Klaros-Testmanagement User Manual



Version 5.7.3

Publication date May 08 2025



Copyright © 2009-2025 verit Informationssysteme GmbH

Klaros-Testmanagement User Manual

by Talal Arif, Sabrina Gidley, Fabian Klaffke, Claudia Könnecke, Klaus Mandola, Patrick Reilly, and Torsten Stolpmann

Version 5.7.3

Publication date May 08 2025 Copyright © 2009-2025 verit Informationssysteme GmbH

Abstract

This document serves as the reference documentation for the Klaros-Testmanagement application. It gives a detailed description of the user interface, and the provided functionality.

Legal Notice. Copyright 2009-2025 verit Informationssysteme GmbH, Europaallee 10, 67657 Kaiserslautern, Germany. All rights reserved. This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or documentation may be reproduced in any form by any means without prior written authorization of verit Informationssysteme GmbH and its licensors, if any.



Trademarks

DockerTM and the Docker logo are trademarks or registered trademarks of Docker, Inc. in the United States and/ or other countries. Docker, Inc. and other parties may also have trademark rights in other terms used herein.

OracleTM and JavaTM are trademarks of Oracle and/or their affiliates.

Microsoft®, Excel®, SQLServer®, and Windows® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

JIRA® is a registered trademark of Atlassian Pty Ltd.

Other names may be trademarks of their respective owners.

Table of Contents

1. Feature Overview	. 1
1.1. Test Data Management	1
1.2. Test Planning	. 2
1.3. Test Execution	2
1.4. Evaluation, Statistics, Reports	. 3
1.5. Configuration	. 3
2. Introduction	. 5
2.1. Structure and Objects	. 5
2.2. Project - Managing a Test Project	5
2.3. Test Case - Defining a Test	. 5
2.4. Test Step - Determine the Test Activity Sequence	. 6
2.4.1. Test Step Result	. 7
2.4.2. Automated Test Cases	7
2.5. Test Segment - Reusing Test Steps	. 7
2.6. Test Suite - Arranging Test Cases	8
2.7. System under Test - The Test Object	. 8
2.8. Test Environment - The External Influence on the Test Result	. 8
2.9. Test Run - The Results of Executed Test Cases	. 9
2.10. Requirement – Properties of the System under Test	. 9
2.11. Iteration - Subdividing a Project into Phases	10
2.12. Job - Planning the Test Process	10
2.13. User - Users and their Rights	11
2.14. Overview of Objects	12
2.15. Issue - Reporting and Resolving Defects	12
3. Installation	14
3.1. License Model for Community and Enterprise Edition	14
3.2. System Prerequisites	14
3.2.1. Client Prerequisites	14
3.2.2. Server Prereguisites	15
3.3. Installation	17
3.3.1. Language Selection	17
3.3.2. Step 1: Welcome	18
3.3.3. Step 2: Information	18
3.3.4. Step 3: Licensing Agreements	19
3.3.5. Step 4: Target Path	20
3.3.6. Step 5: Select Installation Packages	21
3.3.7. Step 6: User Data	22
3.3.8. Step 7: Installation	23
3.3.9. Step 8: Perform External Processes	25
3.3.10. Step 9: Setup Shortcuts	25
3.3.11. Step 10: Installation Finished	26
3.4. Console-based Installation	27
3.4.1. Step 1: Language	27
3.4.2. Step 2: Welcome	28
3.4.3. Step 3: Information	28
3.4.4. Step 4: Licensing Agreements	28
3.4.5. Step 5: Target Path	29

3.4.6. Step 6: Select Installation Packages	29
3.4.7. Step 7: User Data	29
3.4.8. Step 8: Installation	30
3.4.9. Step 9: Perform External Processes	30
3.4.10. Step 10: Setup Shortcuts	30
3.4.11. Step 11: Installation Finished	31
3.5. Automated Installation Script	31
3.6. Starting the application	32
3.7. Stopping the application	32
3.8. Accessing the application	33
3.9 Undate Process	33
3.10 Important Files and their Locations	33
3 10 1 Log Files	34
3 10 2 Database Settings	34
3 10 3 Language Files	34
3.10.4. The Quotes File	25
3.10.5. The Derby Database	25
3.10.5. The Derby Database	25
2.11 Changing the Default Database	25
2 11 1 Maria DP	30 25
3.11.1. MidlidDD	30 26
	30
3.11.3. MYSQL	30
3.11.4. POSIGIESQL	30
3.11.5. Apache Derby	30
3.11.6. Setting up the Database Instance	37
3.12. Installing as a System Service	37
3.12.1. Installing as a Linux Service	3/
3.12.2. Installing as a windows Service	38
3.13. Monitoring	41
3.14. Configuring External Issue-Management-Systems	42
3.14.1. Jira Configuration	42
3.14.2. Redmine Configuration	43
3.14.3. Trac Configuration	43
3.15. SSL Support	44
3.16. Upgrading from Version 4	44
3.16.1. Required Version for Upgrade	44
3.16.2. Unattended Database Upgrade	44
3.16.3. Startup Time for Initial Upgrade to Version 5	45
3.16.4. Embedded Java Runtime Environment	45
3.16.5. Automatic Database Version Detection	45
3.16.6. Oracle Database no Longer Supported	45
3.17. Uninstall	45
Customization	47
4.1. Languages	47
4.1.1. Defining Language Files	47
4.2. Quote of the Day	47
Functional Overview	49
5.1. Login	49
5.2. Page Overview	50

4.

5.

5.2.1. Side Menu	50
5.2.2. Page Header	. 51
5.2.3. Content	54
5.3. Main Functions	. 68
5.3.1. Resolving save conflicts	. 68
5.3.2. Deleting, Purging and Restoring Objects	. 69
5.3.3. Referencing Object Properties	71
5.3.4. Referencing Attachments	73
6. Define	. 75
6.1. Projects	. 75
6.1.1. Overview Page	. 75
6.1.2. Details Page	77
6.2. Iterations	. 87
6.2.1. Overview Page	. 87
6.2.2. Details Page	89
6.3. Requirements	94
6.3.1. Overview Page	. 94
6.3.2. Details Page	97
6.4. Test Environments	101
6.4.1. Overview Page	101
6.4.2. Details Page	103
6.5. Systems under Test	107
6.5.1. Overview Page	107
6.5.2. Details Page	109
6.6. Test Segments	115
6.6.1. Overview Page	115
6.6.2. Details Page	117
6.7. Test Cases	120
6.7.1. Overview Page	120
6.7.2. Details Page	123
6.8. Test Suites	132
6.8.1. Overview Page	132
6.8.2. Detail Page	134
7. Plan	139
7.1. Jobs	139
7.1.1. Overview Page	139
7.1.2. Job - Details	142
7.2. Jobs from Test Cases	149
7.3. Jobs from Test Suites	150
7.4. Jobs by User	152
7.4.1. Jobs by User - Details	153
8. Execute	154
8.1. My Jobs	154
8.1.1. Action	155
8.1.2. Table Operations	155
8.1.3. Executing a Job	155
8.2. Run Test Case	156
8.2.1. Action	157
8.2.2. Table Operations	157

8.2.3. Executing a Test Case	157
8.3. Run Test Suite	163
8.3.1. Action	164
8.3.2. Table Operations	164
8.3.3. Executing a Test Suite	164
8.4. Continue Test Run	169
8.4.1. Action	170
8.4.2. Bulk Actions	171
8.4.3. Table Operations	171
8.4.4. Continue test run	171
8.5. Import Test Results	172
8.5.1. Supported Frameworks	173
9. Evaluate	177
9.1. The Dashboard	177
9.1.1. Default Dashboard	178
9.1.2. Editing a Dashboard Report	178
9.1.3. Report Types	179
9.2. Reports	186
9.2.1. Predefined Reports	186
9.3. Test Runs	190
9.3.1. Action	191
9.3.2. Bulk Actions	192
9.3.3. Table Operations	192
9.3.4. The Test Run Report	192
9.3.5. Details Page	193
9.3.6. Changes	195
9.4. Test Case Results	195
9.4.1. Action	197
9.4.2. Bulk Actions	197
9.4.3. Table Operations	197
9.4.4. Single Test Case Results	198
9.5. Test Suite Results	203
9.5.1. Action	203
9.5.2. Table Operations	203
9.5.3. Single Test Suite Results	204
9.6. Issues	207
9.6.1. Overview Page	207
9.6.2. Action	209
9.6.3. Bulk Actions	209
9.6.4. Table Operations	209
9.6.5. Details Page	210
9.6.6. Issue Details (Creating a new Issue)	213
9.6.7. Link Issue	221
10. Configure	223
10.1. Overview	223
10.2. Report Templates	223
10.2.1. Action	225
10.2.2. Report Templates Details	225
10.3. Users	226

	10.3.1. Overview Page	226
	10.3.2. Details Page	230
10.4.	System	232
	10.4.1. Miscellaneous	233
	10.4.2. Notifications	234
	10.4.3. Test Execution	237
	10.4.4. Interface	239
	10.4.5. Languages	240
10.5.	Integration	240
	10.5.1. Issue Management	241
	10.5.2. Requirements Management	249
	10.5.3. E-Mail	251
	10.5.4. Network	252
	10.5.5. LDAP	253
	10.5.6. CAS	256
10.6.	Backup	257
	10.6.1. Export	257
	10.6.2. Import	258
11. Custon	n Reports	259
11.1.	The Context Object	260
11.2.	Creating a Report Template	260
	11.2.1. Supported Parameter Types	263
	11.2.2. Dealing with Parameters	263
11.3.	Generating a Report	264
11.4.	Example Report	264
	11.4.1. Creating the Report Script	265
	11.4.2. Creating a PDE Report Template	265
11.5.	Creating a Chart	266
	1151 Pie Chart Report Script	266
	11 5 2 Pie Chart Report Template	267
	11.5.3 Embedding Images	267
	11.5.4 Creating an Excel Report Template	268
12 Import	/Fyport	260
12. 1110010	Importing Test Cases from Excel	269
12.1.	12.1.1 Import Format	269
	12.1.2 Prorequisites	205
	12.1.2. Fletequisites	271
12.2	Importing Test Cases from YMI	272
12.2.		272
	12.2.2. Execution	272
10.0	Importing Dequirements from Even	2/3 272
12.3.	12.2.1 Import Format	213 272
	12.3.1. IIIPUIT FUITIdt	∠/3
	12.3.2. Freeduisites	2/4
10 4	12.3.3. EXECUTION	2/4
12.4.	Importing and Synchronizing Requirements from XML	2/5
	12.4.1. FIEFEQUISITES	2/5
	12.4.2. Importing	2/5
40 F	12.4.3. Synchronization	2/5
12.5.	Importing Test Results	277

