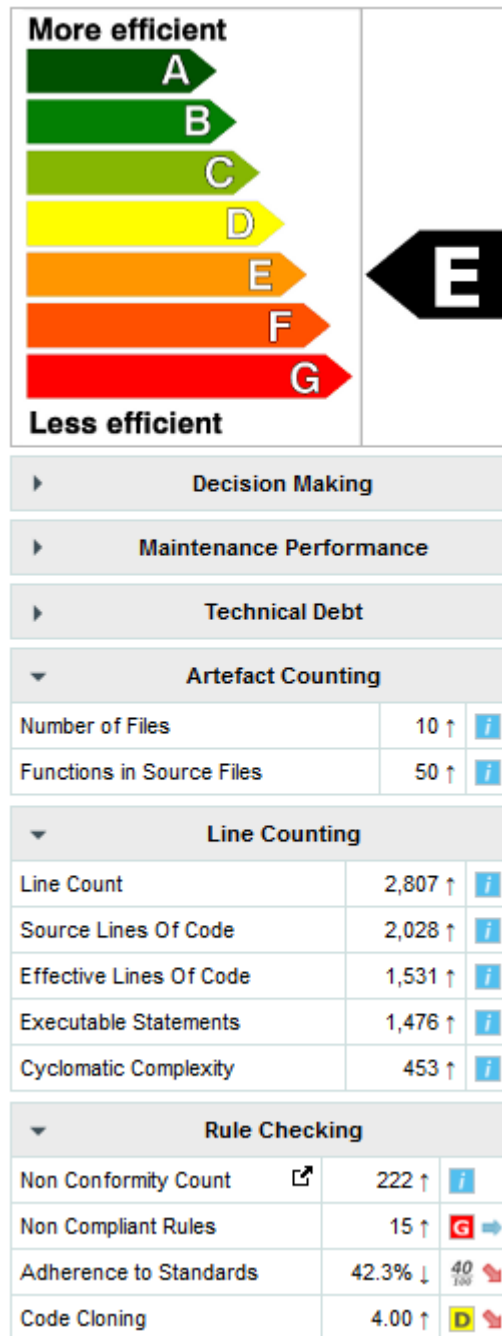
Log into Squore as the demo user using `demo/demo` and observe the evolution of the Earth project in the Project Portfolios:




The versions of the Earth Project

The downward arrow before the project name in the the Project Portfolios indicate that the quality of the last version decreased. By expanding the Earth project to show the version history, you learn that the quality of the current version decreased compared to V5, which itself was worse than V4. (More information about the quality indicator icons is available in Section 4.9, “Where Are My Analysis Results?”.)

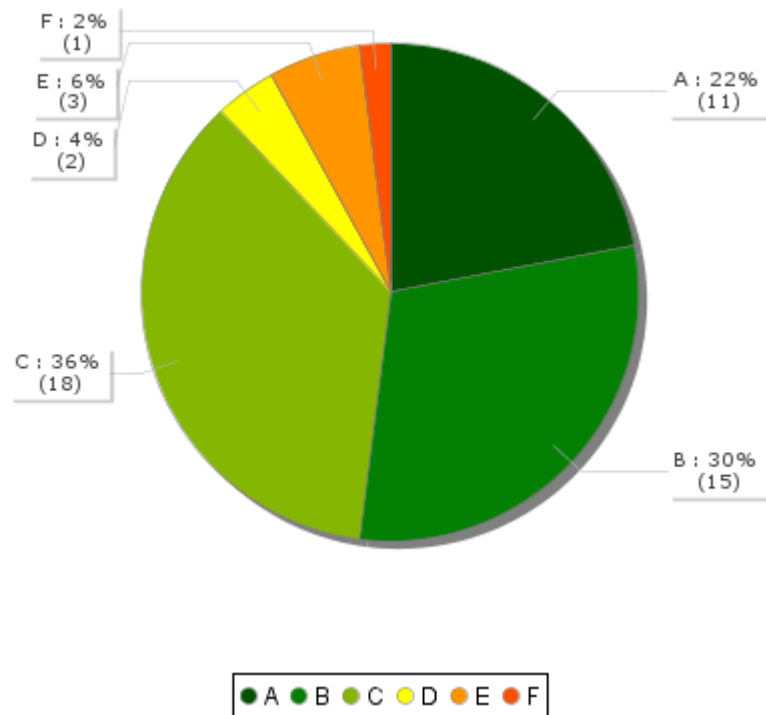
You can start evaluating the version by looking at the score card, which rates Earth at E. Some values under **Artefact Counting**, **Line Counting** and **Rule Checking** explain the lower score: the application contains more files and functions, more lines of code, less comments and more rules violations.



The score card for the current version of Earth

By now you probably want to find out which components in your project are responsible for lowering the score of the application in this version. If you want the Artefact Tree to reflect this information, you can change the sort order to show the worst scores first by clicking on the **sort** icon () and selecting **LEVEL > Worst first** to display artefacts from worst-scored to best-scored.

The Function Maintainability Level chart is a good place to begin to understand how bad some of the components in your project are affecting the overall score.



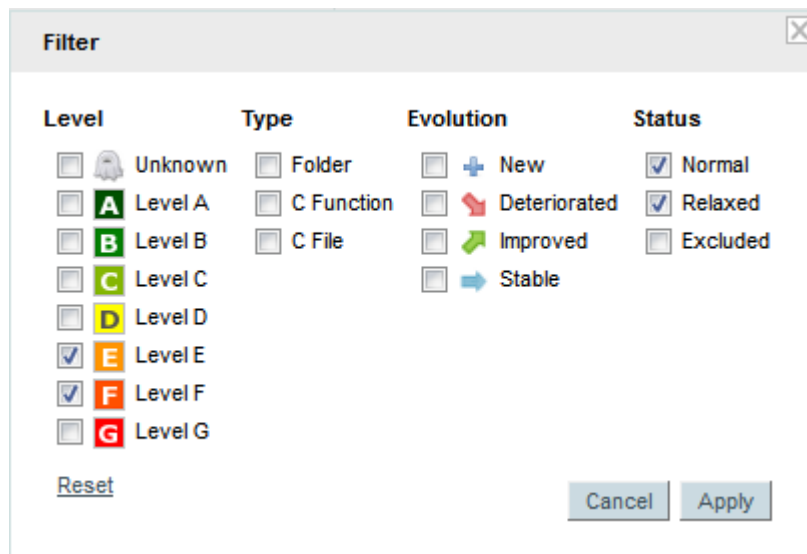
The Function Maintainability Level chart

From the chart above, you can read that 6% of all functions are rated E or F. It is a good idea to go look for these functions in the Artefact Tree and see what is wrong with them. There are several ways to do this: via filtering and Searching, or using the Highlights feature.

5.1.1. Finding Artefacts Using Filters and Search

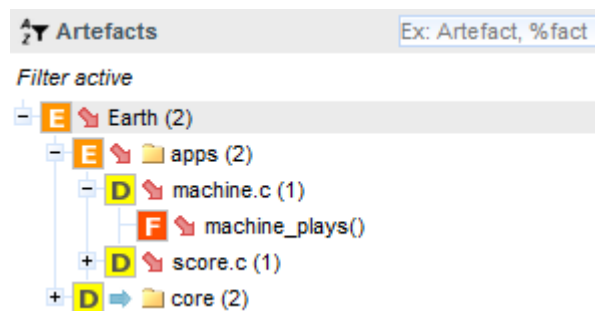
In this section, you will learn to look for artefacts the hard way by using filters and search. For a more automated way to find artefact that fit a specific category, take a look at Section 5.1.2, "Finding Artefacts Using Highlights".

Click the Filter icon (🔍) above the Artefact Tree to show the filter options and select only the levels E and F:



The filter popup with the boxes checked to filter artefacts rated E or F

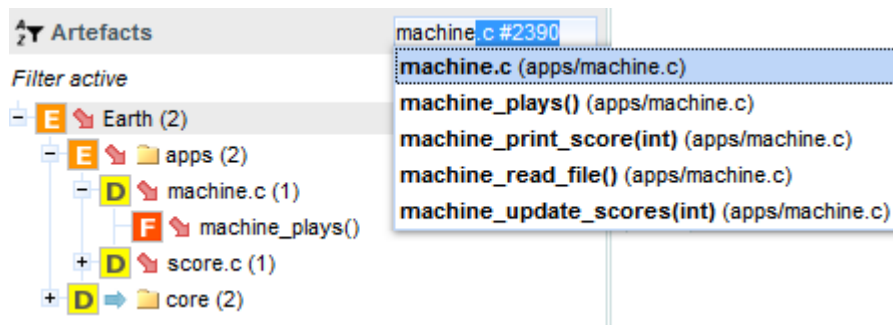
Click **Apply** to apply the changes, and the Artefact Tree should refresh to show only artefacts in the levels selected, as shown below:



The filtered Artefact Tree showing artefacts rated E or F

The notice **Filter active** is always displayed above the Artefact Tree when you are using a filter. The tree now only shows artefacts rated E or F, along with their ancestors (i.e. their parents and their parents' parents), even though the ancestors may not be rated E or F.

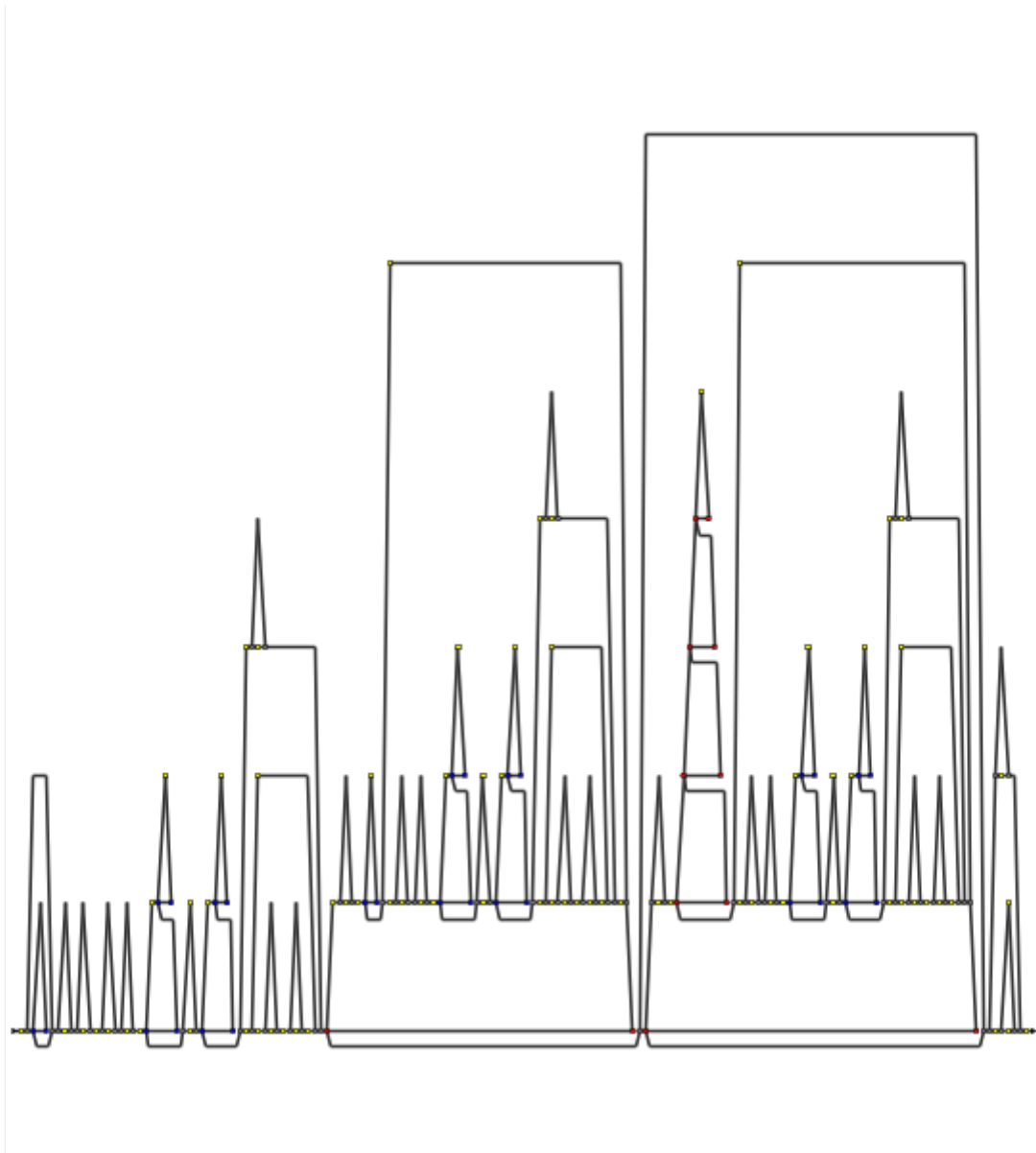
While a filter is active, you can still search for other artefacts by typing a search term in the search box. Try typing **ma** in the search box above the Artefact Tree, and watch the search results list get populated as you type:



The search results for the term entered in the search box

If you select the first search result in the list above, you will open the dashboard for `machine.c`.

Let's go back to our filtered tree. The filter singled out two artefacts with the required score range: `machine_plays()` and `send_score(int)`. The function `machine_plays()` deteriorated in version 6. Click on the function `machine_plays()` to view its dashboard. Note how the score card indicates that the cyclomatic complexity, non-conformity count and non-compliance to rules have all gone up. Now click on the Control Flow Chart, to see the function's complexity.

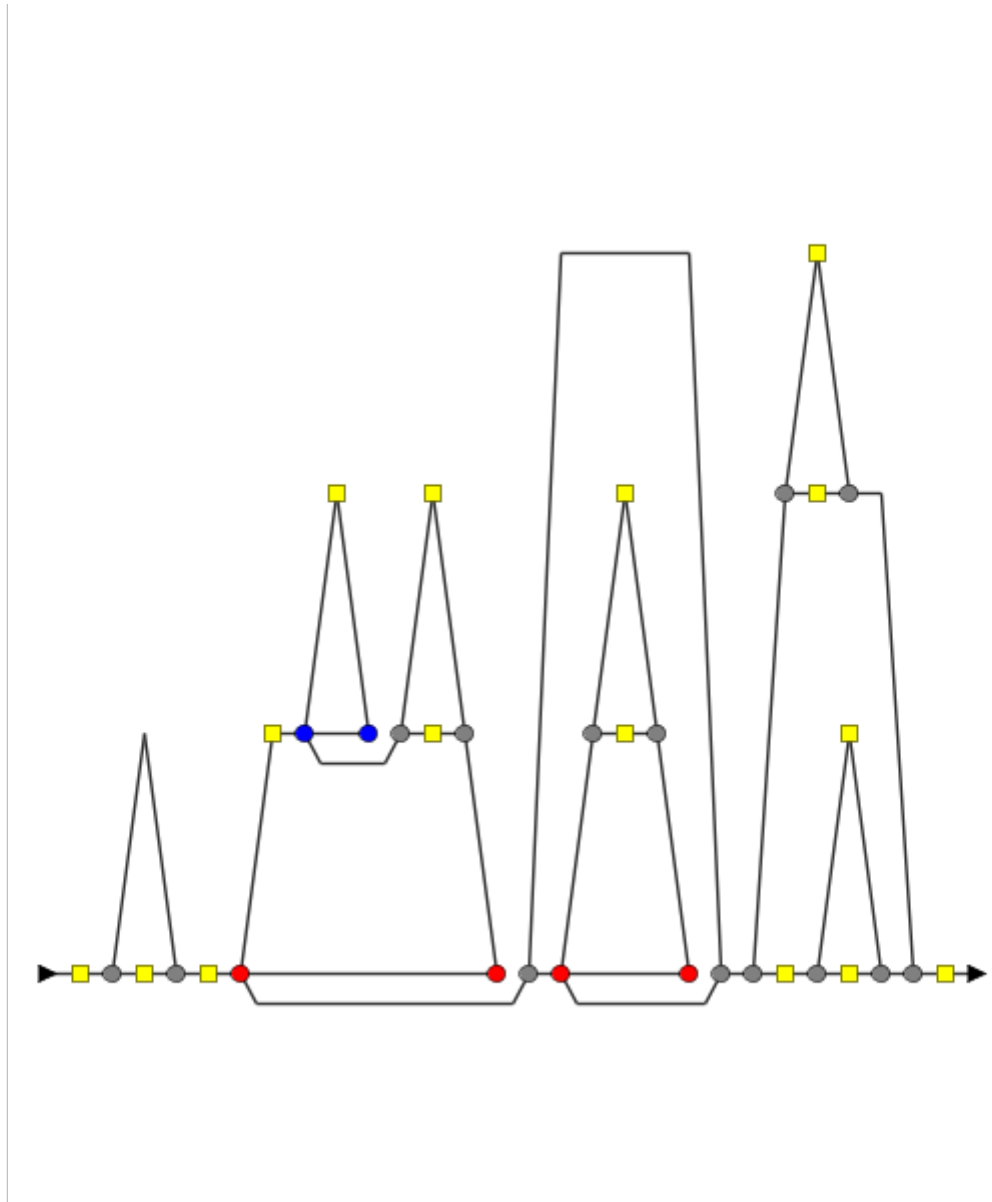


The Control Flow Chart for machine_plays()

The Control Flow Chart is a graphical representation of the way the code of a function can be executed. The line representing the entry into the function starts on the left and branches every time it reaches a logical operator, represented on the chart as described below:

- Gray circle: If, End If
- Blue circle: While, End While, Do, End Do and GOTO
- Red circle: For, Foreach, End For, End Foreach and Return
- Pink circle: Switch, End switch, Try, End Try, Catch, End Catch, Finally, End Finally
- Green circle: Break, Continue
- Yellow rectangle: Statement

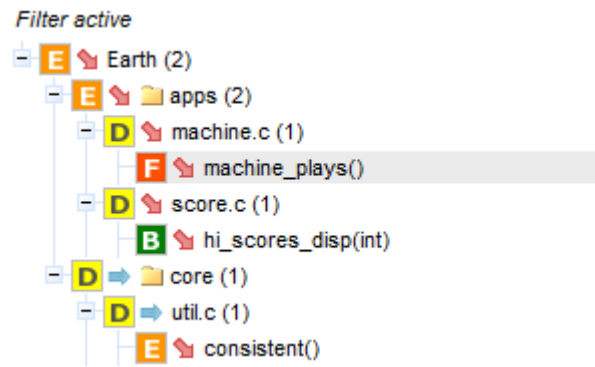
More importantly, you can compare this version's Control Flow chart with the one generated for the previous analysis of the Earth project. Click on V5 in the the Project Portfolios and watch the dashboard refresh to show the analysis results for version 5. Observe how much simpler the control graph is for this version:



The Control Flow graph for version 5 of machine_plays()

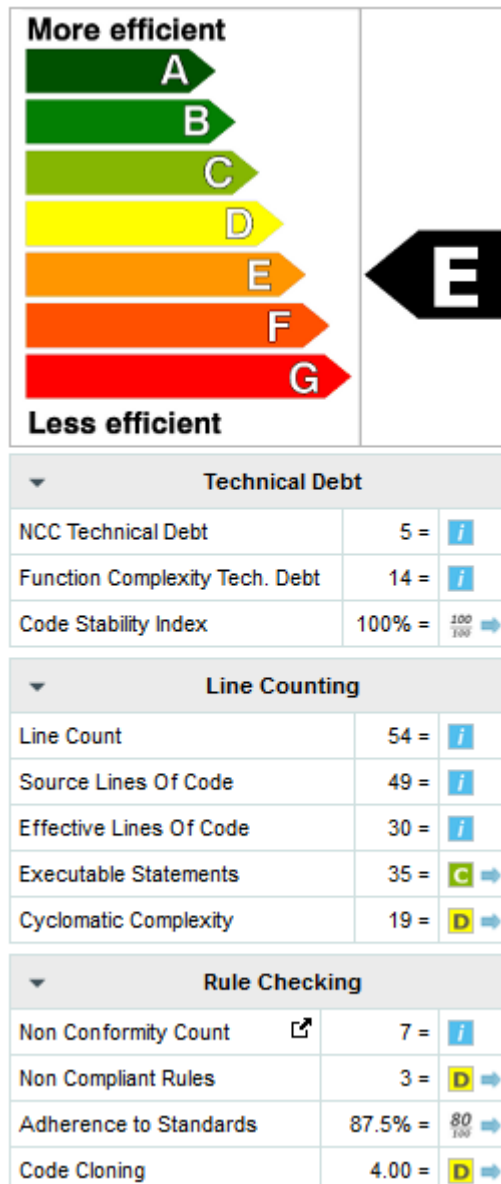
Another convenient way to try and find why a project's quality is deteriorating is to filter on the evolution of an artefact.