12.5.1. The JUnit XML++ format	277
12.5.2. Prerequisites	. 279
12.5.3. Execution	. 279
12.5.4. Jenkins Plugin	282
12.6. Exporting Test Results	282
12.6.1. Execution	. 282
12.7. Exporting Table Content	283
12.8. Backup/Recovery	284
12.8.1. Backup via REST	284
13. The Remote API	285
13.1. Overview	285
13.2. REST Clients	. 285
13.3. Documentation	286
A. Access Permissions	287
A.1. Role Permission Overview – Community Edition	. 287
A.2. Role Permission Overview – Enterprise Edition	. 287
B. The Klaros Object Model API Reference	288
B.1. Klaros Object Model API Reference	288
B.1.1. de.verit.klaros.core.model	. 288
B.1.2. Deprecated API	. 403
B.2. Scripting API Reference	403
B.2.1. de.verit.klaros.scripting	403
B.2.2. de.verit.klaros.scripting.context	. 407
B.2.3. de.verit.klaros.scripting.model	412
C. Test Case Import File Specification	416
C.1. <step></step>	. 418
C.2. <steps></steps>	418
C.3. <attachments></attachments>	418
C.4. <attachment></attachment>	418
C.5. <attribute></attribute>	. 419
C.6. <attributes></attributes>	419
C.7. <category></category>	419
C.8. <categorytree></categorytree>	. 419
C.9. <container></container>	420
C.10. <testcases></testcases>	420
C.11. <testcase></testcase>	. 420
C 12 <testsuite></testsuite>	421
C 13 <externalid></externalid>	421
C.14 <areatonic></areatonic>	422
C 15 <depends></depends>	422
C 16 <description></description>	422
C 17 <dochase></dochase>	422
C.18. <estimatedduration></estimatedduration>	422
C 19 <evaluation></evaluation>	422
C 20 <execution></execution>	422
C 21 <expectedresult></expectedresult>	422
C 22 < level>	422
C 23 <method></method>	422
C 24 <note></note>	423
	120

C.25. <postcondition></postcondition>	423
C.26. <precondition></precondition>	423
C.27. <revision></revision>	423
C.28. <priority></priority>	423
C.29. <shortname></shortname>	423
C.30. <state></state>	423
C.31. <team></team>	424
C.32. <traceability></traceability>	424
C.33. <variety></variety>	424
C.34. <name></name>	424
C.35. <value></value>	424
C.36. <content></content>	424
C.37. <categoryname></categoryname>	424
C.38. <categorydescription></categorydescription>	424
D. Requirement Import File Specification	425
D.1. <attachments></attachments>	426
D.2. <attachment></attachment>	426
D.3. <attributes></attributes>	426
D.4. <attribute></attribute>	426
D.5. <category></category>	426
D.6. <categorytree></categorytree>	427
D.7. <container></container>	427
D.8. <requirements></requirements>	427
D.9. <requirement></requirement>	427
D.10. <externaltestcaseids></externaltestcaseids>	428
D.11. <externalld></externalld>	428
D.12. <externalrevision></externalrevision>	428
D.13. <description></description>	428
D.14. <priority></priority>	429
D.15. <revision></revision>	429
D 16 <shortname></shortname>	429
D 17 <summary></summary>	429
D 18 <name></name>	429
D 19 <value></value>	429
D 20 <content></content>	429
D 21 <categoryname></categoryname>	429
D 22 <categorydescription></categorydescription>	429
F The Reporting Context API	431
F 1 The Klaros Report Context	431
E 2 KlarosScript Interface	431
E 3 Example Lavout Template	432
F. Icon Index	435
E1. Sections	435
E2. Actions	435
F21 Actions	435
F22 Execution Actions	436
E.2.3. Test Execution Actions	436
F24 Arranging Actions	437
F3 Table Operations	437

F.4. Information	. 437
F.5. Properties	. 438
F.5.1. Results	. 438
F.5.2. Priorities	. 438
F.6. Document Formats	. 438
Glossary	. 440
Index	449

List of Figures

2.1. Test Case Structure	5
2.2. Test Step	. 6
2.3. Test Segment	8
2.4. Test Suite	8
2.5. Test Run with Test Results	9
2.6. Job	11
2.7. Objects Overview	12
3.1. Language Selection	18
3.2. The "Welcome" Screen	18
3.3. The "Information" Screen	19
3.4. The "Licensing Agreement" Screen	20
3.5. The "Target Path" Screen	21
3.6. The "Select Installation Packages" Screen	22
3.7. The "User Data" Screen	23
3.8. The "Installation in Progress" Screen	24
3.9. The "Installation Finished" Screen	24
3.10. The "Processes" Screen	25
3.11. The "Setup Shortcuts" Screen	26
3.12. The "Installation Finished" Screen	27
3.13. Additional Tomcat Memory and Java Option Settings	40
3.14. Additional Tomcat Settings	41
3.15. The Redmine Authentication Section	43
3.16. The Trac Plugin Section	43
3.17. The Trac Permission Section	44
3.18. The Uninstaller	45
5.1. The "Login" Screen	49
5.2. The "Projects" Page	50
5.3. The Search Field	51
5.4. The "Search Results" Page	52
5.5. The Status Bar	52
5.6. The Log Panel Icon	53
5.7. The Log Panel	53
5.8. The Help Menu	53
5.9. The Oser Menu Bullon	53
5.10. The opened user menu	54 54
5.11. Number of Active Filter in Ose	55
5.12. Filter by multiple selection	55
5.13. The "Categorization Edit" View	57
5.15. Select Categorizes	57
5.16. The Bulk Actions Section	58
5.17 Bulk Editing Objects	59
5.18 The "Test Suite" Print Page	60
5.19. Creating Bookmarks	61
5.20. The "Overview" Tab	61
5.21. The "User Defined" Tab	62
5.20 Edition on Enumeration Descents	63
5.22. Editing an Enumeration Property	00

5.23. The "User Defined" Tab	63
5.24. The "Revisions" Tab	64
5.25. The "Upload Attachments" Page	. 66
5.26. The "Results" Tab	67
5.27. The "Changes" Tab	68
5.28. The "Conflict Resolution" Dialog	69
5.29. The "Delete Objects" Dialog	. 70
5.30. The "Purge Objects" Dialog	70
5.31. The "Restore Objects" Dialog	. 71
5.32. Referenced Object Property	. 72
6.1. The "Projects" Page	. 75
6.2. Quick Selection of Projects	77
6.3. The "Properties" Tab - Projects	. 77
6.4. The "User Defined" Tab	. 78
6.5. The "Copy Objects" tab	. 79
6.6. The "Copy Objects" Dialog	80
6.7. The "Access" Tab	81
6.8. The "Assign Project Role" Dialog	81
6.9. The "Integration/Issue Management" Tab	83
6.10. The "Integration/Requirements Management" Tab	84
6.11. The Synchronized Types of the Requirements Management System	. 85
6.12. Linking Enumeration Values	86
6.13. The "Iterations" Page	87
6.14. Quick Selection of Projects and Iterations	89
6.15. The "Overview" Tab	. 90
6.16. The Iteration Success Rate	. 91
6.17. The Iteration Radar Chart	. 91
6.18. The "Properties" Tab - Iterations	92
6.19. The "Requirements" Page	. 95
6.20. The "Requirements Page" Using Remote Synchronization	. 97
6.21. The "Overview" Tab	. 98
6.22. The "Properties" Tab	99
6.23. The "Test Cases" Tab	100
6.24. The "Revisions" Tab	101
6.25. The "Test Environments" Page	101
6.26. The "Overview" Tab	104
6.27. The "Overview" Tab	105
6.28. The "Properties" Tab	106
6.29. The "Iterations" Tab	107
6.30. The "System under Test" Page	108
6.31. The "Overview" Tab	110
6.32. The "Overview" Tab	111
6.33. The "Properties" Tab	112
6.34. The "Issues" Tab	113
6.35. The "Iterations" Tab	114
6.36. The "Test Segments" Page	115
6.37. The "Properties" Tab	118
6.38. The "Steps" Tab	119
6.39. The "Test Cases" Page	120

6.40. The "Overview" Tab	123
6.41. The "Properties" Tab	126
6.42. The "Steps" Tab	127
6.43. The "Insert Attachment" Dialog	128
6.44. A Reference to an Attachment in the Expected Result Field	128
6.45. An Attachment Reference Replaced by a Preview of the Attachment	129
6.46. An Attachment Reference Replaced by a Hyperlink to the Attachment	129
6.47. The "Issues" Tab	130
6.48. The "Jobs" Tab	131
6.49. The "Test Suites" Page	132
6.50. The "Overview" Tab - Test Suite	135
6.51. The "Properties" Tab	136
6.52. Jobs	137
7.1. The "Maintain Jobs" Page	139
7.2. The "Overview" Tab	143
7.3. Status Graph of a Job	145
7.4. The "Properties" Tab	146
7.5. The "Dependencies" Tab	147
7.6. The "Add Dependency" Dialog	147
7.7. The "Add a Comment" Dialog	148
7.8. The "Work Log" Dialog	149
7.9. The "Jobs from Test Cases" Page	149
7.10. The "Schedule Execution Jobs" Dialog	150
7.11. The "Create a review job" Dialog	150
7.12. The "Jobs from Test Suites" Page	151
7.13. The "Schedule Execution Jobs" Dialog	151
7.14. The "Create a review job" Dialog	152
7.15. The "Jobs by User"Page	153
7.16. The "Jobs by User" Page	153
8.1. The "My Jobs" Page	154
8.2. The "Run Test Case" Page	156
8.3. The "Execute Test Case" Dialog	158
8.4. The "Step-by-step Instructions" View	159
8.5. The "Tabular Step Instructions" View	160
8.6. The "Edit step result" Dialog	161
8.7. The "Test Run Overview" Page	162
8.8. Creating a Review Job	163
8.9. The "Run Test Suite" Page	163
8.10. The "Execute Test Suite" Dialog	165
8.11. The "Test Case Overview" View	166
8.12. The "Permanently Skipping a Test Case" Dialog	167
8.13. The "Inserting a Reason Template" Dialog	167
8.14. The "Test Run Overview" View	168
8.15. The "Test Suite Execution Overview" View	169
8.16. The "Continue Test Run" Page	170
8.17. The "Continue Test Run" Page	172
8.18. The "Import Test Results" Page	172
8.19. The "Import Test Results" Page	173
9.1. The "Dashboard" Page	177

9.2. The "Configure Report" Dialog	179
9.3. The "Project Overview" Report	180
9.4. The "Latest Success Rate" Report	181
9.5. The "Test Activity" Report	181
9.6. The "System under Test Overview" Report	182
9.7. The "Test Environment Overview" Report	183
9.8. The "Test Progress" Report	184
9.9. The "Test Progress History" Report	185
9.10. The "Reports" Page	186
9.11. The "Test Environment" Report	187
9.12. The "System under Test" Report	188
9.13. The "Test Suite Overview" Report	188
9.14. The "Test Run History" Report	189
9.15. Generate a Parameterized Report	190
9.16. The "Test Runs" Page	190
9.17. The "Test Run" Report	193
9.18. The "Test Run" Report (continued)	193
9.19. The "Test Run Details" Page	195
9.20. The "Test Case Results" Page	196
9.21. The "Add test cases to test suite" Dialog	197
9.22. The "Test Case Results" Page	198
9.23. The "Test Case Result" Page	200
9.24. Editing a Test Step Result	201
9.25. The "Test Case Result" Print Page	202
9.26. Viewing the Reason for Skipping a Test Case	202
9.27. The "Test Suite Results" Page	203
9.28 The "Test Suite Results" Page for a Single Test Suite	204
9.29. The "Test Suite Result" Details Page	206
9.30. The "Test Suite Result" Print View	207
9.31. The "Issues" Page	208
9.32. The "Properties" Tab	210
9.33. The "Test Cases" Tab	211
9.34. The "Assign Test Cases" Dialog	211
9.35. The "Systems under test" Tab	212
9.36. The "Assign Systems under test" Dialog	213
9.37. The "Create Issue" Page	213
9.38. The "Bugzilla Issue " Page	215
9.39. The "GiHub Issue" Page	216
9.40. The "Gitl ab Issue" Page	216
9.41. The "Jira Issue" Page	217
9.42. The "Mantis Issue" Page	218
9.43. The "Redmine Issue" Page	219
9.44. The "Trac Issue"Page	220
9.45. The "YouTrack Issue"Page	221
9.46. The "Link Issue" Panel	221
10.1. The "Overview" Page	223
10.2. The "Report Templates" Page	
	224
10.3. Generating a Parameterized Report	224 225
10.3. Generating a Parameterized Report10.4. The "Report Details" Page	224 225 226

10.5. The "Users" Page	227
10.6. The "Merge User" Dialog	228
10.7. The "Save New User" Dialog	229
10.8. The "Properties" Tab	230
10.9. The "Project Roles" Tab	231
10.10. The "Assign Project Roles" Dialog	232
10.11. The "Miscellaneous" Tab	233
10.12. The "Notification Schemes" Page	235
10.13. The "Add notification" Dialog	236
10.14. Assigning Projects to a Notification Scheme	237
10.15. The "Test Execution" Tab	237
10.16. The "Edit Reason" Dialog	238
10.17. The "Interface" Tab	239
10.18. The "Languages" Tab	240
10.19. The "Issue Management" Tab	242
10.20. The Bugzilla Project ID	243
10.21. The Jira Project ID	243
10.22. The Mantis Project ID	243
10.23. The Redmine Project ID	244
10.24. The YouTrack Project ID	245
10.25. The "Properties" Tab	247
10.26. Edit the status mappings	248
10.27. The "Requirements Management" Tab	249
10.28. The "E-Mail" Tab	251
10.29. The "Network" Tab	252
10.30. The "LDAP" Tab	253
10.31. The "LDAP Authentication" Dialog	255
10.32. The "CAS" Tab	256
10.33. The "Export" Tab	257
10.34. The "Import" Tab	258
11.1. The Report Generation Process	259
11.2. The "Report Templates" Page	261
11.3. The "Report Templates Details" Page	261
11.4. The "Parameters" Tab	262
11.5. Generating a Report	264
11.6. Entering Report Parameters	264
11.7. Pie Chart Example	266
12.1. Test Case Excel Sheet Sample	270
12.2. Requirement Excel Sheet Sample	273
12.3. Export Table Content to Excel	283
13.1. JSON Output after Searching for a User by its Name.	285
13.2. Remote API Documentation	286

List of Tables

3.1. Supported Operating Systems	. 15
3.2. Supported Database Systems	15
3.3. Supported External Issue-Management-Systems	. 42
12.1. General Property Coordinates	270
12.2. Test Step Coordinates	271
12.3. Custom Property Coordinates	271
12.4. General Property Coordinates	274
12.5. Custom Property Coordinates	274
A.1. Role Permission Overview – Community Edition	287
A.2. Role Permission Overview – Enterprise Edition	287
C.1. Element summary	416
C.2. <step> elements</step>	418
C.3. <steps> elements</steps>	418
C.4. <attachments> elements</attachments>	418
C.5. <attachment> elements</attachment>	418
C.6. <attribute> elements</attribute>	419
C.7. <attributes> elements</attributes>	419
C.8. <category> elements</category>	419
C.9. <categorytree> elements</categorytree>	419
C.10. <container> elements</container>	420
C.11. <testcases> elements</testcases>	420
C.12. <testcase> elements</testcase>	420
D.1. Element summary	425
D.2. <attachments> elements</attachments>	426
D.3. <attachment> elements</attachment>	426
D.4. <attributes> elements</attributes>	426
D.5. <attribute> elements</attribute>	426
D.6. <category> elements</category>	427
D.7. <categorytree> elements</categorytree>	427
D.8. <container> elements</container>	427
D.9. <requirements> elements</requirements>	427
D.10. <requirement> elements</requirement>	428
D.11. <externaltestcaseids> elements</externaltestcaseids>	428
E.1. Context Variables	431
F.1. Sections	435
F.2. Common Actions	435
F.3. Execution Actions	436
F.4. Test Execution Actions	436
F.5. Arranging Actions	437
F.6. Table Operations	437
F.7. Information	437
F.8. Results	438
F.9. Priorities	438
F.10. Document Formats	438

List of Examples

3.1. Sample auto-install.xml	31
3.2. Additional entries in mysql.cnf to support UTF-8 Character Sets	37
12.1. Excel Test Case Import via Command Line	272
12.2. XML Test Case Import via Command Line	273
12.3. Excel Requirement Import via Command Line	275
12.4. XML Requirement Import via Command Line	275
12.5. XML Requirement Synchronization via Command Line	276
12.6. XML Requirement Import via Command Line	277
12.7. Sample Test Test-step-result.xml	278
12.8. QF-Test import URL sample	281
12.9. Curl Command Line Example	281
12.10. Powershell Curl Alias Command Line Example	282
12.11. Curl Command Line Example	283
12.12. Powershell Curl Alias Command Line Example	283

Chapter 1. Feature Overview

1.1. Test Data Management

Management of Test-related Objects	All data of a test project like test cases, test suites, test require- ments, test environments, test objects/(SuT), test runs and test case results are stored together in a single database. The administration, processing and evaluation of the data is per- formed via a modern and comfortable web interface.
Agile and Classic Processes	In addition to classical development processes such as the waterfall or V-Model, iterative and agile methods such as Scrum or Kanban are also supported.
Requirements Coverage	Test requirements can either be managed directly in Klaros- Testmanagement or synchronized with external sources such as JIRA. The current requirements coverage can be displayed at any time.
Modularization	Frequently recurring test steps can be combined into seg- ments and can be reused. Test steps can be inserted as in a modular system and maintained with minimal effort.
Sharing	Test cases and test suites can be centrally maintained and used simultaneously in several projects. Changes are only nec- essary in a single location and will be propagated to selected projects.
Tracking Changes	For a seamless tracking of all changes applied to the managed data, such as test cases and test steps, all changes are logged automatically and displayed in a detailed change history.
User-Defined Custom Fields	Each test project can be configured individually using custom fields. These fields are available for jobs, requirements, iterations, test segments, test cases, test suites, test environments, test systems and test runs.
Categorization, Individual Views	All involved objects can be categorized and arranged in multi- ple tree structures as required. This is similar to a file system based on individual criteria and allows different views of a test project.
Saving Binary Attachments	Additional information such as test data, text documents, graphics, screenshots etc. can be uploaded into the applica- tion, saved and assigned to the objects iteration, job, require- ment, system under test, test environment, test segment, test case, test step, test suite and test case results.
Import Interfaces for Test Cas- es	Existing requirements and test cases can be imported from XML and Excel files. A REST interface, if needed, provides con- tinuous synchronization with an external data source or appli- cation.