Select the current version of Earth again and edit the active filter: Uncheck the boxes for levels E and F, and select the **Deteriorated** category in the Evolution column. When applying the filter, you should see the artefacts in the tree that have the 🚩 icon next to their name.



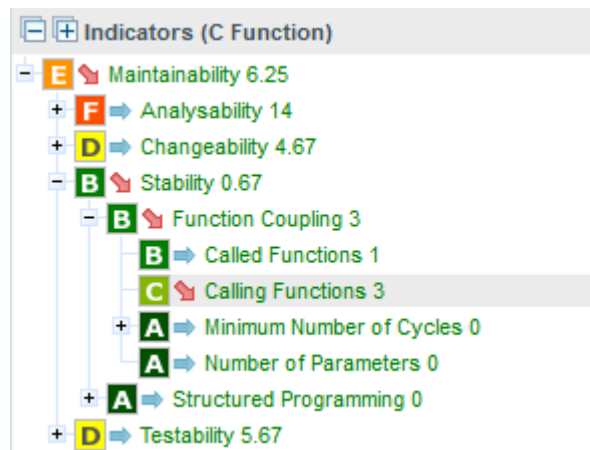
The artefacts that deteriorated in the current version of Earth

The functions you looked at earlier are still here, but there are more deteriorated artefacts that you can start reviewing. If you click on `consistent()` for example, which is rated E, it is not immediately obvious why the function has deteriorated: the score card indicates that the cyclomatic complexity remained constant, so did all the entries in the Rule Checking table. This is shown in the image below:



The score card for `consistent()`

In order to find out where the degradation took place, you can look at the indicators tree to understand where the decline in quality comes from. Expand all the nodes in the indicators tree to reveal the issues with the function:



The Indicator Tree for consistent()

Squore makes it easy to spot the irregularities quickly, like the fact that the **Calling Functions** causes many of the indicators to get a worse score in the latest version than before. This is probably the first item to review in this function.

You can dive further into the analysis results by looking at the information contained in other tabs and assign action items to your team by referring to Chapter 6, *Managing Your To-Do List With Squore* or report your progress as explained in Chapter 8, *Communicating With Squore*.

5.1.2. Finding Artefacts Using Highlights

In the previous section (Section 5.1.1, “Finding Artefacts Using Filters and Search”), you got familiar with searching and filtering to find the artefact that have a negative impact on the overall rating of a project. in this section, you will learn to master the Highlights functionality, which aims to make the process of finding certain categories of artefacts easier.

Highlights are flat lists of artefacts ordered in predefined categories for a model.

Let's try to confirm our findings about the worst and most deteriorated artefacts in Earth. Click on the current version of Earth and clear the filter. Click the **Highlights** tab of the Explorer and select the **Top 10 worst artefacts** category. The list appears as shown below:

Dashboard	Action Items	Highlights	Findings	Forms	Comments	✕
Top 10 worst artefacts		C Function				
<input checked="" type="checkbox"/>	Rating ↕	Artefact ↕	Path ↕			
<input checked="" type="checkbox"/>	F	machine_plays()	apps/machine.c			
<input checked="" type="checkbox"/>	E	send_score(int)	apps/score.c			
<input checked="" type="checkbox"/>	E	cf_9_anifelsif()	core/control.c			
<input checked="" type="checkbox"/>	E	consistent()	core/util.c			
<input checked="" type="checkbox"/>	D	consistent_ex()	core/newutil.c			
<input checked="" type="checkbox"/>	D	instruction()	core/write.c			
<input checked="" type="checkbox"/>	C	format_output(char*,int)	core/base.c			
<input checked="" type="checkbox"/>	C	cf_10_asequenceofif()	core/control.c			
<input checked="" type="checkbox"/>	C	refresh()	core/util.c			
<input checked="" type="checkbox"/>	C	waiting_loop()	core/write.c			

The Top 10 worst artefacts in the current version of Earth

The list confirms that the artefacts with the worst rating are `machine_plays()` and `send_score(int)`. The Highlights table shows you the artefact rating, name and path, and allows you to go to the artefact's dashboard directly by clicking the artefact name.

Now you can also find the deteriorated artefact `consistent()` that you identified with a filter earlier in Section 5.1.1, "Finding Artefacts Using Filters and Search": select the Top 10 most deteriorated artefacts highlight category to see the artefact appear in the list of deteriorated artefacts in this version.

Top 10 most deteriorated artefacts		C Function		
<input checked="" type="checkbox"/>	Rating ↕	Artefact ↕	Sort and Filter ↕	Path ↕
<input checked="" type="checkbox"/>	F	machine_plays()	10.75	apps/machine.c
<input checked="" type="checkbox"/>	B	hi_scores_disp(int)	0.75	apps/score.c
<input checked="" type="checkbox"/>	A	make_code(guess*)	0.25	core/base.c
<input checked="" type="checkbox"/>	C	refresh()	0.25	core/util.c
<input checked="" type="checkbox"/>	E	consistent()	0.25	core/util.c

The Top 10 most deteriorated artefacts in the current version of Earth

Artefacts are sorted by degradation, i.e. the difference between the value of the main indicator in the previous baseline version compared to the current value. By clicking the Export, you can export the selected items to a CSV file.

Tip

By default, the list of most deteriorated artefacts is compiled based on the previous version. By using the Reference list in the Explorer Settings menu (✕) and choosing another reference version, you can update the statistics based on the new reference version you just selected.

Note: The notions of baseline and draft versions is explained in Section 4.3, “Working with Draft and Baseline Versions”.

The following categories are available for all default models:

- Top 10 worst artefacts
- Top 10 most deteriorated artefacts
- Top 10 most improved artefacts
- Top 10 'borderline' artefacts
- All new artefacts
- All modified artefacts

Squore administrators can customise and expand this list by referring to the Squore Configuration Guide.

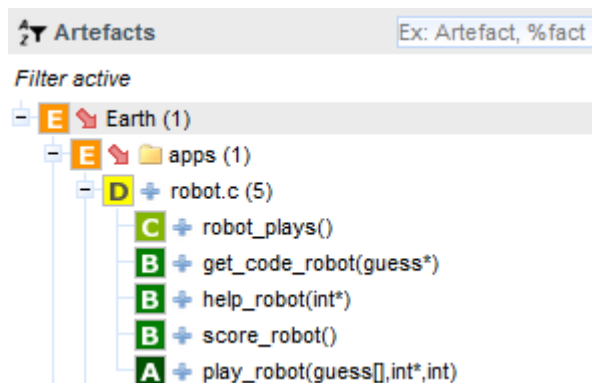
5.2. How Do I Find and Keep Track of Artefacts?

For some projects, you may want to collect artefacts so you can review them later. Squore enables you to build a Review Set, a flat list containing artefacts that you want to keep track of. Let's log in as the demo user to review all the new artefacts added to a project, in order to evaluate their level of quality.

Isolating the new artefacts can be done in three steps:

1. Log in using the demo user (demo/demo).
2. Click on **Earth** in the Project Portfolios to display the dashboard for the last version of the project.
3. Click the Filter icon to display only items in the Evolution column with the status **New** and apply your changes

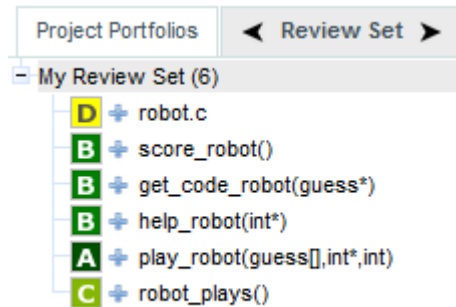
You should see the following artefacts in the Artefact Tree:



The new artefacts in the current version of Earth

Squore makes it easy for you to keep track of these artefacts. Click on the icon above the Artefact Tree and select **Add Filtered Results to Review Set**.

You can now clear your filter, the artefacts you want to review are stored in your Review Set. Click the **Review Set** tab in the left pane of the explorer to find the items you just saved.



The Review Set filled with new artefacts for the new version of Earth

At any moment, the artefact currently selected in the Artefact Tree can be sent to the Review Set as well. Simply display the context menu for an artefact and click **Add to Review Set** to add it to the Review Set. Clicking an item in the Review Set pane has the same effect as clicking it in the Artefact Tree: the dashboard refreshes to show the information for that artefact. You can use the left and right arrows in the Review Set pane to go to the previous and next artefact in the list.

If you want to know more about what actions you can take after reviewing artefacts, refer to Chapter 6, *Managing Your To-Do List With Squore* and Chapter 8, *Communicating With Squore*.

5.3. How can I Understand and Enhance My Model?

Squore provides tools to understand, verify and enhance your model under the Models menu.

- The **Viewer**, a graphical representation of all the analysis models on Squore Server
- The **Validator**, a debug tool for model writers
- The **Dashboard Editor**, which allows customising the dashboards that all users will see
- The **Analysis Model Editor (beta)**, which allows turning rules on or off in your model

Users whose profile grants the "View Models" permission have access to the first two tools.

Users whose profile grants the "Modify Models" permission have access to the last two tools.

5.3.1. Viewer

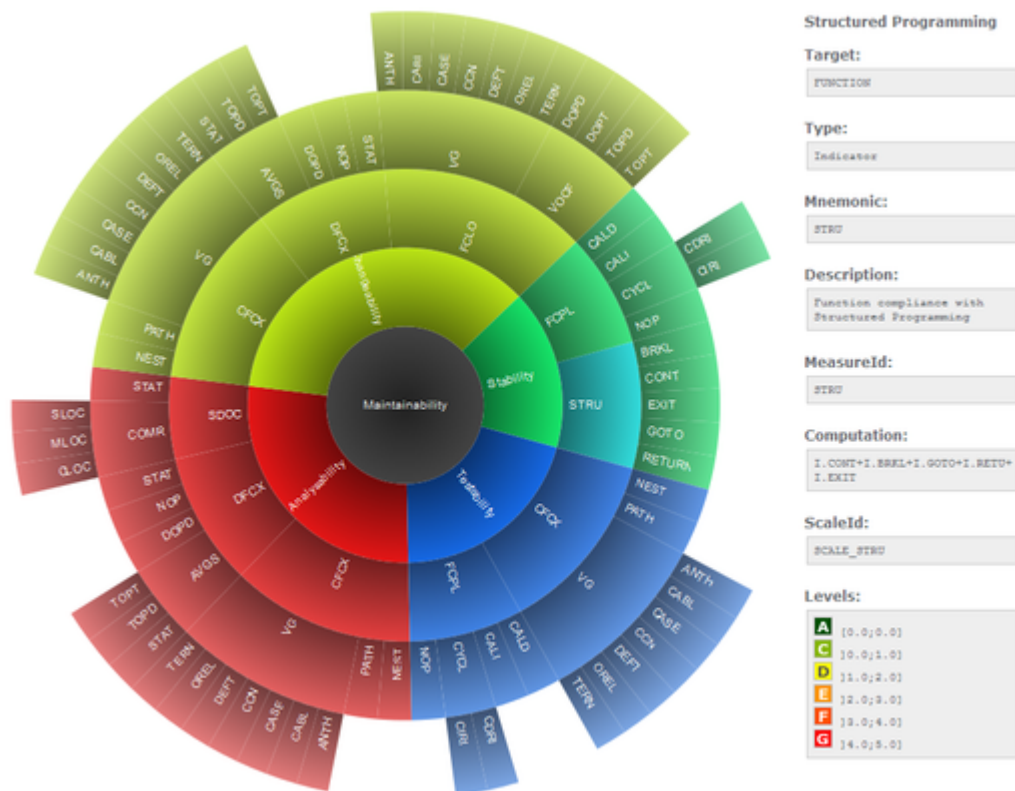
To use the Viewer:

1. Click **Models > Viewer** in the toolbar.
2. Select the analysis model you want to browse.
3. Select the artefact level you want to browse.
4. Choose your preferred graphical representation between Space-Tree and Multi-level pie.
5. Select whether measures are displayed using their full name or their mnemonic.

Upon selecting the parameters above, the page is refreshed with the top-level indicators in the model, and you can click each indicator to unveil sub-indicators and their characteristics. You can drag the tree left and right to reveal all sub-levels if necessary. For each indicator selected, Squore displays the following information:

- **Target** is the target artefact type for the selected item
- **Type** is the type of the selected item
- **Mnemonic** is the short code for the selected item
- **description** is the description of the selected item
- **Data Provider** is the Data Provider responsible for computing the selected item
- **MeasureId** is the measure ID of the selected item
- **Computation** is the formula used to compute the value of the selected item
- **ScaleId** is ID of the scale associated with the selected item
- **Levels** is the list of levels available for the selected item and their ranges

Note: The information available depends on the type of item selected.



The iso9126 model presented as a multi-level pie in the Viewer

5.3.2. Validator

If your work involves adjusting the model's metrics or dashboard, you can use the Validator to verify its integrity during as you make changes. Click **Models > Validator** to display the diagnostics organised by category, as shown below:

Model: ISO9126_Maintainability_C ▼

Summary	Analysis	Decision	Description	Dashboards	Wizards	Exports	Reports	Highlights	Properties
[INFO]: Analysis: INFO (12) [INFO]: Decision: OK [INFO]: Description: OK [INFO]: Dashboards: INFO (2) [INFO]: Reports: OK [INFO]: Wizards: OK [INFO]: Exports: OK [INFO]: Highlights: OK [INFO]: Properties: OK									

The iso9126 model in the Validator

The Summary tab displays a summary of all the diagnostics run for each category. By clicking any of the other tabs, more details are shown about potential problems found in your model. You can also show the complete XML of the model to understand the errors reported. The XML can be searched by using the Ctrl +F key combination to bring up the search dialogue, and then Ctrl+G to search for the next occurrence of the search term:

Model Validator

Model:

Summary Analysis Decision Description **Dashboards** Wizards Exports Reports Highlights

See XML Model

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE BUNDLE>
3 <roles>
4   <role name="Default">
5     <dashboard type="Model" nbColumns="2" defaultWidthValue="400" defaultHeightValue="400"
6       xml:base="file:///C:/Documents/src/Squore-2012-
7       B/ProfessionalServices/squoring/6G/configuration/models/process-
8       improvement/Dashboards/dashboard_am.xml">
9       <charts>
10        <chart type="Quadrant" id="MAINTAINABILITY_VS_BUSINESS_VALUE" width="400"
11          height="400">
12          <xindicator>MATURITY_LEVEL</xindicator>
13          <yindicator>TESTS</yindicator>
14          <zindicator>DELTA_QUALITY</zindicator>
15          <area xMin="0" xMax="100" yMin="0" yMax="100" />
16          <markers>
17            <marker value="50" red="80" green="255" blue="0" alpha="50" isVertical="true" />
18            <marker value="50" red="80" green="255" blue="0" alpha="50" isVertical="false" />
19          </markers>
20        </chart>
21      </charts>
22      <table id="PROJECT_SUMMARY" hideLastVersion="true" hideCreator="true" hideGroup="true"
23        hideLanguage="true" hideLevel="false">
24        <column indicatorId="MATURITY_LEVEL" headerDisplayType="NAME" displayType="ICON"
25          decimals="0" suffix="" />
26        <column indicatorId="TESTS" headerDisplayType="NAME" displayType="ICON" decimals="0"
27          suffix="" />
28      </table>
29    </dashboard>
30  </role>
31 </roles>
    