Revisioning	Klaros-Testmanagement supports the versioning of test re-
	quirements, test segments, test cases and test suites. Each
	version can be maintained, executed, and evaluated separate-
	ly.

1.2. Test Planning

Task Planning	At the push of a button, jobs for executing test cases and test suites are created and assigned to individual persons. The sta- tus of the job execution is visible at any time.
Test Coordination	Jobs can be broken down into sub jobs, arranged hierarchical- ly and assigned to testers. By defining dependencies, you can set preconditions for successive test sequences.
Utilization, Progress and Suc- cess	Clear tables and graphs show usage, duration, and progress of each user's test activity. The progress and success rates of the tests are directly available and always up to date.
Test Case Reviews	Even test cases can be defective. Checking and fixing these tests can be assigned to users as a special job. The further course is logged automatically.

1.3. Test Execution

Guided Manual Test Execution	Klaros-Testmanagement guides the tester step by step through the manual tests. The test process is automatically logged and makes the data obtained available for evaluation. For each step, annotations and attachments can be added to document the test result.
Continuation of Interrupted Work	Manual execution can be paused whenever and be resumed any later time.
Separate Test Data and Test In- structions	The input and test data used in test cases, such as credentials, IDs, etc., can be defined and stored externally in Excel sheets. Tests can thus be parameterized in a targeted manner and can be reused in different scenarios.
Interoperability with continu- ous integration servers (Jenk- ins/Hudson)	A plug-in for the Jenkins Continuous Integration Server auto- matically imports the test results generated by a Jenkins build into Klaros-Testmanagement.
Integration with Issue/Defect Management	Found defects can be created and modified directly in issue management systems such as Bugzilla, GitHub, GitLab, JIRA, Mantis, Redmine and Trac during test execution, without hav- ing to leave Klaros-Testmanagement.
Import Interface for Test Au- tomation	Test results generated by test automation tools can be import- ed manually or via REST interface and then be merged with manual test results. Over 30 different formats and tools are supported: <i>AUnit, Boost Test, Check, CppTest, CppUnit, ctest,</i>

CUnit, embUnit, Fitnesse, Free Pascal Unit, Gauge (via xml-report plugin), GoogleTest, Glib / gtester, JBehave, JMeter, Jubula / GUIDancer, JSUnit, JUnit, MbUnit, MSTest, NUnit, PHPUnit, QF-Test, QTestLib, Ranorex, Selenium (via JUnit XML result format), Squish (via JUnit XML result format), TESSY, TestComplete, TestNG, TUSAR, UFT / QTP, UnitTest++, Valgrind and xUnit.net.

Integration with PerformanceFor comprehensive reporting, the results of non-functional
tests such as load and performance tests often need to
be evaluated and prepared. Klaros-Testmanagement offers
the possibility to directly import results from leading tools.
Supported load test software includes: BlazeMeter, JMeter,
Gatling, OctoPerf, SmartMeter.

1.4. Evaluation, Statistics, Reports

	Configurable Dashboard	The Dashboard provides numerous reports that show the state and progress of each activity and the entire project at a glance. It can be configured by the users according to their individual needs. Individual reports can be exported from there in several formats.
	Reports and Statistics	Evaluations and statistics can easily be prepared and retrieved from the information stored in the database. Klaros-Test- management ships with numerous predefined reports that can be applied directly.
	Customized Reports	Using a programming interface that provides access to the en- tire database, custom reports in PDF or Excel format can be freely defined and applied to your data.
	Excel Export of Data Tables	In case of a required post-processing or further evaluation, da- ta tables can be exported to an Excel file.
	Print View	Most views, diagrams and result tables provide a customiz- able print-friendly page view and can be sent directly from the browser to the printer.
	Print View Report File Export	Most views, diagrams and result tables provide a customiz- able print-friendly page view and can be sent directly from the browser to the printer. The report files can be exported into various file formats, such as PDF, HTML and CSV.
1.5.	Print View Report File Export Configuration	Most views, diagrams and result tables provide a customiz- able print-friendly page view and can be sent directly from the browser to the printer. The report files can be exported into various file formats, such as PDF, HTML and CSV.
1.5.	Print View Report File Export Configuration Rights and Roles	Most views, diagrams and result tables provide a customiz- able print-friendly page view and can be sent directly from the browser to the printer. The report files can be exported into various file formats, such as PDF, HTML and CSV. Klaros-Testmanagement supports a fine-grained role-based user management system, which is linked to a rights and roles system. In this way, individual assignments can be made or the access and editing rights of individual users for critical data can be restricted.

	be sent to the users or other persons involved as email noti- fication on request. For this purpose there are numerous con- figuration options that can be individually tailored to individual projects.
LDAP and Active Directory for User Authentication	User authentication and password management can be han- dled directly in Klaros-Testmanagement or via an external LDAP or Active Directory.
Single Sign-On Authentication via CAS	Klaros-Testmanagement supports Single Sign-on via CAS (Central Authentication Service).
Docker Integration	In addition to the native installation on Windows and Linux sys- tems, Klaros-Testmanagement can also be operated as a con- tainer within a docker-based environment. The application can be installed and managed in isolation, making updates easi- er to perform. Supported databases includes: <i>Apache Derby</i> , <i>MariaDB</i> , <i>MySQL</i> , <i>PostgreSQL</i> , <i>Microsoft SQL Server</i> .
Remote Data Interface (REST)	The data stored can be made available to other tools via the REST interface. Individual integrations with your own tools are easy to implement. This allows for a wide variety of additional integration options.
Backup and Restore	The backup data is stored in an XML format. Single projects or the entire database can be extracted and recovered.

Chapter 2. Introduction

2.1. Structure and Objects

Klaros-Testmanagement is a web-based test management solution for organizing, executing and evaluating all test activities. The functional scope covers all areas of the test process: test planning, test creation, test execution, assignment and evaluation of test tasks as well as test evaluation and report generation.

In this introduction the internal structure and function of the individual components, the objects, described. The following elements are defined as *Objects* in Klaros-Testmanagement: **Project**, **Test Case**, **Test Step**, **Test Segment**, **Test Step Result**, **Test Case Result**, **Test Suite**, **Test Suite**, **Test Suite**, **Test System**, **Test Run**, **Requirement**, **Iteration** and **Job**.

2.2. Project - Managing a Test Project

The top-level object is the *Project*. It contains all other objects required to define, plan, execute, and evaluate a particular test project.

Projects are typically self-contained units, but there are however also ways to reference objects in other projects. For more details on defining projects see <u>Section 6.1</u>.

2.3. Test Case - Defining a Test

Test Cases are the central objects of a project and may contain any number of test steps. They can be executed manually or automatically.



Figure 2.1. Test Case Structure

Test cases can be provided with attributes and include more detailed information:

Descriptive Attribute

Description Precondition Postcondition Expected result Test steps

Classifying Attributes

Execution type (Manual/Automated) **Priority** (High/Medium/Low)

Status (Draft/Locked/Approved/Skip)

Test Type (Functional, Non-Functional, Structural, Regression, Re-Test)

Design Technique (Black-Box, White-Box)

Test Level (Component Test, Integration Test, System Test, Acceptance Test)

Variety (Positive, Negative)

Requirement reference (Document/traceability)

enterprise dition Only available in Klaros-Testmanagement Enterprise Edition

Additional fields can be defined for special requirements.

The test cases consist of individual test steps in which the test activities are defined and described.

2.4. Test Step - Determine the Test Activity Sequence

The *Test Steps* describe the test execution procedure. Each test step contains an instruction to the tester as to which action to perform. If desired, images and documents can be attached to each test step.



Figure 2.2. Test Step

The description of the test step includes the following components:

Action	The textual description of the action the tester has to perform.
Precondition	The state the test system is in before the action begins.
Expected Result	The predicted behavior of the test system or component, based on the specification.
Postcondition	The expected state of the test system after the action has been performed.

2.4.1. Test Step Result

Klaros-Testmanagement guides the tester through the test execution step by step. Each step is completed by the tester with a **Test Step Result** and automatically recorded.

The test step result contains the following information:

Result	Passed: The test was completed as planned.
	Failure: The test does not deliver the expected result.
	Error: The test could not be performed.
	Inconclusive: The test does not deliver a clear result
	Skipped: The test step has been skipped.
Summary (optional)	A short summary of the test step result.
Description (optional)	A textual description of the test step result.
Duration	The automatically logged execution time of each individual test step.

2.4.2. Automated Test Cases

Automated test cases usually do not have test steps, since they are executed by an automated test tool (e.g. JUnit). The test results are provided in the form of a result file which will be imported into the system. Once Manual and automated test results are available can be jointly evaluated.

2.5. Test Segment - Reusing Test Steps

Only available in Klaros-Testmanagement Enterprise Edition

A *Test Segment* is a predefined sequence of test steps that is created as a separate object that can be inserted into any test case.





Often, certain test steps or sequences of test steps occur exactly the same in several test cases. To ensure that they are identical, these steps can be encapsulated in test segments and centrally managed, edited and stored.

The test steps of a test segment have exactly the same attributes as the test steps of a test case.

2.6. Test Suite - Arranging Test Cases

Test cases can be grouped into *Test Suites* to be executed together one after the other in one test run.





A test case can occur in multiple test suites. It is merely referenced, so that changes to the test case automatically affect all test suites that contain this test case.

2.7. System under Test - The Test Object

The *System under Test* is the component or system that is to be tested. Usually this is a software version, but can also be a specific device or other hardware. A test case can only be executed once a test system has been defined.

2.8. Test Environment - The External Influence on the Test Result

A *Test Environment* represents external conditions that may affect the test result. This can consist of hardware or software, such as a browser version used or the operating system in which the test system software was installed. A test case cannot be executed until a test environment has been defined.