```

[ERR]: No id and name for table in dashboard: APPLICATION
 [ERR]: Dashboard APPLICATION -> Chart KVIAT (APPLICATION) -> Number of 'indicator' children (2) is not between 3 and 50.
 [WARN]: Dashboard APPLICATION -> Chart Quadrant -> [Deprecated] use of elements xindicator instead of xmeasure.
 [WARN]: Dashboard APPLICATION -> Chart Kviat -> Invalid variable RATIO_BAD_FU.
 [WARN]: Dashboard APPLICATION -> Chart Kviat -> Invalid variable RATIO_BAD_FOLDER.
 [WARN]: Dashboard APPLICATION -> Chart Kviat -> Invalid variable RATIO_BAD_PACKAGE.
 [WARN]: Dashboard APPLICATION -> Chart OptimizedTemporaEvolutionStackedBar -> Invalid variable DELTA_FUNCTION_LEVEL.
 [WARN]: Dashboard APPLICATION -> Chart TemporaEvolutionLine -> Invalid variable R_TEST_MANUELS.

The Validator reporting errors

Your Squore administrator can help you get more information model development. You can also refer to the Squore Configuration Guide for a complete reference.

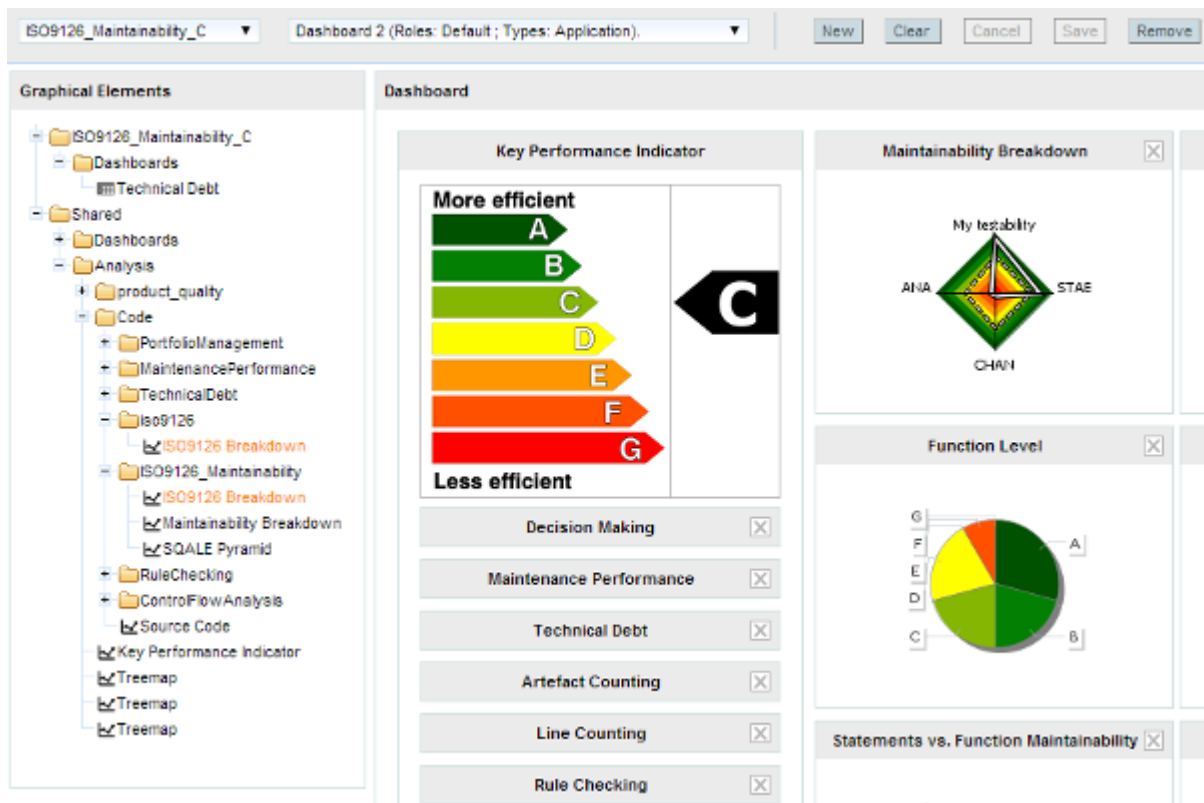
5.3.3. Dashboard Editor

The Dashboard Editor is a graphical editor for the dashboards of a particular model. Dashboards consist of a key performance indicator, a list of tables and one or more columns of predefined charts. With the Dashboard Editor, you can rearrange the information shown on the dashboard for all users, or create a completely new dashboard for a new role in your project.

In order to use the Dashboard Editor:

1. Click **Models > Dashboard Editor**
2. Select a model and load an existing dashboard

The current dashboard skeleton is loaded in the editor, as shown below:



The iso9126 model in the Dashboard Editor

The tree contains the list of pre-configured graphical elements that you can add to your current dashboard. When you hover over a dashboard element, a tooltip explains what metrics it displays. You can drag and drop an element onto the current dashboard, drag charts and tables to rearrange them. When you are satisfied with your changes, you can save your modifications. You can also create a new dashboard, using an existing one as a basis for the new one, or start from a blank canvas.

Tip

In the tree, graphical elements are colour-coded so that you know before you add them to your dashboard if they are compatible with your model. Blue elements have minor incompatibilities with your current model and orange elements are likely not compatible. Use the tooltip for an element to understand why the element is flagged as incompatible.

More detailed explanations about the Dashboard Editor can be found in the Squore online help.

5.4. Reviewing Multiple Projects

Project Managers may be interested in monitoring several projects as a whole. Squore provides a special dashboard view which compounds information about several projects into an analysis model dashboard, which can help you prioritise projects according to their current status.

In order to view the analysis model dashboard:

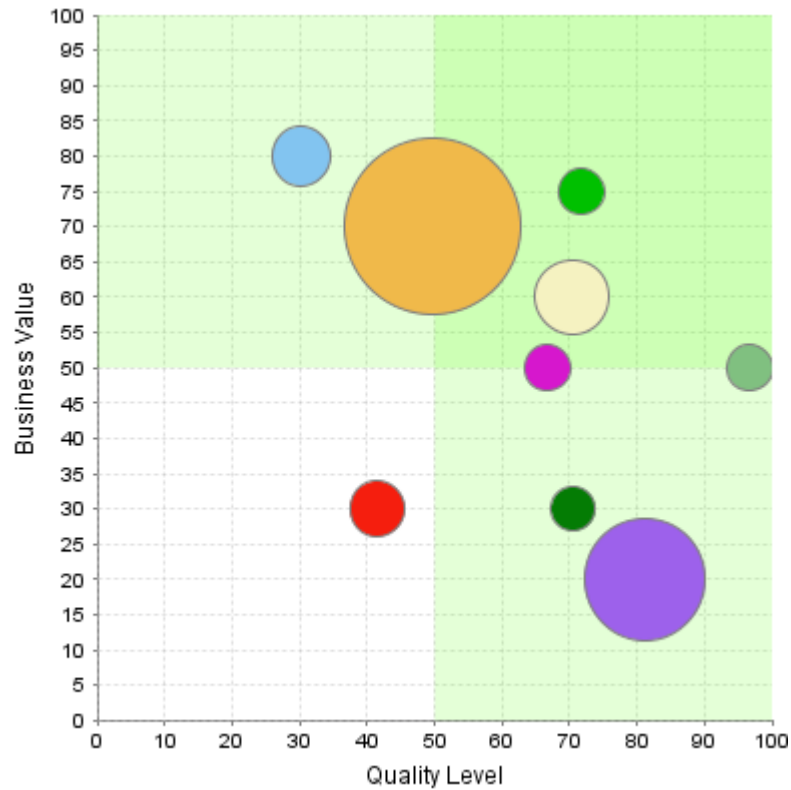
1. Log into Squore with the demo user.
2. Click the model name "iso9126" in the Project Portfolios.

The dashboard refreshes to show the compounded information for all projects analysed with this model using Quadrant charts and tables:



The analysis model dashboard for iso9126 projects

In the quadrants, each project is represented as a bubble. Two indicators define the horizontal and vertical position of the bubble along the axes, while a third indicator defines the bubble size. Let's see how you should prioritise work on the sample projects based on their current status. Click on the **Maintainability Level vs. Business Value** quadrant to view the full version:



Maintainability Level vs. Business Value for current iso9126 projects

In this chart, projects with a high business value appear higher, while more maintainable projects appear more to the right. The bigger a bubble, the bigger the number of functions of methods implemented. Therefore, a project with a high business value like Earth (blue) is in a difficult situation because it does not have a high maintainability index. Saturn (yellow), which is comparatively bigger and more complex, is more maintainable but may require a comparable level of attention. As a project manager, you can worry less about Mars (red) and Pluto (green), which despite having very different maintainability indexes, are not critical for the company.

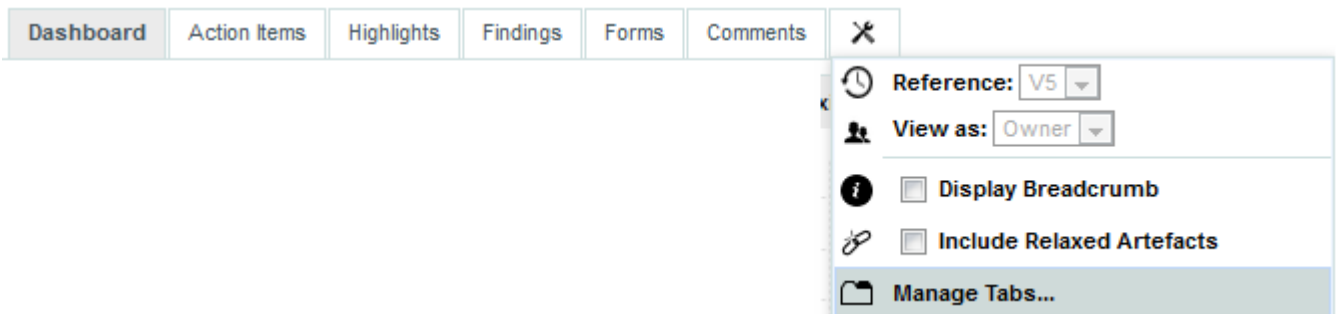
Below the quadrants, Squore displays tables with the values used in the graphs so you can refine the information read in the charts. All the information shown in the analysis model dashboard can be configured by a Squore administrator. Refer to the Squore Configuration Guide for more information.

6. Managing Your To-Do List With Squore

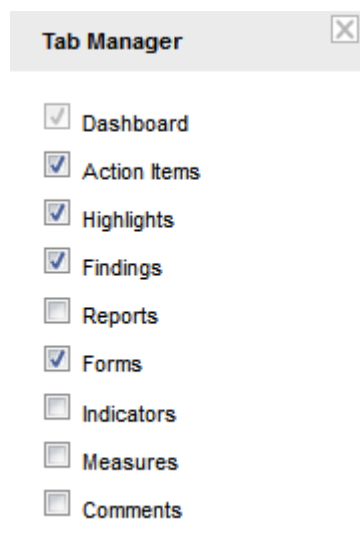
The analysis results you obtained by creating your first projects in Chapter 4, *Creating Projects and Versions* and observed in Chapter 5, *Understanding Analysis Results* can be drilled down further by looking at the other tabs available in the Explorer. In this chapter, you will learn how to use the information contained in Indicators, Measures, Findings and Action Items to better understand and reuse the information provided by Squore in your development workflow.

6.1. How do I understand and Improve My Ratings?

If you need more background information about the measures and indicators used in the charts and tables in the dashboard, the **Indicators**, **Measures** and **Findings** tabs can provide more details about the statistics recorded for the current artefact. Note that these tabs are not displayed by default. If you want to show them in Squore, click the Explorer Settings menu and then Manage Tabs to display the Tab Manager to enable these tabs, as shown below:



The Manage Tabs option in the Explorer Settings allows to display the Tab Manager



The Tab Manager allows to display tabs hidden by default by checking them.
Note that not according to your configuration, some tabs may not be removeable


If you want to understand the scale used for a particular indicator, to see for example how close you are to moving up the rating scale, you can check the scale used for this indicator in the **Indicators** tab of the dashboard.

Log in and search for the artefact **DB5_backup.c** in the Neptune project, where the indicator **Non Compliant Analysability Rules** is rated D. While this tells you about the current rating for this artefact, this does not tell you how to improve it. In order to learn how to improve this score, let's first take a look at the scale used for this indicator. Click the **Indicators** tab of the Explorer. The table of indicators opens, as shown below:

Name	Mnemonic	Value	Rank
Technical Debt Density	DDEBT	398.87	32
Compliant Rules	RKO	4	4
Reference to 'Analysability' Standards	ROKR_ANA	62.5	8
Stability NCC	NCC_TEST	6	4
Changeability NCC	NCC_CHAN	7	4
Analysability NCC	NCC_ANA	7	4
Analysability	ANA	3.67	4
Technical Debt Average	ADEBT	5.88	3
Compliant Testability Rules	RKO_TEST	3	3
Compliant Changeability Rules	RKO_CHAN	3	3
Compliant Analysability Rules	RKO_ANA	3	3
Maintenance Performance	MPI	0	0
Reference to 'Changeability' Standards	ROKR_CHAN	76.92	4
Stability	TEST	1.67	2
Maintainability	MAIN	2.25	2
Changeability	CHAN	2.33	2
Reference to 'Stability' Standards	ROKR_STAB	90.91	2
Reference to 'Testability' Standards	ROKR_TEST	81.25	2
Descriptiveness	SDESCR	23.21	1
Comment Rate	COMR	23.21	1

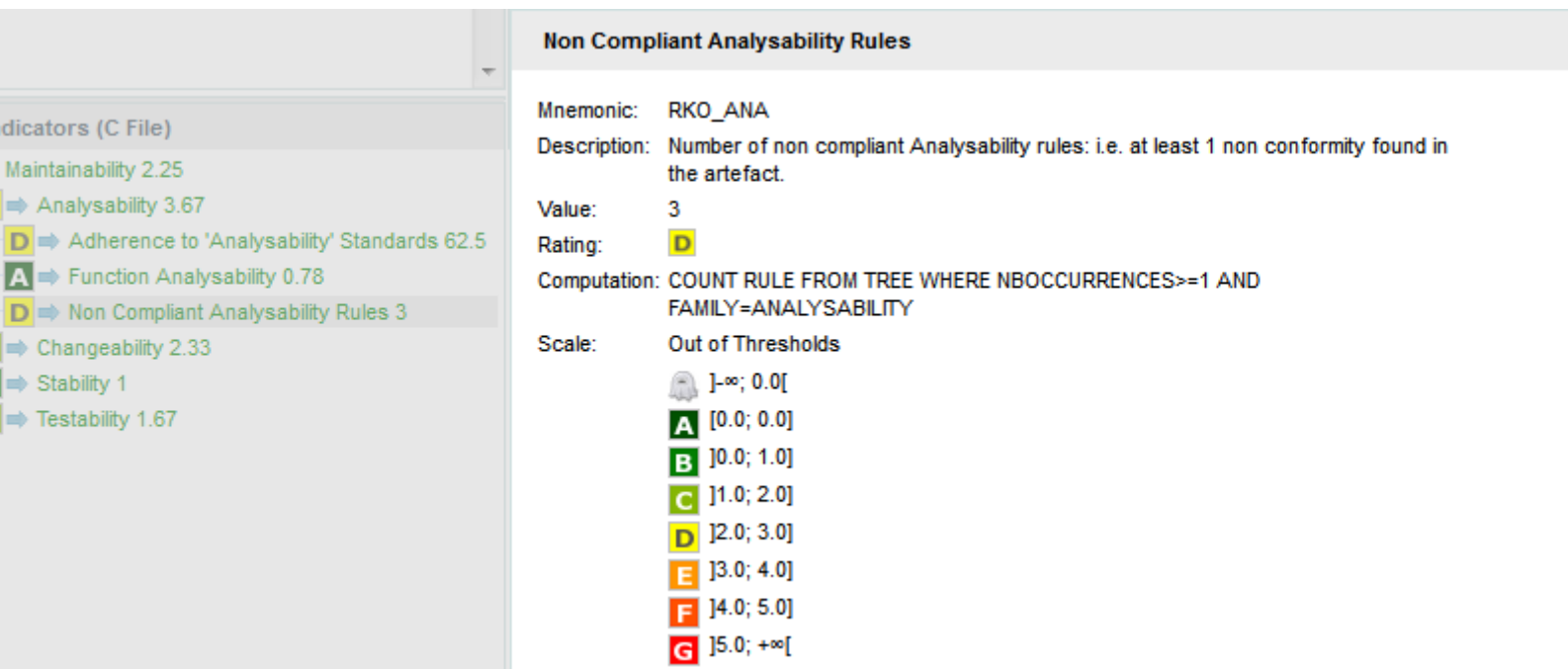
The indicators table for DB5_backup.c

The table lists all the indicators available for the artefact over several pages. The scale and levels available for an indicator can be viewed in a tooltip by placing your mouse over a rating. Using the filter above the "name" column, look for the entry named **Non Compliant Analysability Rules**, then click its value in the rating column. The scale for the indicator indicates that the artefact is rated D because the value of the indicator is 3. In order to improve the score, the value would need to decrease to under 2 to be rated C, as shown below:

Scale: Out of Thresholds	
] -∞; 0.0[
A] 0.0; 0.0]
B] 0.0; 1.0]
C] 1.0; 2.0]
D] 2.0; 3.0]
E] 3.0; 4.0]
F] 4.0; 5.0]
G] 5.0; +∞[

The scale used for rating Non Compliant Analysability Rules

To understand how to improve the rating, you need to know how the indicator's value is computed. Clicking the indicator name in the Indicator Tree shows the following explanation in the indicator popup:



Non Compliant Analysability Rules

Mnemonic: RKO_ANA


Description: Number of non compliant Analysability rules: i.e. at least 1 non conformity found in the artefact.

Value: 3

Rating: **D**

Computation: COUNT RULE FROM TREE WHERE NBOCCURRENCES >= 1 AND FAMILY = ANALYSABILITY

Scale: Out of Thresholds

] -∞; 0.0[
A] 0.0; 0.0]
B] 0.0; 1.0]
C] 1.0; 2.0]
D] 2.0; 3.0]
E] 3.0; 4.0]
F] 4.0; 5.0]
G] 5.0; +∞[

The indicator popup for the Non Compliant Analysability Rules indicator

The computation, i.e. the formula used to calculate the rating is `COUNT RULE FROM TREE WHERE RULE.NBOCCURRENCES >= 1 AND RULE.FAMILY = ANALYSABILITY`, meaning that the indicator is calculated based on the number of other metrics that need to be rules of type analysability broken more than once. To find out what these rules are, click the **Findings** tab.

Squore displays all the findings for a particular artefact in a table in the Findings tab. Next to the finding's label is a number of occurrences followed by a colour-coded delta value (red for more occurrences, green for less) compared to a previous analysis.

If you want to find out which rules are taken into account by the Non Compliant Analysability Rules indicator, click the >> button next to the default filter to show the advanced filtering options. Highlight **ANALYSABILITY** in the characteristics filter to see the corresponding rules, as shown in the picture below:

Dashboard
Action Items
Highlights
Findings
Forms
Indicators
✕

Violations <<

Data Providers:
 All
 SQuORE

Characteristics :
 All
Analysability
 Changeability
 Code Presentation
 Fault Tolerance
 Maturity
 Modifiability

Remediation Cost:
 All
 Heavy
 High
 Medium
 Low
 Little
 Tiny

Severity:
 All
 High
 Major
 Medium
 Low
 Information

Nature:
 All
 Relaxed Finding

Practice	Occ.	Delta	Data Provider	Remediation Cost	Severity	Nature
▶ Use of goto	4	+4	SQuORE	Medium	High	
▶ Multiple exits	1	+1	SQuORE	Low	Low	
▶ No compound if	2	+2	SQuORE	Tiny	Low	

Total: 7; delta: +7

The findings table for DB5_backup.c

The rules **No Compound if**, **Use of Goto** and **Multiple exits** are the rules that should be fixed in order to improve the analysability rating of DB5_backup.c.

You can expand the use of Goto rule to show each occurrence of the rule being broken, and also review the location in the source code that breaks the rule, as shown below:

Occ. ↕	Delta ↕	Data Provider ↕	Remediation Cost ↕	Severity ↕	Nature ↕
4	+4	SQuORE	Medium	High	
te())	4	+4			
1	+1	SQuORE	Low	Low	
2	+2	SQuORE	Tiny	Low	


Rule: Use of goto

Name: Use of goto
Mnemonic: NOGOTO
Description: The 'goto' statement shall not be used (see [1])
Data Provider: SQuORE
Characteristics : Analysability, Changeability, Testability, Structurability

Artefact: hi_scores_write()

Path: DB5_backup.c

Findings:

 Line: 41	Goto are not allowed
 Line: 46	Goto are not allowed
 Line: 52	Goto are not allowed
 Line: 58	Goto are not allowed

The location of the broken occurrences of the use of Goto rule

Finally, clicking on the line number for each rule breaking occurrence opens the source code viewer in full screen so you can carry out your code review:

X
Source Code

C	📄	DB5_backup.c	
B	📄	hi_scores_write()	Compare to: ▼