2.9. Test Run - The Results of Executed Test Cases

A *Test Run* is automatically created when the tester starts to execute a test case or test suite. It is assigned to the test environment and system under test in use and collects all results that occur during the execution of the test case or test suite. The test runs thus document the entire test activities.



Figure 2.5. Test Run with Test Results

Test runs can be paused and continued later. The result (Passed, Failed, Error, Inconclusive, Skipped) appears in the evaluation only after all test cases of the test run have been completely executed. Until then, the test run with its collected results is not displayed.

Logged Parameters of a Test Run

During a test run, the following parameters are automatically logged and can be used for evaluation:

The test system used

The test environment used

The executed test case/test suite

The execution date

The executing user

enterprise edition

2.10. Requirement – Properties of the System under Test

Only available in Klaros-Testmanagement Enterprise Edition

A *Requirement* is a predefined condition to be fulfilled that describes the desired properties of the system under test. To ensure that the system under test satisfies these properties, test cases are defined and these are assigned to the requirement that check the test cases. In this way, it can be verified which requirements are covered by tests (*coverage*) and which of them were executed with the result *passed* (*conformity*).

From this set of information Klaros-Testmanagement is able to automatically determine coverage and compliance information for the whole project or selected objects once the first test results are available.

Attributes

Name

Priority (High/Medium/Low)

Short Description

Long Description

2.11. Iteration - Subdividing a Project into Phases

enterprise edition Only available in Klaros-Testmanagement Enterprise Edition

An *Iteration* represents a selected phase of a project. It encapsulates a subset of objects including all tasks and test results executed during that phase. Grouping objects in this way makes it easier to identify different test cycles in a project and enables better integration of the testing process, especially with agile software development practices.

In an iteration, the test systems, test environments and requirements under consideration are defined within a project for a specific period of time.

2.12. Job - Planning the Test Process



Test activities are planned and coordinated by means of a *Job*. For this purpose, the execution of a test case or a test suite is specified in the job.

After a job is created, it can be assigned to a user and executed and evaluated by him. Jobs can be repeated and executed multiple times, so that they can be assigned to multiple test runs. Jobs can be arranged hierarchically and dependencies between jobs can be specified.





In addition to test execution, other job types are supported. These are used for checking test cases, planning requirements and formulating any other jobs in text form.

Jobs can be nested and individually assigned to users. Therefore, it is possible to track progress of larger, distributed test activities.

2.13. User - Users and their Rights

Klaros-Testmanagement offers four different user roles, which are assigned when the user is created and can be changed later if necessary: *Administrator*, *Manager*, *Tester* and *Guest*.

Administrator	An Administrator always has access to all functions in Klaros- Testmanagement. In addition to the test-specific activities, this also includes user administration, backups and recovery as well as system-wide settings.
Test Manager	Test Managers have full read and write access to all data of a project. They can create and delete objects and manage project settings. This also includes creating, editing and delet- ing users with the "Tester" or "Guest"role.
Tester	Testers can run tests, edit test results and issues, and display reports. Beyond this, they have read-only access.
Guest	The "Guest" role is intended for project managers, customers or reviewers. This role has complete but read-only access to all data and can display, save and export reports in other formats.

The global user roles apply to all existing projects. However, if necessary, the role of a user can be assigned individually in each project.

Project Specific User Roles



In addition to the global assignment, it is possible to reassign the user roles per project or to exclude users from individual projects. The user retains his global role, but receives extended or restricted rights or no access in the selected project.

This is useful, for example, when integrating external employees who are only active in a specific project. On the other hand, a tester may need test manager rights in certain projects and can receive the corresponding assignment.

Detailed information about the role permissions can be found in <u>Appendix A</u>.



2.14. Overview of Objects

Figure 2.7. Objects Overview

2.15. Issue - Reporting and Resolving Defects

Klaros-Testmanagement integrates with issue tracking systems like Jira, Redmine, GitLab and many others (see <u>Section 12.5</u>). If a defect is found in a test step, it can be passed directly from Klaros to the connected issue management system without leaving the application.

The entry is automatically linked in both systems. Changes in the issue tracker are regularly synchronized with test management in the background. An automatically generated backlink in the issue management system leads back to the test case in Klaros-Testmanagement with one click.

Chapter 3. Installation

This chapter provides all information necessary to install Klaros-Testmanagement.

Klaros-Testmanagement is providing installers for Microsoft Windows- and Linux-based operating systems. The installer contains all additional programs which are needed for the first use of Klaros-Testmanagement. These include the file-based database (Apache Derby), the Apache Tomcat application server and a suitable Java runtime environment.



Ready to use Docker images for various databases are available as well.

These can be found along with detailed documentation at. https://github.com/klaros-testmanagement/klaros-docker.



Database not Suitable for Production Environments

The installer configures Klaros-Testmanagement to use the embedded Apache Derby database which is not intended for production use.

For production systems, it is strongly recommended to use one of the other recommended databases like MariaDB, MySQL, PostgreSQL or SQLServer. Each of them will provide significantly better performance. The database can be changed after running the initial installation process, (see <u>Section 3.11, "Changing the Default Database</u>").

3.1. License Model for Community and Enterprise Edition

The license model varies depending on the Klaros-Testmanagement edition used. Compared to the Klaros-Testmanagement Community Edition the Klaros-Testmanagement Enterprise Edition offers a limited amount of functions, but does not limit the number of users who can work with the software.

With Klaros-Testmanagement Enterprise Edition the purchased license limits the number of active users who can work with the software.

3.2. System Prerequisites

Klaros-Testmanagement is a web-based application. Client and server side therefore have different system requirements, which are described in detail in the next sections.

3.2.1. Client Prerequisites

To access Klaros-Testmanagement, a modern web browser supporting JavaScript is required.

3.2.1.1. Supported Browsers

- Apple Safari
- Google Chrome

- Microsoft Edge
- Mozilla Firefox



Microsoft Internet Explorer - Limited Support Notice

Support for Microsoft Internet Explorer will be gradually discontinued in upcoming releases. As of now, basic functionality is still available, but display problems may occur.

Klaros-Testmanagement is designed for a minimum screen resolution of 1280x960 pixels. Full-HD (1920x1080) is the recommended resolution.

3.2.2. Server Prerequisites

To run Klaros-Testmanagement, a Microsoft Windows or Linux 64-Bit operating system running on Intel/AMD x64 is required.

Name	Version	
Microsoft Windows	Windows 10, Windows 11, Windows Server 2012 Windows Server 2012 R2, Windows Server 2016, Windows Server 2019, Windows - Server 2022	
CentOS, Red Hat - Enterprise Linux (RHEL)	8.x, 9.x	
SuSE	SUSE Linux Enterprise Server (SLES) 15	
Ubuntu	20.04, 22.04, 24.04	
Debian	bullseye, bookworm	
Other Linux Distributions	Other Linux distributions are expected to function without prob- lems, as long as they are able to run the supplied 64-bit Java 11 - Runtime environment. Yet, we do not make any compliance state- ment regarding these distributions.	

Table 3.1. Supported Operating Systems

For optimal performance, using a separate database installation is highly recommended. This database is not required to run on the same physical machine as the Klaros-Testmanagement server.

Name	Version	
Apache Derby	10.15.1.3 (embedded and preconfigured as default)	
Microsoft SQLServer	SQLServer 2008, 2008 R2, 2012, 2014, 2016, 2017, 2019, 2022	
MySQL	Version 5.5, 5.6, 5.7, 8.0	
MariaDB	Version 10.3 or later	
PostgreSQL	Version 9.4 or later	

Table 3.2. Supported Database Systems

3.2.2.1. Microsoft Windows

To run Klaros-Testmanagement on Microsoft Windows operating systems the following requirements should be met:

• Minimum requirements

2 GB RAM, 3 GHz Dual Core CPU

Recommended requirements (Remote Database)

4 GB RAM, 3 GHz Quad Core CPU

Recommended requirements (Local Database)

8 GB RAM, 3 GHz Quad-Core CPU

• Operating system

Microsoft Windows Server 2012, Microsoft Windows Server 2012 R2, Microsoft Windows Server 2016, Microsoft Windows Server 2019, Microsoft Windows Server 2022, Microsoft Windows 10, Microsoft Windows 11.

3.2.2.2. Linux

To run Klaros-Testmanagement on Linux operating systems the following requirements should be met:

• Minimum requirements

2GB RAM, 2GHz Dual-Core CPU

Recommended requirements (Remote Database)

4GB RAM, 3GHz Quad-Core CPU

• Recommended requirements (Local Database)

8GB RAM, 3GHz Quad-Core CPU

Operating System

Linux Intel64/x86-64 (Intel 64 bit architecture) distribution containing glibc version 2.17 or better and version 2.2.1 or better of the GTK+ widget toolkit and associated libraries.



GTK+ libraries only required in Windowed Mode

The GTK+ libraries are only needed to run the Klaros-Testmanagement installer in windowed mode. Running the installer in console mode is possible without this requirement.

Restricting the Number of Open Files



Caution

Klaros-Testmanagement stores database search indices in the .klaros/indices folder in home directory of the user account that is running the Klaros-Testmanagement application server. During the indexing process the amount of open files may exceed the predefined limit in the Linux operating system.

The command **ulimit -n** shows the configured number of open files allowed, which should be set to a value of at least 4096. Please consult your system documentation on how to set this value (Usually by editing /etc/security/limits.conf).

3.2.2.3. Available Disc Space

Klaros-Testmanagement stores binary attachments in a file based repository inside the .klaros folder in home directory of the user account that is running the Klaros-Testmanagement application server.

Depending on the usage pattern a sufficient amount of disk space should be reserved for this account. If the file based Derby database is active, the database files will also be located in this directory.



Recommended Available Disc Space

Reserving 10 GB of disc space should be enough for typical usage patterns.

3.3. Installation

Under Microsoft Windows the installer is invoked by running the binary **Klaros-Testmanage**ment-<version>-Setup.exe.