```

                                /DB5_backup.c
34 int hi_scores_write ()
35     /* update the score file*/
36 {
37     int i,status;
38
39     hi_scores_file = fopen ("hi_score.lst", "w");
40
41     if (hi_scores_file == NULL) goto openerror;
42     {
43         /* writing loop */
44         i = 0;
45     loopwrite:
46         if (i == last_hi_score) goto endloopwrite;
47         {
48             fprintf (hi_scores_file, "%s\n", hi_scores_tab [i].name);
49             fprintf (hi_scores_file, "%s\n", hi_scores_tab [i].firstname);
50             fprintf (hi_scores_file, "%d\n", hi_scores_tab [i].score);
51             i++;
52             goto loopwrite;
53         }
54     endloopwrite:
55         /* close the high score file*/
56         fclose (hi_scores_file);
57         status = 0;
58         goto endwrite;
59     }
60     openerror:
61     {
62         /* error when opening the high score file*/
63         status = 1;
64         format_output("opening problem, file: hi_score.lst\n",1);
65     }
66     endwrite:
67     {
68         return(status);
69     }
70 }
71 }
                    
```

The source code viewer highlighting the first occurrence of No Goto rule breaking

The source code viewer allows comparing the code against another version of the code. Select a version name in the **Compare to:** list to switch to diff mode, as shown below:

Sources

hi_scores_write() W25

```

/DB5_backup.c
34 int hi_scores_write ()
35     /* update the score file*/
36 {
37     int i,status;
38
39     hi_scores_file = fopen ("hi_score.lst", "w");
40
41     if (hi_scores_file == NULL) goto openererror;
42     {
43         /* writing loop */
44         i = 0;
45         loopwrite:
46         if (i == last_hi_score) goto endloopwrite;
47         {
48             fprintf (hi_scores_file, "%s\n", hi_scores_ta
49             fprintf (hi_scores_file, "%s\n", hi_scores_ta
50             fprintf (hi_scores_file, "%d\n", hi_scores_ta
51             i++;
52             goto loopwrite;
53         }
54         endloopwrite:
55         /* close the high score file*/
56         fclose (hi_scores_file);
57         status = 0;
58         goto endwrite;
59     }
60     openererror:
61     {
62         /* error when opening the high score file*/
63         status = 1;
64         format_output("opening problem, file: hi_score.
65     }
66     endwrite:
67     {
68         return(status);
69     }
70 }
            
```

hi_scores_write() Current

```

/DB5_backup.c
34 int hi_scores_write ()
35     /* update the score file*/
36 {
37     int i,status;
38
39     hi_scores_file = fopen ("hi_score.lst", "w"
40
41     {
42         /* writing loop */
43         i = 0;
44         loopwrite:
45         if (i == last_hi_score) goto endloopwrite
46         {
47             fprintf (hi_scores_file, "%s\n", hi_sco
48             fprintf (hi_scores_file, "%s\n", hi_sco
49             fprintf (hi_scores_file, "%d\n", hi_sco
50             i++;
51             goto loopwrite;
52         }
53         endloopwrite:
54         /* close the high score file*/
55         fclose (hi_scores_file);
56         status = 0;
57         goto endwrite;
58     }
59     openererror:
60     {
61         /* error when opening the high score file
62         status = 1;
63         format_output("opening problem, file: hi_
64     }
65     endwrite:
66     {
67         return(status);
68     }
69 }
            
```

The source code viewer in diff mode

Tip

In diff mode, use the top arrows to switch the left and right panes, and the bottom arrows to turn synchronised scrolling on or off. Characters that were removed are underlined in green, while characters that were added are underlined in red.

Analysing findings helps improving the quality of the code in your project. There is much more you can do with the Findings tab by using the built-in filters to detect regressions and improvements:

- **Violations:** displays all the rules violated in this version
- **Lost Practices:** displays violations that are new in this version since a specified version
- **Acquired Practices:** displays all violations not occurring anymore in this version since a previous version
- **Deteriorated Practices:** displays all violations with more occurrences in this version than in a previous version
- **Improved Practices:** displays all violations with less occurrences in this version than in a previous version

- **All changed practices:** displays all the rules where a change in the number of violations was detected, essentially providing the combination of Improved Practices and Deteriorated Practices in one list
- **All rules:** displays all the rules checked by the model, i.e. the violated ones as well as the ones that are not

Tip

By default, the Findings tab displays violations compared to the previous analysis, but you can refine the search by adjusting the **Reference** drop-down list (under the Explorer Settings menu) that contains all the versions analysed in your project.

You can learn about more automated ways to review and fix code in Section 6.6, "How Do I Review And Manage Action Items Flagged by Squore?".

6.2. Relaxing Violations in Code

Squore provides a violation relaxation mechanism that is triggered via comments found in the source code itself. There are two pre-requisites for relaxation to work:

- The model used to analyse your source code must implement a special rule called **R_RELAX** for relaxation to take place.
- You need to know the mnemonic of the violated rule you want to relax, in order to use it as a key in your comment.

Squore interprets comments formatted in one of these three ways:

1. Inline Relaxation

This syntax is used to relax violations on the current line.

```
some code; /* %RELAX<keys> : Text to justify the relaxation */
```

2. Relax Next Line

This syntax is used to relax a violation on the first following line that is not a comment. In the example the text of the justification will be: "Text to justify the relaxation the text of the justification continues while lines are made of comments only"

```
/* >RELAX<keys> : Text to justify the relaxation */  
/* the text of the justification continues while */  
/* lines are made of comments only */  
some code;
```

3. Block Relaxation

This syntax is used to relax violations in an entire block of code.

```
/* {{ RELAX<keys> : Text to justify the relaxation */  
/* like for format 2 text can be on more than one line */  
int my_func() {  
    /* contains many violations */  
    ...  
}  
/* }} RELAX<keys> */
```

<keys> can be one of the following:

- **<*>**: relax all violations
- **<MNEMO>**: relax violations of the rule MNEMO
- **<MNEMO1,MNEMO2,...,MNEMOn>**: relax violations of rules MNEMO1 and MNEMO2 ... and MNEMOn

The relaxed violations are still shown in the Findings page after the next analysis, but they appear under the rule R_RELAX, showing the mnemonic of the relaxed violation and the justification text.


As an example, this is how you would relax the violations of the rule Backward goto for Non Compliant Analysability Rules in Neptune:

1. click the violation of **Backward goto** on the Findings page to find the rule's mnemonic (BWGOTO) and the location of the finding (DB5_backup.c line 52).

Rule: Backward goto

Name: Backward goto
Mnemonic: BWGOTO
Description: Backward gotos shall not be used.
Data Provider: SquORE
Characteristics : Stability, Changeability, Testability, Structured Programming

Artefact: hi_scores_write()

Path: DB5_backup.c
Findings:  **Line: 52** Backward Goto are not allowed (goto loopwrite)

The details of the Backward goto violation

2. Edit the code of the sample project to relax the violation as shown below.

```
goto loopwrite; /* %RELAX<BWGOTO> : This backward goto is acceptable in our code.
```

3. Create a new version of the project.

On the Findings page, the violation now listed under the **Relaxed Violation** category, and your justification comment appears in the details pane of the finding:

Dashboard
Action Items x
Highlights x
Findings x
Forms x
Indicators x
x

Reference: V1 V1 i

Violations v >>

Practice ▲	Occ. ⇅	Delta ⇅	Data Provider ⇅	Remediation Cost ⇅	Severity ⇅	Nature ⇅
▶ Assignment in Boolean	4	0	SQuORE	Little	Low	
▶ Multiple exits	2	0	SQuORE	Low	Low	
▶ No compound if	32	0	SQuORE	Tiny	Low	
▶ No compound statement	22	0	SQuORE	Tiny	Low	
▼ Relaxed violation	1	+1	SQuORE	None	Information	Relaxed Finding
▼ Relaxed violation	1	+1	SQuORE	None	Information	Relaxed Finding
▶ hi_scores_write()	1	+1				

Rule: Relaxed violation

Name: Relaxed violation
Mnemonic: RELAX
Description: A rule violation is relaxed and justified.
Data Provider: SQuORE
Characteristics:

Artefact: hi_scores_write()

Path: DB5_backup.c
Findings:

Line:
52

Rule BWGOTO relaxed: : This backward goto is acceptable in our code.

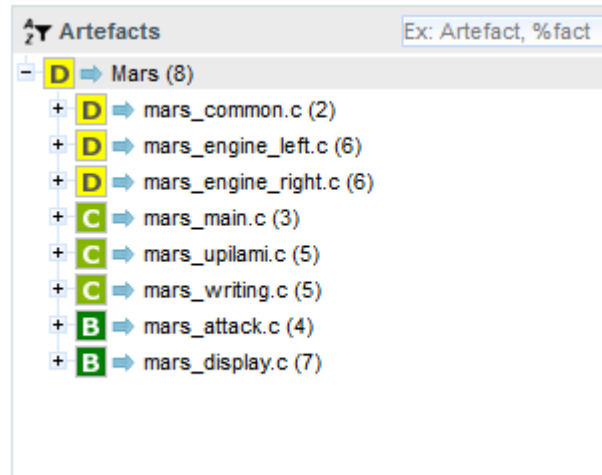
The relaxed violation

6.3. Relaxing Artefacts

In this section, you will learn how to relax artefacts directly from the Artefact Tree instead of relaxing violations by editing the source code of the application. Relaxing artefacts ensures that their metrics do not impact the rating of the project, however, data providers will still generate findings for the relaxed artefacts.

This example uses the Mars project from the samples folder. Ensure that you are a Project Manager in this project, or are part of a role with the **View Drafts of Projects** and **Modify Artefacts** privileges before you begin.

The following is an overview of the artefacts in the Mars project as created by the demo script:



The artefacts in the Current version of the Mars project and their rating

To relax an artefact and therefore tell Squore that its rating should not impact the rest of the project, display the context menu for this artefact. The relaxation options appear at the bottom of the menu if they are available for your model, as shown below:



The artefact context menu

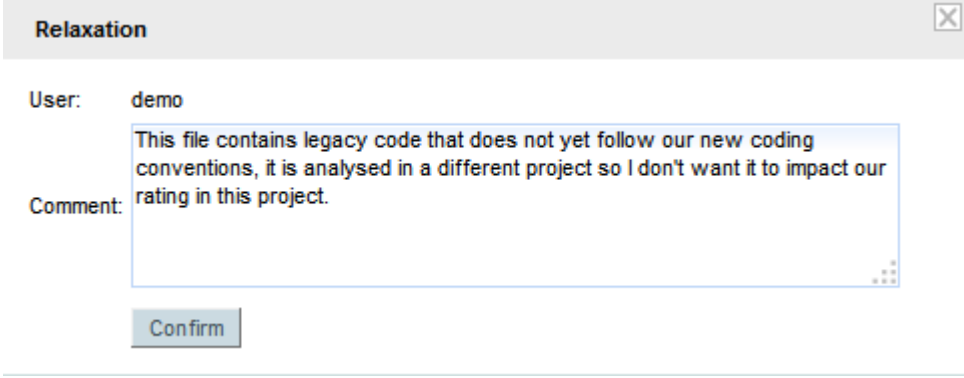
There are two actions that can be taken to relax an artefact:

- **Relax...** allows simply marking an artefact as relaxed, leaves it in the tree in a way that will not impact the overall rating of the project.
- **Exclude...** also relaxes the artefact but then removes it from the Artefact Tree so it will not be visible anymore in future analyses.

Tip

In both cases, the relaxation action is only made on a draft version and can be reversed by selecting the **Un-relax...** entry in the menu or the **Clear unapplied changes** option in the project portfolio.

Clicking **Relax...** or **Exclude...** brings up a pop-up menu where you can type a comment to explain the reason for the relaxation. Let's relax `mars_common.c` so it stops impacting the overall project rating. Click the **Relax...** option in the menu to display the relaxation popup and enter a relaxation comment:



The image shows a 'Relaxation' dialog box with a close button (X) in the top right corner. It contains the following text:

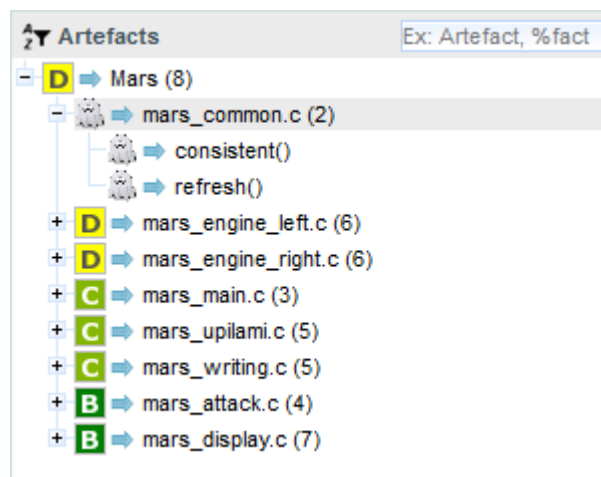
User: demo

Comment: This file contains legacy code that does not yet follow our new coding conventions, it is analysed in a different project so I don't want it to impact our rating in this project.

Confirm

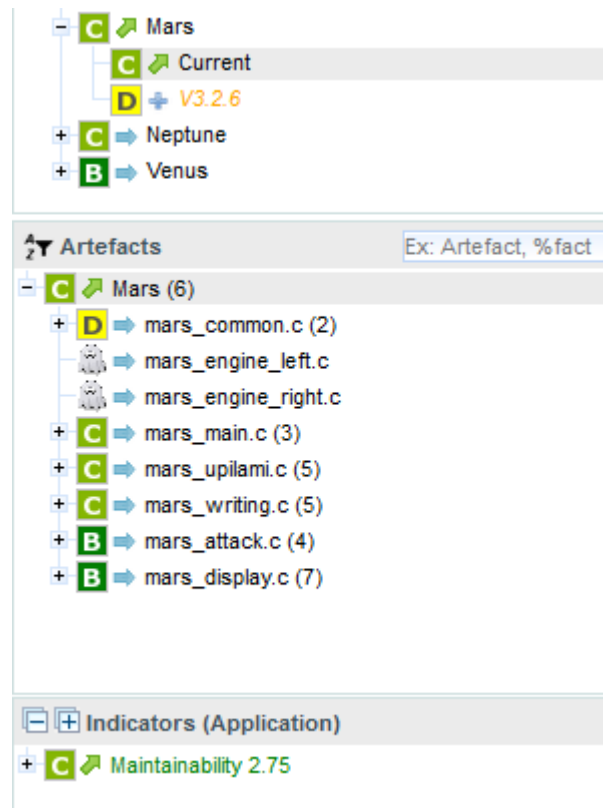
The relaxation justification

Click **Confirm** to save your comment, and notice how the Artefact Tree is updated to reflect the relaxation state:



The relaxed `mars_common.c` in the Artefact Tree

If you keep relaxing artefacts rated D in this project and create a new build of the project, then you will end up seeing changes in the overall rating, as shown below:

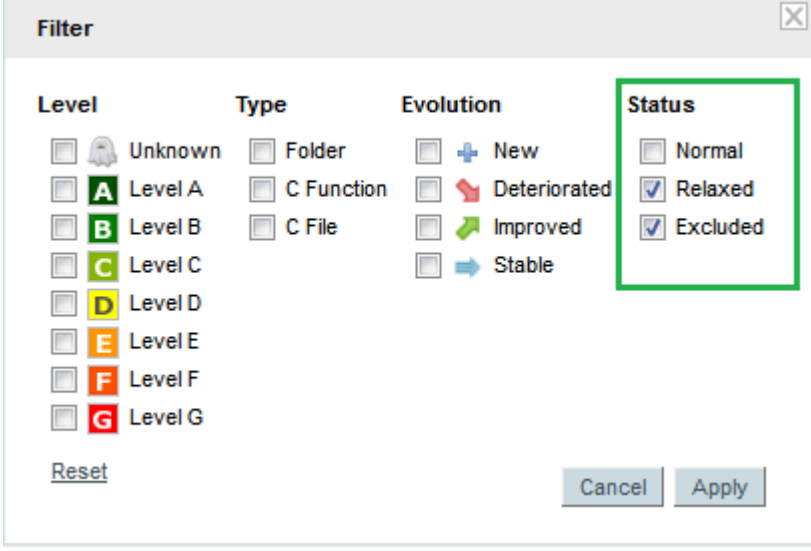


The improved rating of the Mars after a new analysis

Tip

When you relax an artefact, the action items and findings relevant to this artefact are hidden, except when you specifically click on the relaxed artefact. If you want to show them, you can do so by clicking the Include Relaxed Artefacts option from the Explorer Settings menu.

You can show or hide relaxed and excluded artefacts by checking the boxes with the appropriate status in the filter popup:



Level	Type	Evolution	Status
<input type="checkbox"/> Unknown	<input type="checkbox"/> Folder	<input type="checkbox"/> + New	<input type="checkbox"/> Normal
<input type="checkbox"/> A Level A	<input type="checkbox"/> C Function	<input type="checkbox"/> D Deteriorated	<input checked="" type="checkbox"/> Relaxed
<input type="checkbox"/> B Level B	<input type="checkbox"/> C File	<input type="checkbox"/> I Improved	<input checked="" type="checkbox"/> Excluded
<input type="checkbox"/> C Level C		<input type="checkbox"/> S Stable	
<input type="checkbox"/> D Level D			
<input type="checkbox"/> E Level E			
<input type="checkbox"/> F Level F			
<input type="checkbox"/> G Level G			

The improved rating of the Mars after a new analysis

6.4. Working with Forms and Checklists

Squore lets you view and edit project attributes in a dedicated form tab of the explorer. You can therefore design your wizards to present checklists to a user. They can fill in the values manually after an analysis and they will be taken into account when creating the next version of the project. There are permissions associated with editing form values, so you can make them read-only for guests but read-write for project managers. The attributes displayed on the Forms tab depend on the type of the current artefact, and values are saved individually for each artefact in the project.

Dashboard	Action Items	Highlights	Findings	Reports	Forms	X
-----------	--------------	------------	----------	---------	-------	---

Project Information

Project Team size:	<input type="text" value="5"/> Men	Click to comment
Software version date:	<input type="text" value="2014/09/09"/>	Click to comment
Unified Project End Date:	<input type="text" value="2014/09/09"/>	Click to comment
Estimated Delay:	<input type="text" value="0.0"/>	Click to comment
Productivity:	<input type="text" value="0.0"/>	Click to comment
Implemented Requirements	<input type="text" value="1.0"/>	Click to comment
Planned Requirements	<input type="text" value="1.0"/>	Click to comment
Code dedicated to new features	<input type="text" value="50.0"/> %	Click to comment
Code dedicated to maintenance	<input type="text" value="50.0"/> %	Click to comment

Application attributes

Project Business Value:	<input type="text" value="50.0"/> FP	Click to comment
Project Cost:	<input type="text" value="0.0"/> M/M	Click to comment

Remediation Cost

An example form

Note: To begin working with forms, make sure the Forms tab is visible in the Explorer.

When you click a project in the Project Portfolios and view the Forms tab of the Explorer, all the project attributes available at application level are displayed, as shown below:

Dashboard	Action Items	Highlights	Findings	Reports	Forms	X
-----------	--------------	------------	----------	---------	-------	---

Application attributes

Project Business Value:	<input type="text" value="80.0"/> FP	Click to comment	▶
Project Cost:	<input type="text" value="60.0"/> M/M	Click to comment	▶
Project Safety Integrity Level:	<input checked="" type="radio"/> SIL0 <input type="radio"/> SIL1 <input type="radio"/> SIL2 <input type="radio"/> SIL3 <input type="radio"/> SIL4	Click to comment	▶

The Forms tab for the Earth project

The values displayed correspond to the application attributes passed when the last version of Earth was created. Users with the whose role grants them the Modify Artefacts Attributes privilege can edit the current value of the form for any artefact, and the value will be taken into account during the next analysis.

When you modify the values in the form, you can use the comment field to justify the change you made. A history of the modifications can be displayed by expanding the attribute field, as shown below

Dashboard
Action Items
Highlights
Findings
Reports
Forms
Indicators
Measures
Comments
✕

▼ **Application attributes**

Project Business Value: FP updated after we revised our strategy recently ▼

Date Modified	Version	Username	Value	Comments
Sep 8, 2014 4:27:32 PM	V1	demo	80.0 FP	
Sep 9, 2014 4:33:01 PM	Current	demo	120.0 FP	
Sep 9, 2014 4:33:24 PM	Current	demo	120.0 FP	revised our strategy recently
Sep 9, 2014 4:33:42 PM	Current	demo	120.0 FP	updated after we revised our strategy recently

Project Cost: M/M Click to comment ▶

Project Safety Integrity Level: SIL0 SIL1 SIL2 SIL3 SIL4 Click to comment ▶

A history of modifications for the **Project Business Value** attribute

6.5. What Does This Measure Mean Exactly?

If you have doubts about the measures computed by Squore and their meaning, they can usually be solved by looking at the **Measures** tab of the Explorer. The content of the measures tab is also always refreshed to reflect the data for the current artefact, and is organised in a table displaying the measure's mnemonic, full name, description and value for the current selection, as shown in the picture below.

Name	Mnemonic	Description	Value
Rated A Functions	A_FUNC	Number of rated A functions	11
Rated A Functions with ELOC > 5	A_FUNC_5	Number of rated A functions with a number of line of code > 5	8
Statements in A Functions	A_STAT	Number of executable statements in rated A functions	186
Code Cloning Ratio	C_LOR	Percentage of function control flow tokens cloned in the artefact	15.72
Analyzability	ANA	Level of 'Analyzability'	6.87
Average Cyclomatic Complexity	AVGVC	Average Cyclomatic Complexities of the functions defined in the source file(s).	9.06
Rated B Functions	B_FUNC	Number of rated B functions	27
Rated B Functions with ELOC > 5	B_FUNC_5	Number of rated B functions with a number of line of code > 5	19
Statements in B Functions	B_STAT	Number of executable statements in rated B functions	453
Rated C Functions	C_FUNC	Number of rated C functions	7
Rated C Functions with ELOC > 5	C_FUNC_5	Number of rated C functions with a number of line of code > 5	6
Statements in C Functions	C_STAT	Number of executable statements in rated C functions	307
Control Flow Token	CFT	Total number of tokens in the control flow of the function(s)	766
Control Flow Token Cloned	CFTC	Total number of cloned tokens in the control flow of the function(s)	250
Changeability	CHAN	Level of 'Changeability'	6.33
Code Cloning Ratio	C_LOR	Percentage of function control flow tokens cloned in the artefact	31.45
Comment Rate	COMR	Ratio between the number of lines of comments and the number of source lines of code.	25.81
Comment Rate	COMR	Ratio between the number of lines of comments and the number of effective source lines of code.	31.89
Project Complexity Variation	COMPLX_TREND	Variation of the project complexity (e.g. using CyclomaticComplexity) since the previous version	35.22
Rated D Functions	D_FUNC	Number of rated D functions	3

The table of measures for the DB5_backup.c

Measures can be sorted by mnemonic, name, description or value, and the sorting value is remembered when selecting another artefact in the tree so you can easily compare values.

6.6. How Do I Review And Manage Action Items Flagged by Squore?

Searching for issues in your applications can be a manual process, as explained in Section 6.1, “How do I understand and Improve My Ratings?”, but the analysis and decision models configured within Squore can automate this process by automatically suggesting items that require your attention after analysing the latest version of your code. This functionality can be accessed as part of the Explorer, in the **Action Items** tab. In this section, you will learn how to review Squore's suggestions and incorporate them into your own issue management tool.

Note that in order to change the status of action item, you must be working with the current draft version of a project. In order to follow the steps below, ensure that you select the current version of the Earth project, click on the Action Items tab. A list of action items suggested by Squore appears in a table, as shown in the picture below:

Dashboard	Action Items	Highlights	Findings	Reports	Forms	Indicators	Measures	Comments	X
Id: <input type="text"/>		Type: <input type="text"/>		>>					
<input checked="" type="checkbox"/>	Id	Type	Since	Action Type	Priority	Scope	Status	Comments	
<input checked="" type="checkbox"/>	282	Poor code layout	V1	Code Review	Low	C File	Open		
<input checked="" type="checkbox"/>	287	Hardly testable function	V4	Code Review	Medium	C Function	Open		
<input checked="" type="checkbox"/>	281	Poor code layout	V1	Code Review	Low	C File	Open		
<input checked="" type="checkbox"/>	291	Potential cloned code	Current	Code Review	High	C Function	Open		
<input checked="" type="checkbox"/>	292	Hardly testable function	Current	Code Review	Medium	C Function	Open		