Under Linux the installer can be started by running the binary **Klaros-Testmanagement-<version>-Setup.bin** in a user shell.

The following screens show each step involved in installing Klaros-Testmanagement.

3.3.1. Language Selection

Upon startup the installer is requesting the language used throughout the following installation procedure.

Currently both English and German languages are available.

Language Selection		\times
Please select your language		
Englisch		~
	ОК	

Figure 3.1. Language Selection

3.3.2. Step 1: Welcome

The initial step shows a welcome screen to the user (Figure 3.2).



Figure 3.2. The "Welcome" Screen

The installation may be aborted by clicking the Quit button. Clicking Next proceeds with the installation.

3.3.3. Step 2: Information

The second step shows information about the product and the revision history, listing the fixed issues and the newly added features (<u>Figure 3.3</u>).



Figure 3.3. The "Information" Screen

The installation may be aborted by clicking the Quit button. Clicking Previous goes back to the *Welcome* step and clicking Next proceeds with the installation.

3.3.4. Step 3: Licensing Agreements

The third step shows information about the license agreement for Klaros-Testmanagement. The license must be accepted to continue the installation. (<u>Figure 3.4</u>).


Figure 3.4. The "Licensing Agreement" Screen

The installation may be aborted by clicking the Quit button. Clicking Previous goes back to the *Information* step and clicking Next proceeds with the installation.

3.3.5. Step 4: Target Path

The fourth step requests the target path where Klaros-Testmanagement will be installed. The user can use the Browse button to search for the specific path in the local file system (Figure 3.5).



Install Folder Location

It is highly recommended using the suggested install location and **not** the Windows Programs folder. By default, Klaros-Testmanagement will not be able to start if installed there, as the Tomcat application server requires write access to the installation folder in order to deploy the web application. This is blocked by Microsoft Windows User Account Control (UAC).

If you wish to install Klaros-Testmanagement in this location, UAC must be disabled.

💦 IzPack - Ir	nstallation of Klaros-Testmanagement 5.1.0-SNAPSHOT	– 🗆 X
Target Pa	ath	ᢣ klaros
	, 10	
	Select the installation path:	Browse
1		gonaciii
*		
<i>6</i> 9		
8		
(Made with Iz	(Pack - http://izpack.org/)	Previous Next Ouit
-14P		True Tane

Figure 3.5. The "Target Path" Screen

The installation may be aborted by clicking the Quit button. Clicking Previous goes back to the *Licensing Agreement* dialog and clicking Next proceeds with the installation.

3.3.6. Step 5: Select Installation Packages

The fifth step allows selecting optional packages that are installed with Klaros-Testmanagement.

💦 IzPack - I	nstallation of Klaros-Testmanagement 5.1.0-SNAPSHOT	– 🗆 X
Select In Step 5 o	Istallation Packages	👱 klaros
*	Select the packs you want to install: Note: Grayed packs are required.	243,38 MB 30,43 MB
	Klaros-Testmanagement 5. 1.0-SNAPSHOT PDF-Documentation	334,78 MB 22,12 MB
8		
	Description The Java Server Runtime Environment - Version 11.0, 10.9.	
90) 88		
(Made with I:	Total space required: Available space: zPack - http://izpack.org/)	630,71 MB 9,05 GB
Help		Previous Next Quit

Figure 3.6. The "Select Installation Packages" Screen

The installation may be aborted by clicking the Quit button. Clicking Previous goes back to the *Target Path* dialog, clicking Next proceeds with the installation.

3.3.7. Step 6: User Data

In the sixth step the settings for the installed Tomcat application server can be changed.

IzPack - Installation	of Klaros-Testmanagement 5.1.0-SNAPSHOT	– 🗆 X
User Data Step 6 of 10		👱 klaros
	Tomcat Server Settings:	
<u></u>	Server-Port: 18005	
	HTTP-Port: 18080 Session Timeout: 60	
•	The minimum and maximum amount of memory available Minimum (MB): 512	to the Tomcat process:
	Maximum (MB): 1024 The username and the password of the administrator ac	count for managing the Tomcat server:
63	Username: admin	
e	Password: Retype Password:	
99	Launch Application Server	
88	Launch Browser	
(Made with IsPact tite		
	///zpack.org/)	Previous Next Quit

Figure 3.7. The "User Data" Screen

It is possible to set the ports used by Tomcat (Server-Port and HTTP-Port), the minimum and maximum amount of memory available to the Tomcat process as well as the user name and password of the Tomcat application server administrator.

In addition it is possible to control the startup of the application during the installation. If the *Launch Application Server* checkmark is set, the application server will be launched automatically during the installation process. If the *Launch Browser* checkmark is set, a native web browser instance will be redirected to the application login page. Both of these actions will be performed during the <u>Section 3.3.9, "Step 8: Perform External Processes"</u> phase.

The installation may be aborted by clicking the Quit button. Clicking the Previous button goes back to the Select Installation Packages step and clicking Next button proceeds with the installation.

3.3.8. Step 7: Installation

The seventh step starts the installation of Klaros-Testmanagement and shows the installation progress. The following screenshots show the *Installation in Progress* Screen and the *Installation finished* screen (Figure 3.8).

IzPack - Installation of Klaros-Testmanagement 5.1.0-SNAPSHOT	– 🗆 X
Installation	
Step 7 of 10	
Pack installation progress: C: Users \Arif\Claros-Testmanagement\webapps\Karos-web.war Klaros-Testmanagement 5, 1, 0-SNAPSHOT	
Overall installation progress:	
3 4	
9	
(Made with IzPack - http://izpack.org/)	Previous Next Quit
1 holp	Henry Quit

Figure 3.8. The "Installation in Progress" Screen

💦 IzPack - I	nstallation of Klaros-Testmanagement 5.1.0-SNAPSHOT	- 🗆 X
Installat	ion set to	ᢣ klaros
Step 7 o	of 10	
०००		
6		
-	Pack installation progress:	
W	[Finished]	
16	Overall installation progress: 4/4	
-		
\$		
88		
(Made with Iz	zPack - http://izpack.org/)	
Help		Previous Next Quit

Figure 3.9. The "Installation Finished" Screen

The installation may be aborted by clicking the Quit button. Clicking Next proceeds with the installation.

3.3.9. Step 8: Perform External Processes

The optional eighth step performs external processes such as setting the environment variables and starting the Tomcat application server.

💦 IzPack - In	stallation of Klaros-Testmanagem	ent 5.1.0-SNAPSHOT	– 🗆 X
Perform I	External Processes	4	klaros
		Processing	
		4 / 4	
	Using CATALINA BASE:	"C:\Users\Arif\Klaros-Testmanagem	ent"
	Using CATALINA_HOME:	"C:\Users\Arif\Klaros-Testmanagem	ent"
	Using CATALINA_TMPDIR:	"C:\Users\Arif\Klaros-Testmanagem	ent\temp"
	Using JRE_HOME:	"C:\Users\Arif\Klaros-Testmanagem	ent\jre"
	Using CLASSPATH:	"C:\Users\Arif\.klaros\resources;	C:\Users\Arif\Klaros-Tes
	Using CATALINA_OPTS:	" -Doak.locksupport=disabledad	d-opens=java.base/java.l
16			
-			
\$			
88			
	<		>
(Made with Iz	Pack - http://izpack.org/)		
Help			Previous Next Quit

Figure 3.10. The "Processes" Screen

The installation may be aborted by clicking the Quit button. Clicking Next proceeds with the installation.

3.3.10. Step 9: Setup Shortcuts

The ninth step allows to set the options for shortcuts. The installer can create shortcuts in the Start-Menu and on the desktop (<u>Figure 3.11</u>).

IzPack - Ir	nstallation of Klaros-Testmanagement 5.1.0-SNAPSHOT		- 🗆 X
Setup Sho	ortcuts of 10	5	klaros
	 ✓ Create shortcuts in the Start-Menu ✓ Create additional shortcuts on the desktop Select a Program Group for the Shortcuts: (Default) 7-Zip Accessibility Accessity	~	create shortcut for: ○ current user ④ all users
(Made with Iz Help	tRanos Festmanägement zPack - http://izpack.org/)		Previous Next Quit

Figure 3.11. The "Setup Shortcuts" Screen

This step looks different on Linux installations but has the same functionality.

The installation may be aborted by clicking the Quit button. Clicking Previous goes back to the *Perform External Processes* step and clicking Next proceeds with the installation.

3.3.11. Step 10: Installation Finished

The last step notifies the user that Klaros-Testmanagement was installed successfully and shows the path to the uninstall program (<u>Figure 3.12</u>).



Figure 3.12. The "Installation Finished" Screen

The Generate an automatic installation script button saves a complete script of the installation with the selected user choices, so that the same installation could be repeated unattended or on other machines.

The installation can be completed by clicking the Done button.

3.4. Console-based Installation

The installer also supports a GUI-less variant which only needs a shell/command window and can be invoked by adding the option -- -console to the command line.

This installer displays the content of an installer page in a panel in the shell window and lets the user input her choices via keyboard.

The buttons are replaced by short key sequences:

Enter	shows the next page or accepts a default value
1+Enter	selects a checkbox
0+Enter	deselects a checkbox
n+Enter	chooses the nth item from a list

3.4.1. Step 1: Language

The Language panel (<u>Section 3.3.1, "Language Selection"</u>) is displayed. Enter **0+Enter** for English or **1+Enter** for German Language.