<input checked="" type="checkbox"/>	286	Poor code layout	V4	Code Review	Low	C File	Open		
<input checked="" type="checkbox"/>	284	Poor code layout	V3	Code Review	Low	C File	Open		
<input checked="" type="checkbox"/>	297	Poor code layout	Current	Code Review	Low	C File	Open		
<input checked="" type="checkbox"/>	293	Deteriotated to F.	Current	Non Regression	Medium	C Function	Open		
<input checked="" type="checkbox"/>	296	Potential missing break in 'switch' case	Current	Debt Management	High	C Function	Open		
<input checked="" type="checkbox"/>	298	Potential missing break in 'switch' case	Current	Debt Management	High	C Function	Open		
Total: 11									
Add Artefacts to Review Set		ClearQuest <input type="button" value="v"/>		Export Selected Items					

The action items table for the current version of the Earth project

You can filter action items if needed by using the filters above the table. The name given by Squore to the action item is the name defined in your analysis model for this alert. Priorities are also predefined, and your input is needed to validate or invalidate the reports based on your priorities.

In the action items list, 291, 296 and 298 are high priority, therefore their status should be changed to **Todo**.

If you are unsure about a report, you can click the action item ID to display the full details, which includes the location(s) in the source code that triggered the alarm:

<input checked="" type="checkbox"/>	296	Potential missing break in 'switch' case	Current	Debt Management	High	C Function	Open	
Potential missing 'break' statement in 'switch' statement in the function <code>player_plays()</code> in file <code>apps/player.c</code> . Check first if falling through the next case is intentional. If not, add the missing break else document the code to make explicit the purpose and relax the rule.								
Artefact: <code>player_plays()</code> Path: <code>apps/player.c</code>								
• Falthrough (1) <code>player_plays()</code>								
Detailed View - 1 reasons								
<input checked="" type="checkbox"/>	298	Potential missing break in 'switch' case	Current	Debt Management	High	C Function	Open	

Action Item details for 296

You can review the code in a popup window before you decide to fix or relax the action item.









Finally, you can export the action items generated by Squore and feed them into your own issue tracker: select the export format you want to use (CSV, ClearQuest, Mantis, XML, or any other custom format you defined in your configuration) and click the **Export** button to download the list to a file. You can also add all artefacts that triggered an action item to your Review Set by clicking the appropriate button.

Tip

If you are looking for a way to present action items instead of exporting them, you should look into Squore reporting functionality, described in Section 8.3, "Reporting Project Status"

6.7. Can I Perform Advanced Data Mining?

The Capitalisation Base provides statistics aggregates, distribution graphics and correlation coefficients across a portfolio of projects. To begin using the Capitalisation Base to understand historical trends about your projects and find out if your analysis models are suited to your development style, click the **Capitalisation Base** menu item in the Squore toolbar.

Portfolio						
Statistics Aggregates		Distribution	Correlation			
<input checked="" type="checkbox"/>	Name	Version	Rating	Analysis Model	Color	Owner
<input checked="" type="checkbox"/>	Earth	V6	E	C Code ISO-9126 Maintainability Level		demo
<input checked="" type="checkbox"/>	Mars	V3.2.6	D	C Code ISO-9126 Maintainability Level		demo
<input checked="" type="checkbox"/>	Mercury	V2010B	E	C Code ISO-9126 Maintainability Level		demo
<input checked="" type="checkbox"/>	Neptune	V2	C	C Code ISO-9126 Maintainability Level		demo
<input checked="" type="checkbox"/>	Pluto	R9	C	C Code ISO-9126 Maintainability Level		demo
<input checked="" type="checkbox"/>	Saturn	Prel	D	C Code ISO-9126 Maintainability Level		demo
<input checked="" type="checkbox"/>	Uranus	B625	D	C Code ISO-9126 Maintainability Level		demo
<input checked="" type="checkbox"/>	Venus	Beta	B	C Code ISO-9126 Maintainability Level		demo

The Capitalisation Base Projects tab

In the projects tab, choose the projects that will be used to aggregate statistics. In the example below, we will look at statistics for the Earth and Mars projects, which both use the same analysis model and have similar overall ratings. Select Earth and Mars from the list and click the **Statistics Aggregates**.

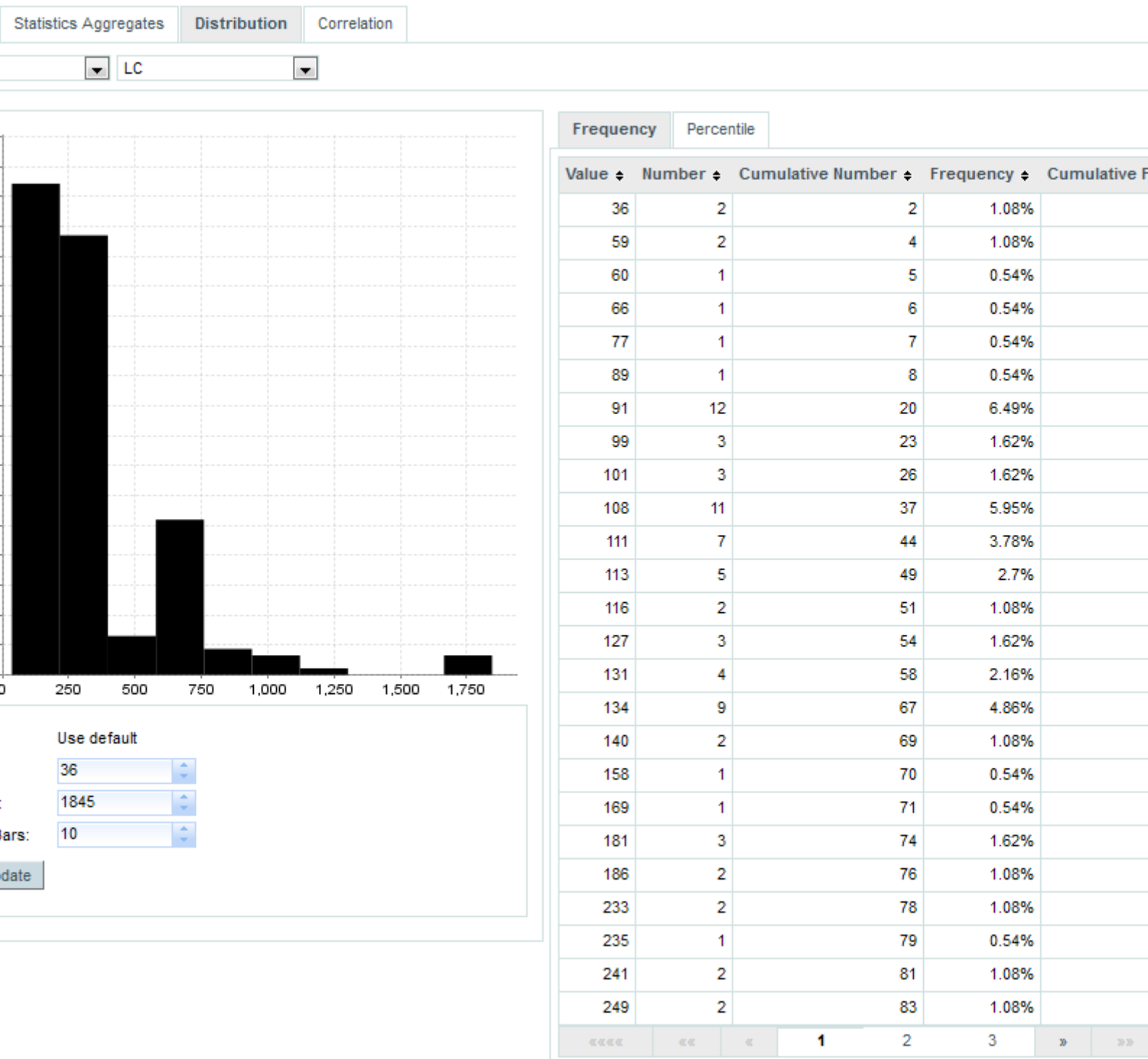
The Statistics Aggregates tab offers an overview of all your projects' data by providing minimum, maximum, average, number of occurrences, deviation, mod, sum and median results. Results are based on all the measures of an artefact type. This means that you have to specify an artefact type before any data is displayed.

Portfolio								
Statistics Aggregates		Distribution	Correlation					
APPLICATION <input type="text"/>								
Measure	Min	Max	Occ.	Avg.	Dev.	Sum.	Med.	Mod.
A_FUNC	5	119	8	33.88	40.17	271	11	11 (4: 0%)
A_FUNC_GREATER_5	3	14	8	7.5	3.28	60	8	8 (2: 0%)
A_STAT	43	290	8	167.62	76.85	1,341	188.5	43 (1: 0%)
ACLOR	0	37.36	8	14.76	15.06	118.06	8.81	0 (2: 0%)
ANALYSABILITY	2.67	8	8	5.75	1.74	46	5.83	8 (2: 0%)
AVGVG	4.29	10.55	8	6.79	2.05	54.32	6.67	4.29 (1: 0%)
B_FUNC	12	257	8	67.5	82.83	540	23	19 (2: 0%)

The Capitalisation Base Statistics Aggregates at Application level

The Distribution tab offers the possibility to display any kind of distribution. The distribution is based on a measure of an artifact type. As a result, you have to select both an **Artefact type** and a **Measure** before you see any results. Note that you can change the parameters of the distribution graph by adjusting the number

of bars, and the minimum and maximum values for the axes. The picture below shows the distribution of lines of code (a measure called LC) across all artefacts of type FILE for Mars and Earth.



The Capitalisation Base Distribution Graph for lines of code per file

The Correlation tab displays the matrix of correlation of any data stored in the Squore database. Correlations are computed between artefacts of the same type, so you have to select the artefact type before any data is displayed. Squore highlights cells in the table in which correlations are above the threshold defined by moving the slider or entering a correlation coefficient directly in the text box provided. You can choose to include derived data by checking the box above the matrix table.

Portfolio
Statistics Aggregates
Distribution
Correlation

FILE
0
1
0.8
Include Derived Measures

Occ: 119

	BLAN	BRAC	CLOC	DOPD	DOPT	FUNC	LC	MLOC	SLOC	STAT	TOPD	TOPT	TXADD	TXMOD	TXREM
BLAN		0.87	0.75	0.90	0.73	0.83	0.92	0.25	0.89	0.88	0.87	0.87	N/C	0.04	N/C
BRAC	0.87		0.76	0.86	0.76	0.80	0.98	0.32	0.98	0.97	0.95	0.96	N/C	0.06	N/C
CLOC	0.75	0.76		0.73	0.61	0.59	0.80	0.58	0.74	0.73	0.70	0.71	N/C	0.07	N/C
DOPD	0.90	0.86	0.73		0.85	0.85	0.90	0.30	0.88	0.86	0.84	0.84	N/C	0.07	N/C
DOPT	0.73	0.76	0.61	0.85		0.74	0.76	0.22	0.75	0.73	0.72	0.72	N/C	0.11	N/C
FUNC	0.83	0.80	0.59	0.85	0.74		0.81	0.09	0.80	0.77	0.78	0.77	N/C	0.06	N/C
LC	0.92	0.98	0.80	0.90	0.76	0.81		0.33	0.99	0.98	0.97	0.98	N/C	0.07	N/C
MLOC	0.25	0.32	0.58	0.30	0.22	0.09	0.33		0.32	0.33	0.30	0.31	N/C	0.05	N/C
SLOC	0.89	0.98	0.74	0.88	0.75	0.80	0.99	0.32		0.99	0.98	0.99	N/C	0.07	N/C
STAT	0.88	0.97	0.73	0.86	0.73	0.77	0.98	0.33	0.99		0.97	0.98	N/C	0.07	N/C
TOPD	0.87	0.95	0.70	0.84	0.72	0.78	0.97	0.30	0.98	0.97		1.00	N/C	0.07	N/C
TOPT	0.87	0.96	0.71	0.84	0.72	0.77	0.98	0.31	0.99	0.98	1.00		N/C	0.07	N/C
TXADD	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C		N/C	N/C
TXMOD	0.04	0.06	0.07	0.07	0.11	0.06	0.07	0.05	0.07	0.07	0.07	0.07	N/C		N/C
TXREM	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	N/C	

Export

The Capitalisation Base correlation table for files measures with a highlighting threshold of 0.8

Tip

Base measures are the ones directly reported by various tools included in the analysis. Derived measures are metrics computed based on these base measures or other derived measures. You can choose to export the results of the correlation matrix to a CSV file. The resulting CSV file contains all metrics pairs for which a correlation exists ((new in 15-A-SP2)).

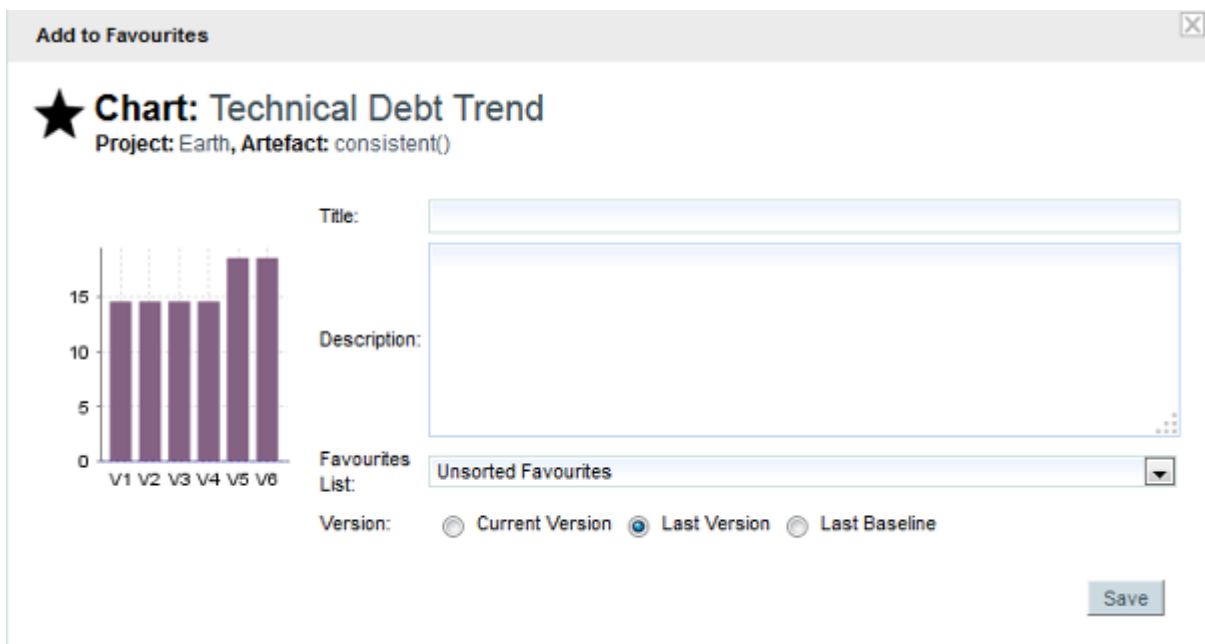
7. Track Your Favourite Indicators

By clicking on My Favourites in the main menu bar, you can view all the charts you marked as favourite in the dashboards across all of your projects. You can group charts into lists and reorder them as you see fit. The charts you mark as favourites are also the ones that are accessible to view on your mobile devices when away from your desk. This section covers everything you need to know about favourites and Squore's mobile interface: Squore Mobile.

7.1. Building a cross-project Dashboard in My Favourites

Each of the chart thumbnails has a star (★) icon that you can click to mark a chart as favourite.

Clicking a star icon brings up the **Add to Favourites** popup, as shown below:



The screenshot shows a modal window titled "Add to Favourites" with a close button in the top right corner. Inside the modal, there is a star icon followed by the text "Chart: Technical Debt Trend" and "Project: Earth, Artefact: consistent()". To the left is a bar chart with six bars labeled V1 through V6, with a y-axis ranging from 0 to 15. To the right of the chart are three input fields: "Title:" (empty), "Description:" (empty), and "Favourites List:" (a dropdown menu currently showing "Unsorted Favourites"). Below these fields are three radio buttons for "Version:" with options "Current Version", "Last Version" (which is selected), and "Last Baseline". A "Save" button is located at the bottom right of the modal.

The **Add to Favourites** popup



The popup allows you to:

- Type a custom title and description for your chart so that you can for example write down why you are monitoring it.
- Select a list of favourites to add the chart to. By default, your charts are added to a list called Unsorted Favourites. You can create more lists and move charts between lists from the My Favourites page.
- Select a version of the chart to display. The latest version is selected by default (**Last Version**), but you can alternatively select the exact version you clicked on (**Current Version**), or the latest baseline (**Last Baseline**).

When you are satisfied with your choices, click on **Save** to add the chart to your favourites. You can add charts from any project you have access to.

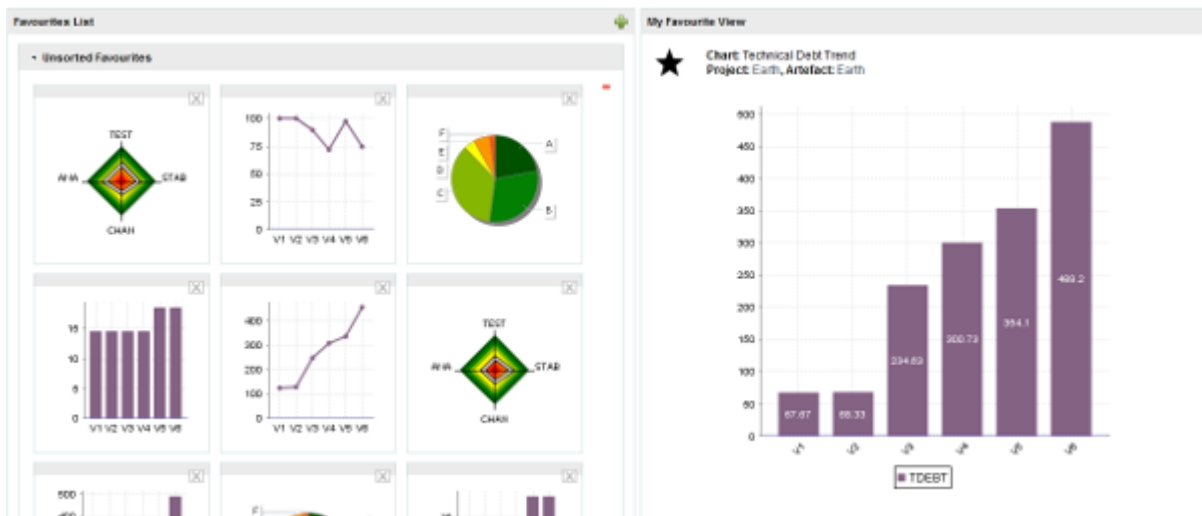
Refer to the next section to learn how to view and manage the charts you saved as favourites.

7.2. Managing Favourites

All the charts you added from the Explorer were added to a list called Unsorted Favourites. You can delete this list and create other lists using the  and  icons.

When you have more than one list, you can drag and drop charts between lists.

In order to see the full size of a chart you marked as favourite, click its thumbnail on the left pane to open it in the right pane. The screenshot below shows an example of a list of favourites and a maximised chart. Note that the right pane contains links that allow you to go back to the project's or artefact's dashboard directly.

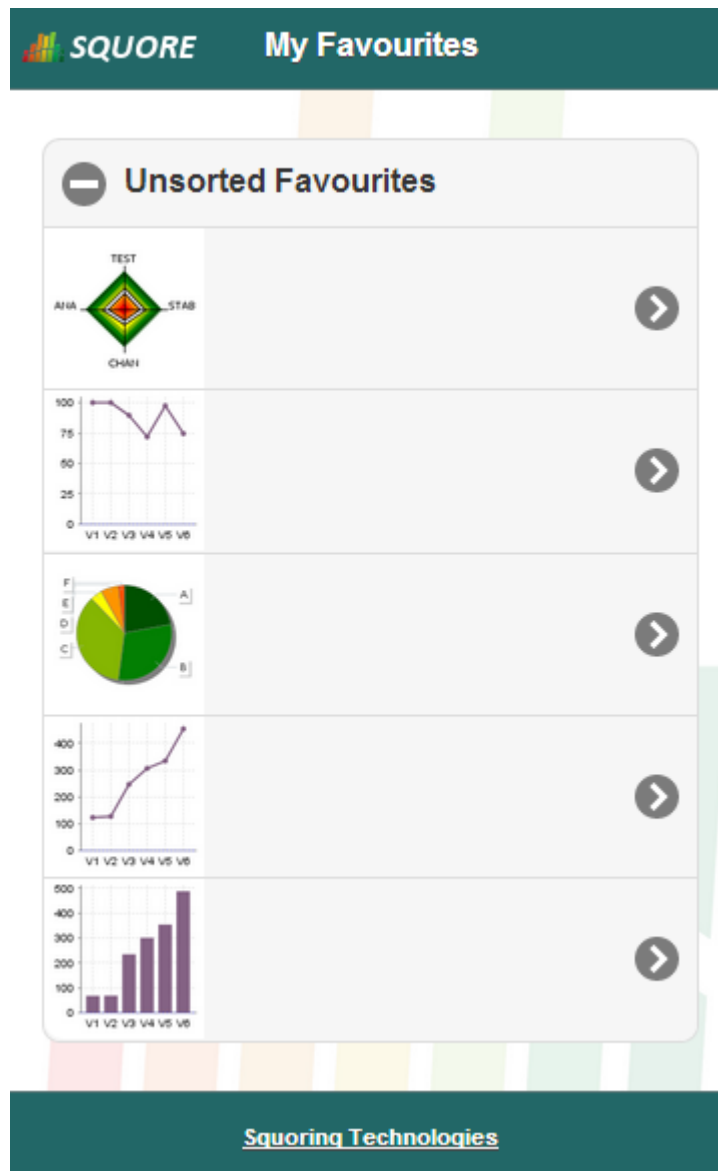


The full My Favourites page

7.3. Squore Mobile

The list of charts you marked as favourites in Squore is the list of charts you can access via Squore Mobile

Squore Mobile is a touch-friendly interface for Squore that is accessible from http://localhost:8180/SquORE_Server/Mobile.



Squore Mobile

When you log into Squore Mobile, you can swipe through all the charts you added to your favourite lists from your mobile device.

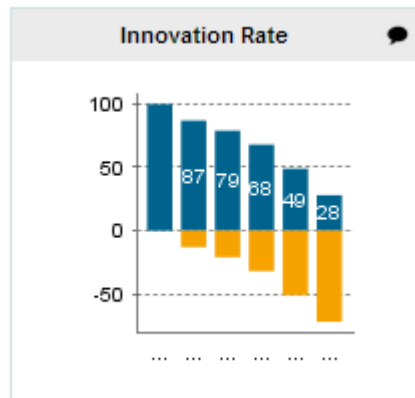
8. Communicating With Squore

8.1. Comments and Notifications

Squore allows posting comments about charts, artefacts, and action items. Users in a project team can view and reply to comments when they notice that a discussion thread has received new posts since their last visit. You can also choose when a discussion no longer accepts comments or is removed from the project. In this section, you will learn the basics of commenting all around the dashboard.

8.1.1. Commenting Charts

Every chart on your dashboard shows a speech bubble icon next to its title, as shown in the picture below:



A chart thumbnail showing a speech bubble icon

Clicking the icon brings up the comment dialog, in which you can confirm which chart you are commenting on and type your comment.

✕
Comments

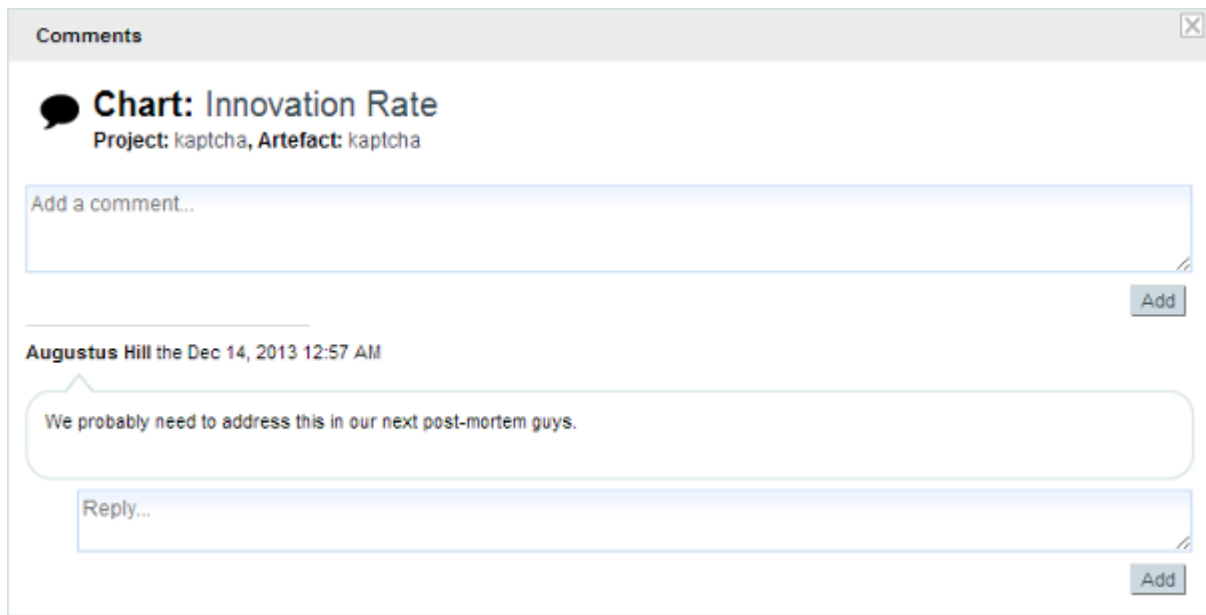
🗨️
Chart: Innovation Rate
Project: kaptcha, Artefact: kaptcha

We probably need to address this in our next post-mortem guys.

Add

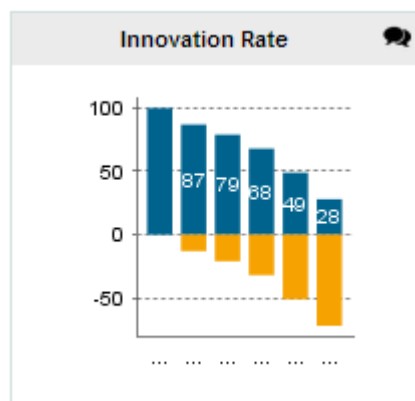
The Comment pop-up

When you click **Add**, your comment is saved, and you can reply or add another comment.



The Comment pop-up after adding your first comment

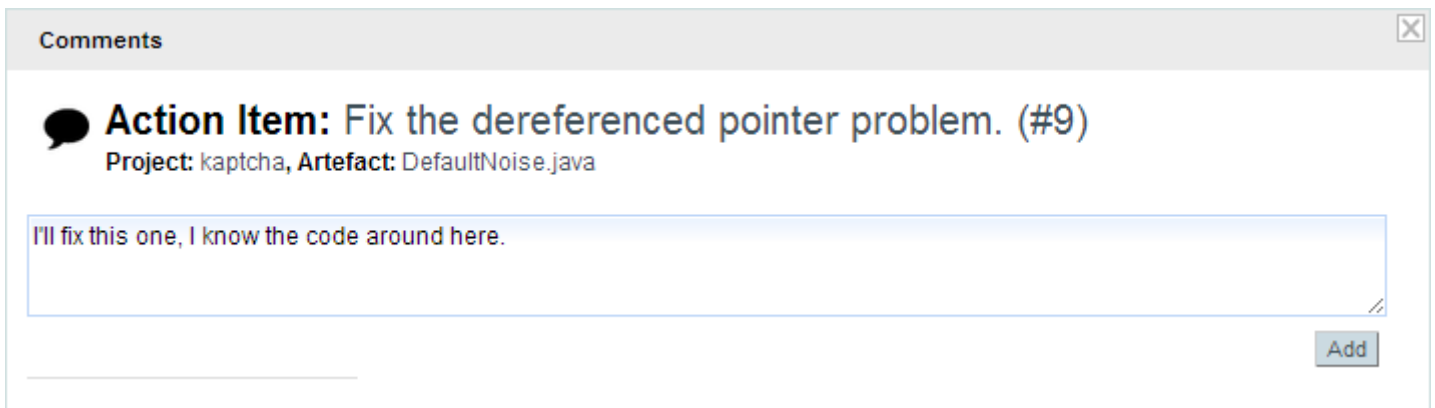
When you close the pop-up, notice that the speech bubble icon next to the chart you commented changes to indicate that a discussion about this item has been started.



A chart thumbnail with a discussion indicator

8.1.2. Commenting Action Items

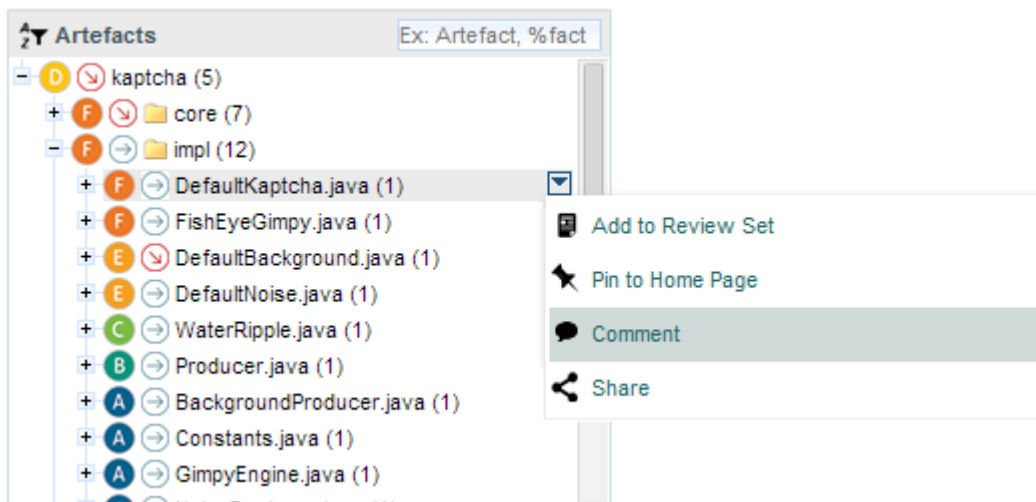
In the Action Items tab, you bring up the comment pop up by clicking the speech bubble in the **Comments** column of the table. Links allow you to jump to the Action Item's detailed description, the application level dashboard or the artefact dashboard directly.



A chart thumbnail with a discussion indicator

8.1.3. Commenting From the Artefact Tree

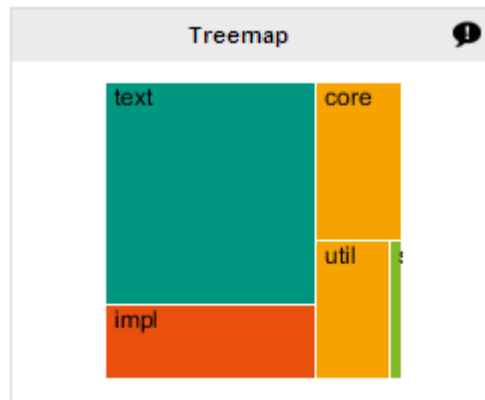
You can start discussions on artefacts by clicking **Comment** item in the artefact action menu, as shown below:



The artefact context menu item to bring up the comment pop-up for artefacts

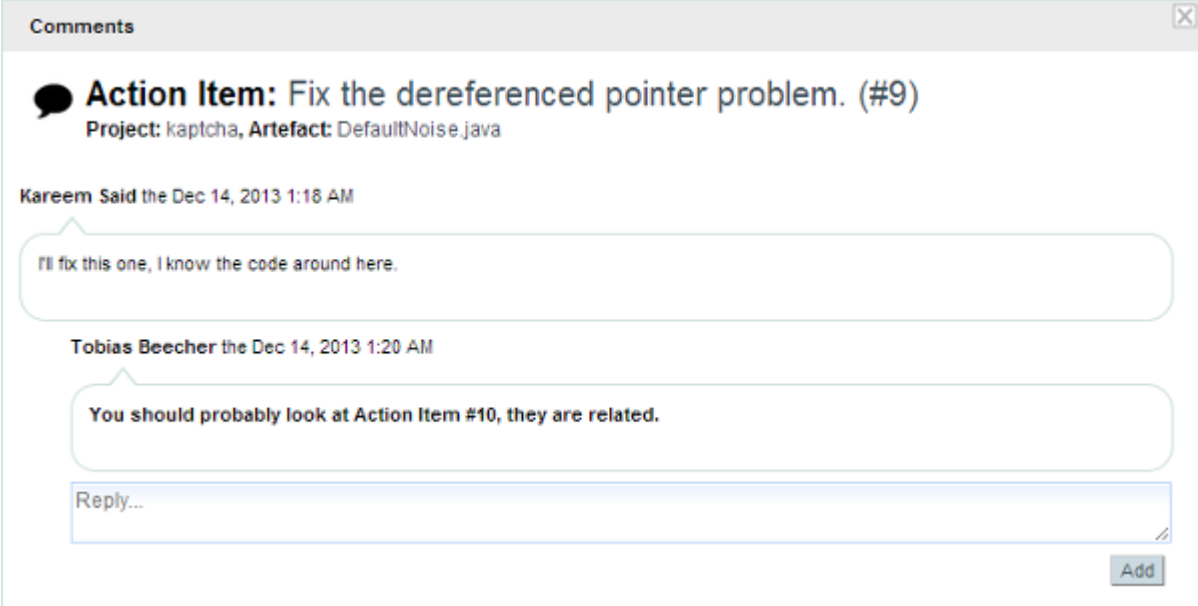
8.1.4. Following Discussions

When you log in to Squore, you can find out which discussions have new comments by looking at the items that show the **New Comment!** icon:



You have unread comments in the discussion about this chart

In the discussion pop-up, new comments since your last visit are highlighted:

A "Comments" pop-up window with a close button in the top right. The title is "Action Item: Fix the dereferenced pointer problem. (#9)" with "Project: kaptcha, Artefact: DefaultNoise.java" below it. The first comment is by "Kareem Said" on Dec 14, 2013 at 1:18 AM, with the text "I'll fix this one, I know the code around here." The second comment is by "Tobias Beecher" on Dec 14, 2013 at 1:20 AM, with the text "You should probably look at Action Item #10, they are related." Below the comments is a "Reply..." input field and an "Add" button.

A reply to my comment appears in bold

You can also get an exhaustive view of all the discussions for the project by viewing them in the **Comments** tab in the Explorer:

Annotation	Element	Replies/Views	Last Comment	Status
OK, this confirms what we thought about v6. Created by Augustus Hill (Dec 14, 2013 1:03 AM)	Chart: Code Stability Trend	Replies: 0 Views: 0	Augustus Hill Dec 14, 2013 1:03 AM	Open
We probably need to address this in our next post-mortem guys. Created by Augustus Hill (Dec 14, 2013 12:57 AM)	Chart: Innovation Rate	Replies: 1 Views: 3	Augustus Hill Dec 14, 2013 1:11 AM	Open
I'll fix this one, I know the code around here. Created by Kareem Said (Dec 14, 2013 1:18 AM)	Action Item: Fix the dereferenced pointer problem. (#9)	Replies: 1 Views: 3	Tobias Beecher Dec 14, 2013 1:20 AM	Open
This is one of our worst files, it actually shows up in the Highlights tab in the top 10! Created by Tobias Beecher (Dec 14, 2013 1:28 AM)	Artefact: DefaultBackground.java	Replies: 0 Views: 0	Tobias Beecher Dec 14, 2013 1:28 AM	Closed Locked
Checked, actually this is fixed in the next build. Created by Kareem Said (Dec 14, 2013 1:34 AM)	Action Item: (#8)	Replies: 0 Views: 1	Kareem Said Dec 14, 2013 1:34 AM	Open
False Positive Created by Kareem Said (Dec 14, 2013 1:35 AM)	Action Item: (#7)	Replies: 0 Views: 0	Kareem Said Dec 14, 2013 1:35 AM	Open
Total: 6				

Overview of discussions about a project in the Comments tab

From this view, discussions can be set to one the following statuses:

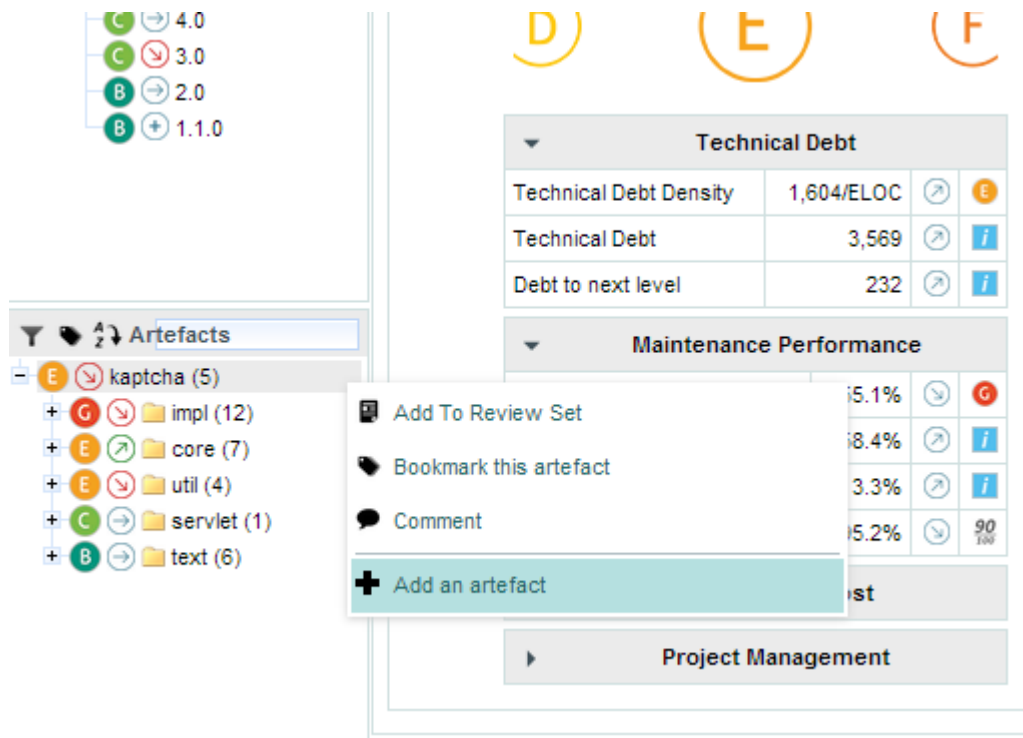
- **Open:** New comments are accepted in this discussion
- **Closed:** No new comments are accepted in this discussion, and it will be deleted in the next analysis
- **Locked:** No new comments are accepted in this discussion, but the discussion thread will be saved for the next analyses

Tip

Any user in the project team can view and take part in all open conversations in the project.

8.2. Adding and Removing Artefacts Manually

While you review results and comments, you can add artefacts manually to your project as needed. To add an artefact, click the node to which you want to add a child artefact. If this node supports adding artefacts, the Add an Artefact option will be available in the menu:



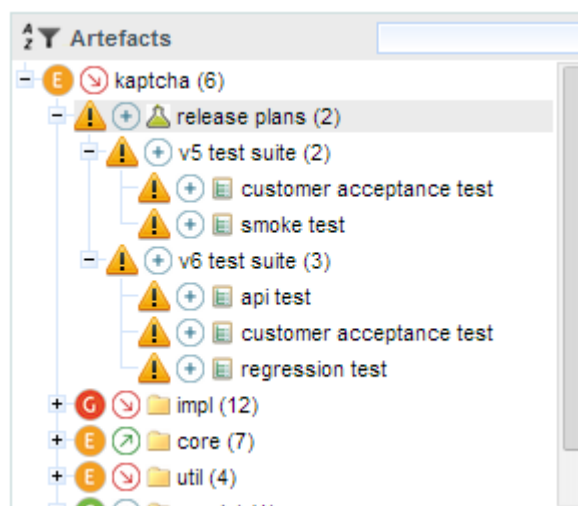
The artefact context menu with the Add an Artefact option highlighted

After you click the menu, you will have to select an artefact type and a name to add the artefact to the tree.

Tip

The type of artefact you can add depends on the model you are using. The model also defines where in the tree the new artefacts can be added.

Here is what the Artefact Tree looks like after manually building a test plan tree:



The artefact tree with manual artefacts not yet rated

When you run a new analysis of the project, the new artefacts will get rated according to what is defined in your model.

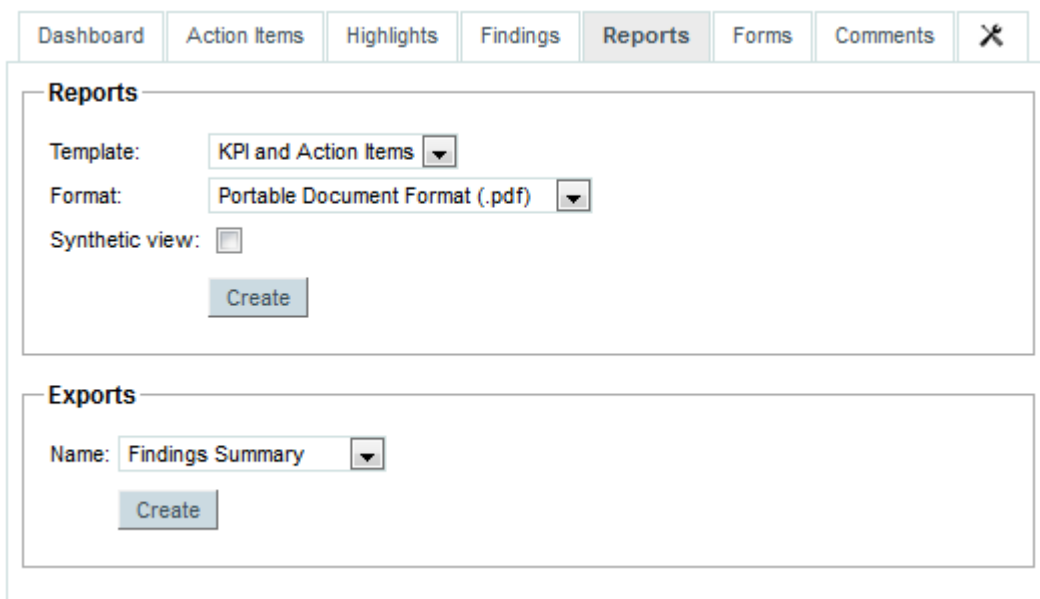
Note

Artefacts that were added manually can also be deleted from the tree. Note that artefact edition is tied to a permission in a user's role within a project. To learn more about roles, refer to Section 3.1.2, "User Roles"

8.3. Reporting Project Status

You can generate reports and export data to csv with Squore so you can communicate your progress to others.

In order to create a report, for the currently selected artefact in the tree, click the **Reports** tab in the Explorer. The Reports page opens as shown in the picture below:



The Reports page

The Reports tab offers a choice of report and export types in various output formats. Reports are primarily used to present information visually including charts and data about action items, while Exports can be used to extract information in a CSV file in order to import it into another tool. Clicking the **Create** button generates and downloads the file in your browser.

Reports and Exports are highly customisable, consult your Squore administrator or refer to the Squore Configuration Guide to learn more about how to tweak the report contents or format.

8.4. Providing Access to Collaborators

When a project is created, only the project creator can view it in Squore by default. In order to make it visible to more users, the project owner has to create a project team of users and groups and assign them permissions. This is done in the **Manage** page of a project in the **Team** tab, as shown below:

Manage Project - Earth

Project Properties
Versions
Team

Load from Project: Mars ▼ Load

Group / User	Role	Delete
demo	Owner ▼	

Add a Group or a User:

Apply
Discard

The Team tab

Tip

The project scope can be set directly from the command line when creating a new project, if you use the **teamUser** and **teamGroup** options. For more details, refer to the Command Line Interface manual.

In order to give visibility to the user **admin** over the projects created by the user **demo**, follow these steps:

1. Log in as the demo user and go to the My Projects page.
2. Click the Manage icon () for the project Earth
3. Click on the Team tab to view the project team.
4. Type admin in the **Add a Group or a User**. The list will show all users () and groups () available matching the search term.

Add a Group or a User:

Apply
Discard

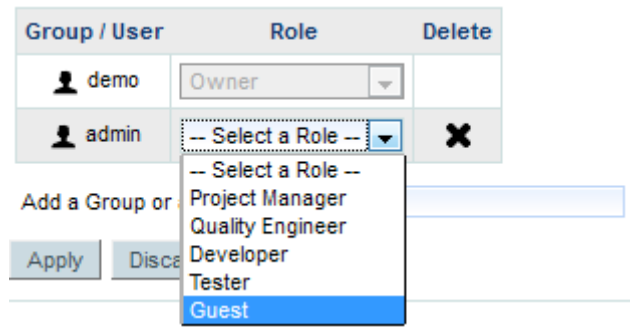
admin

admin

The users and groups matching admin

Click the admin user to add it to the project team.

5. Now that admin is listed in the project team, you need to pick a role for the user within this project. Select **Guest** from the list.



The users and groups matching admin

This predefined role allows a user to consult the results of baseline versions of a project without making any changes. For more information about roles, consult Section 3.1, “Understanding Profiles and Roles”.

6. Click **Apply** to apply your changes.

The admin user can now log in and will see the Earth project in their Explorer.

If you want to configure the rest of the sample projects the way you configured Earth, you can copy the project team to another project:

1. Click on **Manage > Team** for the project Mars
2. Select Earth from the **Load from Project** dropdown and click **Load**.
3. The users and their roles have now been copied as they were set up in the Earth project. You can make adjustments or click **Apply** to confirm your changes.

8.5. E-mail Notifications

You can configure each project in Squore so that an e-mail notification is sent out after a new version is created. This functionality is available for users who can create and manage projects, either in the **General Information** section of the project creation wizard, or the in the Project Properties tab of the Manage Project page:

Manage Project - rEarth

Project Properties

Versions

Team

Id:	94	
Name:	<input type="text" value="rEarth"/>	?
Analysis Model:	C Code SQuORE Risk Index	
Group:	<input type="text"/>	?
Creation Time:	Sep 15, 2014 5:07:35 PM	
Owner:	<input type="text" value="demo"/>	?
Automatic Baselining:	<input type="checkbox"/>	?
Color:	<input type="color" value="#007bff"/>	?
E-mail the creator of a version:		
	<input type="checkbox"/> On draft	<input type="checkbox"/> On baseline <input type="checkbox"/> On error ?
E-mail team members:		
	<input type="checkbox"/> On draft	<input type="checkbox"/> On baseline ?

E-mail notification options in Manage Project

The conditions on which you can to trigger an e-mail are:


- **On draft:** sends an e-mail every time a draft version is successfully analysed.
- **On baseline:** sends an e-mail every time a baseline version is successfully analysed, or every time a draft version is baselined.
- **On error:** sends an e-mail every time an analysis ends with the *Warnings* or *Error* status.

The e-mail contains a description of the version, the number of new artefacts, the number of action items, a list of the new action items and the number of new findings.







9. Keep it Tidy: Project Maintenance in Squore

9.1. Can I Delete a Version?

You can delete one or more of the last versions of a project if needed. This can be done from the **My Projects** page if you are the project creator or are a member of a role that allows managing the project.

If you want to delete the last version of the Earth project, log in as the demo user and click **My Projects**. Click the Manage icon () and open the **Versions** tab to view the list of versions created for this project:

Manage Project - Earth


Project Properties		Versions		Team			
Id	Version	Creation Time	Owner	Status	Baseline	Log	Debug Info
<input type="checkbox"/>	6	V1	Sep 8, 2014 11:27:16 AM	demo	Successful	Yes	 Download
<input type="checkbox"/>	7	V2	Sep 8, 2014 11:27:32 AM	demo	Successful	Yes	 Download
<input type="checkbox"/>	8	V3	Sep 8, 2014 11:28:08 AM	demo	Successful	Yes	 Download
<input type="checkbox"/>	9	V4	Sep 8, 2014 11:28:18 AM	demo	Successful	Yes	 Download
<input type="checkbox"/>	10	V5	Sep 8, 2014 11:28:27 AM	demo	Successful	Yes	 Download
<input type="checkbox"/>	11		Sep 8, 2014 11:28:36 AM	demo	Successful	No	 Download
<input type="checkbox"/>	12				Work in progress		

Delete

The Versions of the Earth project

Check the box next to the version you want to delete. All versions created after the version you selected will also be checked. Click the **Delete** button to reach a summary page where you can confirm which versions will be deleted, and click **Confirm** to launch the delete process.

9.2. Can I Delete a Project?

Projects can be deleted by their creator or members of a role that allows to manage projects. In order to delete a project, click **My Projects** and click the delete icon () next to the project you want to delete. After confirming the operation, the project is deleted from the Squore database and cannot be restored.

9.3. Squore Server Administration

A Squore Administrator can access functionality that does not involve working with projects directly. You can access the **Administration** menu if you need to perform any of the following tasks:

- Create, update, remove and deactivate Squore users (**Administration > Users**)

- Create, update, and remove groups (**Administration > Groups**)
- Create, update, and remove profiles (**Administration > Profiles**)
- Create, update, and remove roles (**Administration > Roles**)
- Configure and monitor the Squore Server installation (**Administration > System**)
- View and manage all projects created on Squore Server (**Administration > Projects**)
- Reload the server configuration from disk (**Administration > Reload Configuration**)
- download the server log (**Administration > Server Log**)

For more information about administration functionality, consult the Online Help.

9.4. What About Server Maintenance?

Server maintenance, including database backups need to be carried out by a system administrator directly on Squore Server. If you need to know more about the backup options offered by Squore, refer to the Squore Installation and Administration Guide.

10. Repository Connectors

10.1. Folder Path

10.1.1. 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.

10.1.2. Usage

Folder Path has the following options:

- **Datapath (path, mandatory)** Specify the absolute path to 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 ]"
```