Select your language

```
0 [x] eng
1 [] deu
Enter selection:
```

3.4.2. Step 2: Welcome

The *Welcome* panel (<u>Section 3.3.2, "Step 1: Welcome</u>") is displayed and wants to be committed by **1+Enter**.

```
Information

Version 5.5.5

(c) Copyright 2009-2023 verit Informationssysteme GmbH. All Rights Reserved.

Press 1 to continue, 2 to quit, 3 to redisplay

1
```

3.4.3. Step 3: Information

The *Information* panel (<u>Section 3.3.3, "Step 2: Information</u>") displays the release notes of the recent versions.

```
...
Version 5.5.5
New Features...
Press 1 to continue, 2 to quit, 3 to redisplay
1
```

3.4.4. Step 4: Licensing Agreements

The *License* panel (<u>Section 3.3.4, "Step 3: Licensing Agreements</u>") displays the Licensing Agreements, which must be accepted to continue with the installation. Commit with **1+Enter**.

```
Licensing Agreements

Limited Use Software License Agreement

PLEASE READ THIS AGREEMENT CAREFULLY BEFORE ATTEMPTING TO USE THE

SOFTWARE OF verit Informationssysteme GmbH AND RELATED DOCUMENTATION

(COLLECTIVELY, THE "SOFTWARE") AND BEFORE INSTALLING OR USING THE

SOFTWARE. BY INSTALLING OR USING THE SOFTWARE, YOU ARE CONSENTING

TO BE BOUND BY THIS AGREEMENT....

limitations may not apply to you. You may also have other rights

that vary from state to state.

Press 1 to accept, 2 to reject, 3 to redisplay

1
```

3.4.5. Step 5: Target Path

The *Target Path* panel (<u>Section 3.3.5, "Step 4: Target Path"</u>) looks like this with the console installer:



You can press **Enter** to accept the proposed path or enter a different one. The path will be created, if it did not already exist. If the directory already exists, a warning is displayed, and you can cancel the installation or continue.

3.4.6. Step 6: Select Installation Packages

The Select Installation Packages panel (<u>Section 3.3.6, "Step 5: Select Installation Packages</u>") is more complex in console mode. The required packages (Java, Tomcat and Klaros-Testmanagement) are displayed for information only. Entering **Y** selects the installation of the PDF documentation, while **N** deselects it. You are then prompted to confirm your entry with **1** and **Enter**.

```
Select Installation Packages

Select the packs you want to install:

[x] Pack 'Java Runtime Environment' required

[x] Pack 'Java Runtime Environment' required

[x] Pack 'Iomcat 9 Application Server' required

[x] Pack 'Klaros-Testmanagement required

[x] Include optional pack 'PDF-Documentation'

Enter Y for Yes, N for No: N

Y

Done!

Press 1 to continue, 2 to quit, 3 to redisplay

1
```

3.4.7. Step 7: User Data

After that, the installer requests user data (<u>Section 3.3.7, "Step 6: User Data"</u>). The installation program displays default values that can be accepted or overwritten by pressing **Enter**. Checkboxes show [x] if the checkbox is set, or [] otherwise. Pressing **Enter** accepts the current value, which can be set and unset explicitly with 1 and 0 respectively.

In this example we adopt most of the default values. Only the memory settings and the tomcat admin password (without default values) are entered and the two checkboxes are unset:

-----User Data

```
Tomcat Server Settings:
Server-Port: [18005]
HTTP-Port: [18080]
Session Timeout: [60]
The minimum and maximum amount of memory available to the Tomcat process:
Minimum (MB): [512]
400
Maximum (MB): [1024]
1024
The username and the password of the administrator account for managing the Tomcat server:
Username: [admin]
Password:
*******
Retype Password:
*******
 [] Launch Application Server
Enter 1 to select, 0 to deselect:
0
 [ ] Launch Browser
Enter 1 to select, 0 to deselect:
0
Press 1 to continue, 2 to guit, 3 to redisplay
```

3.4.8. Step 8: Installation

Packages are displayed as soon as they are installed (Section 3.3.8, "Step 7: Installation").

```
[ Starting to unpack ]
[ Processing package: Java Runtime Environment (1/4) ]
[ Processing package: Tomcat 9 Application Server (2/4) ]
[ Processing package: Klaros-Testmanagement (3/4) ]
[ Processing package: PDF-Documentation (4/4) ]
[ Unpacking finished ]
Installation finished
```

3.4.9. Step 9: Perform External Processes

This panel is skipped since we chose not to launch the server or a browser. Otherwise, startup messages of the server could be seen here. See <u>Section 3.3.9, "Step 8: Perform External Processes</u>" on how this is handled in GUI mode.

3.4.10. Step 10: Setup Shortcuts

In a GUI-less environment we typically do not want to create any menu entries or icons on the desktop, thus we choose **N**. See <u>Section 3.3.10, "Step 9: Setup Shortcuts"</u> for details on shortcut setup in windowed environments.

3.4.11. Step 11: Installation Finished

An automatic installation script with the selected user choices can be generated here, so that the same installation could be repeated for unattended updates or on other machines. Automatic installation scripts will work for unattended installations in console mode as well.

```
Generate an automatic installation script
Enter Y for Yes, N for No:
Y
Select the installation script (path must be absolute)[C:\Users\...\Klaros-Testmanagement
\auto-install.xml]
Installation was successful
Application installed on C:\Users\...\Klaros-Testmanagement
[ Writing the uninstaller data ... ]
[ Console installation done ]
```

After finishing the installation, Klaros-Testmanagement is located in the directory /opt/klaros-testmanagement. In <u>Section 3.12</u>, "Installing as a System Service" you can find more information how Klaros-Testmanagement can be set up as a system service to run it automatically at system startup.

3.5. Automated Installation Script

If an automated installation script was saved during Klaros installation, the program can be reinstalled with the same parameters. In this example it is assumed that the automated installation script was saved under auto-install.xml.

Linux: Klaros-Testmanagement-<version>-Setup.bin -- -console auto-install.xml

Windows: Klaros-Testmanagement-<version>-Setup.exe auto-install.xml

You may also use the following sample script. Please adapt the setting for installpath and the entry fields under userInput. The installation may issue some warnings of the form Automation-Helper class not found for panel ..., which can be safely ignored.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<AutomatedInstallation langpack="eng">
     <com.izforge.izpack.panels.htmlhello.HTMLHelloPanel id="hello" />
```