10.2. Zip Upload

10.2.1. Description

This Repository Connector allows you to upload a zip file containing your sources to analyse. Select a file to upload in the project creation 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 upload 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.

10.2.2. Usage

This Repository Connector is only available from the web UI, not from the command line interface.

10.3. ClearCase

10.3.1. 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

10.3.2. 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_disp
```

10.4. TFS

10.4.1. 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.

10.4.2. 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=[multipleCho
```

10.5. Git

10.5.1. 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/>.

10.5.2. 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=[multipleCh
```

10.6. MKS

10.6.1. Description

This Repository Connector allows analysing sources hosted in PTC's MKS Source Integrity, a software system lifecycle management and application lifecycle management platform developed by MKS Inc. first released in 2001.

For more details, refer to <http://www.ptc.com/products/integrity/>.

10.6.2. Usage

MKS has the following options:

- **Server Hostname (hostname, mandatory)** Specify the name of the MKS server. This value is passed to the command line using the parameter `--hostname`.
- **Port (port)** Specify the port used to connect to the MKS 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 MKS 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 MKS is:

```
-r "type=MKS,hostname=[text],port=[text],project=[text],revision=[text],scope=[text],
```

10.7. Synergy

10.7.1. 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.

10.7.2. 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=[multipleChoice]"
```

10.8. SVN

10.8.1. Description

Connecting to an SVN server is supported using svn over ssh, or by using a username and password. The command run by the server to extract the source code is `svn export --force --non-interactive $url`.

For more details, refer to <https://subversion.apache.org/>.

10.8.2. 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.

→ **Authentication (useAccountCredentials, default: NO_CREDENTIALS)**

→ **Username (username)**

→ **Password (password)**

The full command line syntax for SVN is:

```
-r "type=SVN,url=[text],rev=[text],useAccountCredentials=[multipleChoice],username=[text]"
```

10.9. CVS

10.9.1. 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>.

10.9.2. 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]"
```

10.10. Perforce

10.10.1. 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.

10.10.2. 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=[mult
```

10.11. 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"
```

10.12. 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.

Select Data Providers

AntiC
 CPPCheck (plugin)
 CPPCheck

Note: Data Providers listed as plugins above require a one-time download of binary components before they can be executed for the first time. Consult the Installation Guide for more information.

▼ CPPCheck

XML file containing CPPCheck 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"
```

11. Data Providers

This chapter describes the Data Providers shipped with Squore and the default parameters that they accept via the Command Line Interface.

11.1. AntiC

11.1.1. 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 <INSTALLDIR>/addons/tools/Antic_auto/bin/ && gcc antic.c -o antic
```

11.1.2. 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]"
```

11.2. BullseyeCoverage Code Coverage Analyzer

11.2.1. 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/>.

11.2.2. 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]"
```

11.3. CPD

11.3.1. 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>.

11.3.2. 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]"
```

11.4. CPD (plugin)

11.4.1. 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>.

Note

This data provider requires an extra download to extract the CPD binary in `<INSTALLDIR>/addons/tools/CPD_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.

11.4.2. Usage

CPD (plugin) has the following options:

- **Run CPD during the analysis. (cpd_auto, default: true)** Check this box if you want to run CPD during the analysis in order to generate metrics and findings for the source code specified.

The full command line syntax for CPD (plugin) is:

```
-d "type=CPD_auto,cpd_auto=[booleanChoice]"
```

11.5. Cppcheck

11.5.1. 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/>.

11.5.2. 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]"
```

11.6. Cppcheck (plugin)

11.6.1. 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/>.

Note

On Windows, this data provider requires an extra download to extract the Cppcheck binary in `<INSTALLDIR>/addons/tools/ CPPCheck_auto/`. On Linux, you can install the cppcheck application anywhere you want as long as the cppcheck command is available from the command line. 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.

11.6.2. 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=CPPCheck_auto,dir=[text]"
```

11.7. CPPTest

11.7.1. 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/>.

11.7.2. 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 ]"
```

11.8. CheckStyle

11.8.1. 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/>.

11.8.2. 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 ]"
```

11.9. CheckStyle (plugin)

11.9.1. 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 `<INSTALLDIR>/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.

11.9.2. 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.

The full command line syntax for CheckStyle (plugin) is:

```
-d "type=CheckStyle_auto,configFile=[text],xmx=[text]"
```

11.10. CheckStyle for SQALE (plugin)

11.10.1. 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/>.

Note

This data provider requires an extra download to extract the CheckStyle binary in <INSTALLDIR>/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.

11.10.2. 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]"
```

11.11. Cobertura

11.11.1. 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/>.

11.11.2. 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]"
```

11.12. CodeSonar

11.12.1. 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>.

11.12.2. 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 ] "
```

11.13. Coverity

11.13.1. 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/>.

11.13.2. 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 ] "
```

11.14. FindBugs

11.14.1. 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/>.

11.14.2. 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 ] "
```

11.15. FindBugs (plugin)

11.15.1. 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 <INSTALLDIR>/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.

11.15.2. 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] "
```

11.16. FxCop

11.16.1. 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).

11.16.2. 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 ] "
```

11.17. GCov

11.17.1. Description

GCov is a Code coverage program for C application. GCov generates an XML results file which can be imported to generate metrics.

For more details, refer to <http://gcc.gnu.org/onlinedocs/gcc/Gcov.html>.

11.17.2. Usage

GCov has the following options:

- **XML Results file (xml)** Specify the path to the XML file containing the GCov results.

The full command line syntax for GCov is:

```
-d "type=GCov , xml=[ text ] "
```

11.18. GNATcheck

11.18.1. 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/>.

11.18.2. 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 ] "
```

11.19. GNATCompiler

11.19.1. 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/>.

11.19.2. 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]"
```

11.20. JUnit

11.20.1. 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/>.

11.20.2. Usage

JUnit has the following options:

- **Results folder (resultDir)** 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]"
```

11.21. JaCoCo

11.21.1. 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/>.

11.21.2. Usage

JaCoCo has the following options:

- **XML report (xml)** 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 ] "
```

11.22. Klocwork

11.22.1. 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>.

11.22.2. 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 ] "
```

11.23. Rational Logiscope

11.23.1. 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>.

11.23.2. 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 ] "
```

11.24. NCover

11.24.1. 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/>.

11.24.2. 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] "
```

11.25. Oracle PLSQL compiler Warning checker

11.25.1. 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/>.

11.25.2. 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|node2

The full command line syntax for Oracle PLSQL compiler Warning checker is:

```
-d "type=Oracle_PLSQLCompiler , log=[text] , prefix=[text] "
```

11.26. MISRA Rule Checking using PC-lint

11.26.1. 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>.

11.26.2. 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]"
```

11.27. PMD

11.27.1. 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>.

11.27.2. 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]"
```

11.28. PMD (plugin)

11.28.1. 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 `<INSTALLDIR>/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.

11.28.2. 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]"
```

11.29. Polyspace

11.29.1. 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>.

11.29.2. 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]"
```

11.30. MISRA Rule Checking using Polyspace

11.30.1. 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>.

11.30.2. Usage

MISRA Rule Checking using Polyspace 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 MISRA Rule Checking using Polyspace is:

```
-d "type=Polyspace_MISRA,resultDir=[text],unitByUnit=[booleanChoice]"
```

11.31. Polyspace (plugin)

11.31.1. 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 Deferencement 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 <INSTALLDIR>/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.

11.31.2. 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]"
```

11.32. MISRA Rule Checking with QAC

11.32.1. 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>.

11.32.2. 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]"
```

11.33. Unit Test Code Coverage from Rational Test RealTime

11.33.1. 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/>.

11.33.2. 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]"
```

11.34. ReqIF

11.34.1. 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/>.

11.34.2. Usage

ReqIF has the following options:

- **ReqIF file (file)** Specify the path to the XML ReqIF file. Note that the XML file will be validated using the schema available from <http://www.omg.org/spec/ReqIF/20110401/reqif.xsd>.
- **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,file=[text],objType=[text]"
```

11.35. SQL Code Guard

11.35.1. 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>.

11.35.2. 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]"
```

11.36. Squan Sources

11.36.1. 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. For example, a project mixing php and javascript files can be analysed with:

```
--dp "type=SQuORE,languages=php:.php;javascript:.js,.JS"
```

11.36.2. Usage

Squan Sources has the following options:

→ **Languages (languages)** 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.

- **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], genCG=[booleanChoice], qualified=[booleanCh
```

11.37. Squore Import

11.37.1. 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>.

11.37.2. 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]"
```

11.38. Squore Virtual Project

11.38.1. 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>.

11.38.2. 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]"
```

11.39. StyleCop

11.39.1. 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/>.

11.39.2. 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]"
```

11.40. StyleCop (plugin)

11.40.1. 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

This data provider requires an extra download to extract the StyleCop binary in `<INSTALLDIR>/addons/tools/StyleCop_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.

11.40.2. 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]"
```

11.41. Tessy

11.41.1. 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.

For more details, refer to <http://www.hitex.com/index.php?id=172>.

11.41.2. 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]"
```

11.42. OSLC

11.42.1. 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/>.

Note

This data provider requires extra perl modules to be available on Linux machines: HTML::TreeBuilder, HTTP::Server::Simple, HTTP::Server::Simple::CGI, WWW::Mechanize, Date::Parse, Date::Language, Date::Format and Time::Zone.

11.42.2. 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,mb_code,professional_line,ir_submitted
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=[text]"
```

11.43. pep8

11.43.1. 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>.

11.43.2. 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]"
```

11.44. pylint

11.44.1. 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/>.

11.44.2. 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]"
```

Index

Symbols

- * What's New in Squore 2015-A?
 - Export items from the Highlights tab to CSV., 39

A

- Access Management, 6
 - Profiles, 6
 - Roles, 7
- Action Items, 64
- Analysis, 21
- Analysis Model, 41
 - Analysis Model Editor (beta), 41
- Artefacts, 25
- Attributes, 61

B

- Baseline Version, 18

C

- Capitalisation Base, 66, 69
 - Distribution, 67
 - Statistics Aggregates, 67
- Charts
 - Bubble, 47
 - Control Flow Chart, 34
 - Function Maintainability Level, 31
 - Maintainability Level vs. Business Value, 47
 - Quadrant Chart, 46
- Checklists, 61
- ClearCase, 20
- Client, 19
- Code Comparison, 53
- Code Review, 52
- Collaboration, 79
 - Comments and Notifications for dashboard elements, 73
- Command Line Interface, 19
- Comments, 73
- Computation, 50
- Continuous Integration
 - Jenkins, 20
- Correlation
 - Correlation, 69
- CSV, 79
- Current Version, 18
- CVS, 20

D

- Dashboard, 26

- Analysis Model Dashboard, 45
 - Score Card, 27
- Dashboard Editor, 44
- Data Mining, 66
- Data Providers, 12
 - AntiC, 92
 - BullseyeCoverage Code Coverage Analyzer, 92
 - CheckStyle, 95
 - CheckStyle (plugin), 95
 - CheckStyle for SQALE (plugin), 96
 - ClearCase, 85
 - Cobertura, 96
 - CodeSonar, 97
 - Coverity, 97
 - CPD, 92
 - CPD (plugin), 93
 - Cppcheck, 93
 - Cppcheck (plugin), 94
 - CPPTest, 94
 - CVS, 89
 - FindBugs, 97
 - FindBugs (plugin), 98
 - FxCop, 98
 - GCov, 99
 - Git, 87
 - GNATcheck, 99
 - GNATCompiler, 99
 - JaCoCo, 100
 - JUnit, 100
 - Klocwork, 101
 - MISRA Rule Checking using PC-lint, 102
 - MISRA Rule Checking using Polyspace, 104
 - MISRA Rule Checking with QAC, 105
 - MKS, 87
 - NCover, 101
 - Oracle PLSQL compiler Warning checker, 102
 - OSLC, 111
 - pep8, 112
 - Perforce, 90
 - PMD, 103
 - PMD (plugin), 103
 - Polyspace, 104
 - Polyspace (plugin), 105
 - pylint, 112
 - Rational Logiscope, 101
 - ReqIF, 106
 - SQL Code Guard, 107
 - Squan Sources, 107
 - Squore Import, 109
 - Squore Virtual Project, 109
 - StyleCop, 110

StyleCop (plugin), 110
SVN, 89
Synergy, 88
Tessy, 111
TFS, 86
Unit Test Code Coverage from Rational Test
RealTime, 106
Diff, 53
Distribution, 67
Draft Version, 18
Drill-down, 25, 48

E

E-mail Notifications, 81
Evolution, 35

F

Favourites, 70
Filtering, 31
Findings, 50
Forms, 61

G

Git, 20

H

Help
 Debug Info, 4
 Debug info, 4
 Log Files, 4, 4
 Online Help, 3
 Project Logs, 4
 Support, 4
 User Guides, 4
 Wiki, 4
Highlights, 38

I

Icons
 Build Icon, 15
 Deteriorated Icon, 35
 Explorer Icons, 22
 Filter Icon, 31
 Sort Icon, 30
Indicators, 38, 49

L

Language
 User Interface Language, 9
Login Page, 7
Logout, 9

M

Maintenance, 84
Measures, 63
MKS, 20
Mnemonic, 41
Multi-Level Pie, 42

P

PDF, 79
Perforce, 20
PowerPoint, 79
Privacy, 79
Projects, 83
 Creating Projects, 10
 Deleting a project, 83
 Project List, 15
 Sample Projects, 3
 Updating Projects, 19

Q

Quality, 29

R

Rating, 50
Ratings, 25
Relaxation, 55, 57
Reports, 79, 79
Repository, 20
Repository Connectors
 Folder Path, 85
 Multiple Source Nodes, 90
 Zip Upload, 85
Review Set, 40
 Building a Review Set, 40
Rules, 51

S

Scale, 49
Score Card, 27
Searching, 32
Sorting, 30
Source Code, 53
Space Tree, 41
Squore CLI, 19
Squore Mobile, 71
Statistics, 67
SVN, 20
Synergy, 20

T

Teams, 79
TFS, 20

Themes, 9
Toolbar, 9
Trees, 23
 Artefact Tree, 25
 Indicator Tree, 25
 Project Portfolios, 24

V

Validator, 42
Version Date, 12
Versions
 Deleting a version, 83
 New Version, 15
Viewer, 41
Violations, 29, 55

W

Wizard
 Project Wizard, 10