```
<com.izforge.izpack.panels.htmlinfo.HTMLInfoPanel id="info" />
 <com.izforge.izpack.panels.licence.LicencePanel id="license" />
 <com.izforge.izpack.panels.target.TargetPanel id="target">
   <installpath>/opt/Klaros-Testmanagement</installpath>
 </com.izforge.izpack.panels.target.TargetPanel>
 <com.izforge.izpack.panels.packs.PacksPanel id="packs">
   <pack index="0" name="Java Runtime Environment" selected="true"/>
   <pack index="1" name="Tomcat 9 Application Server" selected="true" />
   <pack index="2" name="Klaros-Testmanagement" selected="true" />
   <pack index="3" name="PDF-Documentation" selected="false" />
 </com.izforge.izpack.panels.packs.PacksPanel>
 <com.izforge.izpack.panels.userinput.UserInputPanel id="portselection">
   <userInput>
     <entry key="tomcat_server_port" value="18005" />
     <entry key="tomcat_http_port" value="18080" />
     <entry key="tomcat_https_port" value="18443" />
     <entry key="tomcat_session_timeout" value="60" />
     <entry key="tomcat_memory_min" value="256" />
     <entry key="tomcat_memory_max" value="1024" />
     <entry key="tomcat_admin_name" value="tcadmin" />
     <entry key="tomcat_admin_password" value="IfYouDontChangeThisDontBlameMe" />
     <entry key="start_server_flag" value="false" />
     <entry key="launch_browser_flag" value="false" />
   </userInput>
 </com.izforge.izpack.panels.userinput.UserInputPanel>
 <com.izforge.izpack.panels.install.InstallPanel id="install" />
 <com.izforge.izpack.panels.process.ProcessPanel id="process" />
 <com.izforge.izpack.panels.shortcut.ShortcutPanel id="shortcuts">
   <createMenuShortcuts>false</createMenuShortcuts>
   <programGroup>Klaros Testmanagement</programGroup>
   <createDesktopShortcuts>false</createDesktopShortcuts>
   <createStartupShortcuts>false</createStartupShortcuts>
 </com.izforge.izpack.panels.shortcut.ShortcutPanel>
 <com.izforge.izpack.panels.finish.FinishPanel id="finish" />
</AutomatedInstallation>
```

Example 3.1. Sample auto-install.xml

3.6. Starting the application

The Klaros-Testmanagement application is executed within a Tomcat application server. It can be started from the command line under Windows with the following call:

<install-dir>/bin/startup.bat

Under Linux this is:

<install-dir>/bin/startup.sh

If the creation of start menu entries or desktop shortcuts was selected during installation, these too can be used to start Klaros-Testmanagement of course.

3.7. Stopping the application

A running Tomcat application server can be stopped under Windows with the following call from the command line:

<install-dir>/bin/shutdown.bat

Under Linux this is:

<install-dir>/bin/shutdown.sh

If the creation of start menu entries or desktop shortcuts was selected during installation, these too can be used to stop Klaros-Testmanagement of course.

3.8. Accessing the application

To access Klaros-Testmanagement, open the following URL in a web browser: http://\${host}: \${port}/klaros-web/ where \${host} is the IP address or domain name of the application server and \${port} is the HTTP port defined in <u>Section 3.3.7, "Step 6: User Data"</u>.



Default URL (local)

Assuming the default installation parameters have been used, Klaros-Testmanagement can be accessed locally using the URL http://localhost:18080/klarosweb/.

3.9. Update Process

Klaros-Testmanagement can easily be updated by just installing the new version. Existing settings and data are automatically retained. Further information on the installation process can be found in <u>Section 3.3, "Installation"</u>. The Klaros-Testmanagement home folder will remain untouched, so all settings, database configuration and the content repository will be unaffected by the update process.



Important

The Tomcat application server must be shutdown before starting the update. Otherwise, the installation process cannot be finished successfully.



Backup your data before updating!

It is highly recommended creating a backup of both the Klaros home folder and your database before starting the update process.

Most database products offer utilities for backing up database schemes, so please refer to the relevant sections in the database vendor documentation for this procedure.

In case you are still using the pre-configured Apache Derby database no further action is required and backing up the Klaros home folder is sufficient.

3.10. Important Files and their Locations

There are two important directories in a Klaros-Testmanagement installation. One is the **installation folder** where the Tomcat application server and the web application will be installed. The other is the **home folder** containing runtime data like configuration files, search indices and attachments. Per default, this folder is named .klaros and located in the home folder of the user running Klaros-Testmanagement.

To allow a simple upgrade the installation/upgrade process will only change files in the installation folder and leaves the content of the home folder untouched.

On Microsoft Windows systems, by default the home folder is located at C:\Users\<username>\.klaros. This can also be addressed by entering %UserProfile%\.klaros in an explorer window address bar or by entering cd %UserProfile%\.klaros on the command line.

On Linux systems this folder can be found using the path ~/.klaros and is generally located at /home/<username>/.klaros.



Relocating the Klaros Home Folder

By setting the environment variable KLAROS-HOME before starting the application the location of the home folder can be moved to another location.

export KLAROS_HOME=/var/application-data/klaros



Heads up When Moving the Klaros Home Folder!

If the application does not find a previous installation in the home folder it will bootstrap an empty default installation including a new local database. If this happens accidentally, e.g. by specifying an invalid Klaros home directory, it seems that the previous data has disappeared.

Of course this is not the case. To resolve this situation point the KLAROS-HOME variable to the correct location and restart.

3.10.1. Log Files

The complete log files for Klaros-Testmanagement are located in the directory logs in the installation directory and also in compressed form in the directory logs in the Klaros home directory.

Attaching these log files to a bug report helps the support team to quickly identify and fix any problems you encounter.

3.10.2. Database Settings

The .klaros/hibernate.properties file is used to specify the database to be used for Klaros-Testmanagement. It defines the driver class and the URI under which the database is accessible as well as the required authentication information. For more information about changing the database settings, see <u>Section 3.11, "Changing the Default Database"</u>.

3.10.3. Language Files

The Klaros-Testmanagement user interface may be customized by adding support for additional languages. This process is described in <u>Section 4.1.1, "Defining Language Files"</u>. The language files can be found under .klaros/resources/messages.

3.10.4. The Quotes File

Klaros-Testmanagement displays a daily changing *quote of the day* on the login page. These quotes are stored in the .klaros/resources/quotes.txt file. For more information, see <u>Section 10.4.4</u>, "Interface" and <u>Section 4.2</u>, "Quote of the Day".

3.10.5. The Derby Database

If you are using the default Apache Derby database, the database files are located in the .klaros/klarosDerby folder.

3.10.6. The Content Repository

All attachments and custom reports are saved in the .klaros/contentRepository folder. If the Klaros-Testmanagement installation and database are migrated to a different location, this folder **must** be moved with them.

3.11. Changing the Default Database

By default Klaros-Testmanagement uses the Apache Derby Database which requires no further installation or configuration but is severely lacking performance in comparison to other databases. For productive use, it is therefore strongly recommended to use a more powerful database system, such as the open source database servers PostgreSQL or MariaDB/MySQL or the commercial Microsoft SQLServer database product.

To switch to another database system, Klaros-Testmanagement must be stopped, and the file hibernate.properties located in <user.home>/.klaros/hibernate.properties must be edited as shown below.

The hibernate.connection.url property must match the location of the database in your network. Please consult your database administrator for the hibernate.connection.username and hibernate.connection.password credentials to use.



Warning

Blank spaces at the end of lines in the hibernate.properties file might get misinterpreted by hibernate! For example, 'hibernate.connection.password=root ' is not the same as 'hibernate.connection.password=root' (ignoring quotes).

An exhaustive list of all parameters can be found in the Hibernate Core Manual.

3.11.1. MariaDB

For MariaDB change the content of the file to:

MariaDB 10.3 or later:

hibernate.connection.driver_class=org.mariadb.jdbc.Driver hibernate.connection.url=jdbc:mariadb://localhost:3306/klaros

```
hibernate.connection.username=root
hibernate.connection.password=root
```

3.11.2. Microsoft SQL-Server

To use a Microsoft SQL-Server change the content of the file to:

Microsoft SQL-Server 2008 or later Versions:

```
hibernate.connection.driver_class=com.microsoft.sqlserver.jdbc.SQLServerDriver
hibernate.connection.url=jdbc:sqlserver://localhost:1433;databaseName=KLAROS
hibernate.connection.username=root
hibernate.connection.password=root
```

3.11.3. MySQL

For MySQL change the content of the file to:

MySQL 5.5 or later versions:

```
hibernate.connection.driver_class=com.mysql.jdbc.Driver
hibernate.connection.url=jdbc:mysql://localhost:3306/klaros
hibernate.connection.username=root
hibernate.connection.password=root
```



Important

When using a MySQL database, it is important to set the following option in the my.ini file.

The maximum size of a query packet the server can handle as well as # maximum query size server can process (Important when working with # large BLOBs). enlarged dynamically, for each connection. max_allowed_packet = 64M

3.11.4. PostgreSQL

To use PostgreSQL change the content of the file to:

PostgreSQL 9.4 or later versions:

```
hibernate.connection.driver_class = org.postgresql.Driver
hibernate.connection.url = jdbc:postgresql://localhost/klaros
hibernate.connection.username=root
hibernate.connection.password=root
```

3.11.5. Apache Derby

To switch back to the integrated Derby database:

```
hibernate.connection.driver_class=org.apache.derby.jdbc.EmbeddedDriver
hibernate.connection.url=jdbc\:derby\:${user.home}/.klaros/klarosDerby;create\=true
hibernate.connection.username=root
hibernate.connection.password=root
```

3.11.6. Setting up the Database Instance



Creating a Database Instance

Klaros-Testmanagement will not automatically create either the database instance (klaros in the above example) or the database user (user root with password root in the above example) in the database server.

Creating a database instance and adding a user is described in the corresponding database manual and will not be covered here. The database user needs permissions to create, drop and alter tables to properly bootstrap the Klaros-Testmanagement database instance.



Make sure to activate support for UTF-8 Character Sets!

Depending on your database product it may be needed to manually activate support for UTF-8 character sets, which may be needed to support languages with uncommon character sets. Especially older versions of MySQL and MariaDB are known for coming with a limited character support in their default configuration. The following example shows how to activate this by editing the mysql.cnf configuration file.

```
character-set-server=utf8
collation-server=utf8_general_ci
```

Example 3.2. Additional entries in mysql.cnf to support UTF-8 Character Sets

3.12. Installing as a System Service

For the long-term use of Klaros-Testmanagement it is essential that the application is available immediately after restarting the computer on which it is installed. Depending on the operating system, different measures have to be taken to achieve this.

3.12.1. Installing as a Linux Service

Depending on your Linux distribution system services are launched using different init daemons.

We are providing sample scripts for the most common ones here, SysVInit and systemd.

3.12.1.1. Installing as a SysVInit Service

To start Klaros-Testmanagement automatically, the startup script shown below should be saved to the /etc/init.d directory as klaros and selected for the appropriate run level. Please refer to the documentation of your linux distribution for details.

The values of the KLAROS_USER and KLAROS_PATH fields must first be replaced with the intended user id and installation path. It is important that the script is executable. This can be ensured with the command **chmod a+x klaros**.

The service can now be started via **service klaros start** and stopped via **service klaros stop**. **chk-config --add klaros** enables the start of the service at system boot.

```
#!/bin/bash
# This will be ignored by systems that don't use chkconfig.
# chkconfig: 345 98 2
# description: klaros (inside Tomcat, a servlet container)
### BEGIN INIT INFO
# Provides:
                  klaros
# Required-Start:
# Required-Stop:
# Default-Start: 3 4 5
# Default-Stop: 0 1 2 6
# Short-Description: klaros
# Description: klaros-testmanagement
### END INIT INFO
KLAROS_USER=klarosdemo
KLAROS_PATH="/opt/klaros-testmanagement"
/bin/su - "${KLAROS_USER}" -c "${KLAROS_PATH}/bin/catalina.sh $@ >/var/tmp/klaros.log 2>&1"
```

3.12.1.2. Installing as a Systemd Service

To start Klaros-Testmanagement automatically the startup script shown below should be saved to the /etc/systemd/system directory as klaros.service.

The values of the User, ExecStart and ExecStop entries must first be replaced with the intended user id and installation path.

The service can now be started via **systemcti start klaros** and stopped via **systemcti stop klaros**. **systemcti enable klaros** enables the start of the service at system boot.

```
[Unit]
Description=Klaros Testmanagement Service
After=syslog.target network.target
[Service]
Environment="JRE_HOME=/opt/klaros-testmanagement/jre"
Environment="PATH=/opt/klaros-testmanagement/jre/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin"
Type=forking
User=klaros
ExecStart=/opt/klaros-testmanagement/bin/startup.sh
ExecStop=/opt/klaros-testmanagement/bin/shutdown.sh
[Install]
WantedBy=multi-user.target
```

3.12.2. Installing as a Windows Service

Under the Windows operating system family the supplied **service.bat** script can be used to install the Tomcat application server as a Windows service.

Please refer to the official Apache Tomcat 9 Windows service HOW-TO for further details.



Changing the Service User Account

Per default, the created Windows service will use the LocalSystem user account to run the Klaros-Testmanagement application. This setting must be changed in the service account settings to the account of the user you intend to use for running Klaros-Testmanagement. This change is needed due to the fact that the default location for the .klaros folder will be located in the user account home.

Skipping this vital step will lead to a newly created .klaros home folder in the C:\-Windows\System32\systemuser folder (this location may vary with your Windows version in use; newer version may use C:\Windows\system32\config\systemprofile instead). The service then uses the newly created, empty database instead of the database that is used when Klaros-Testmanagement is started manually.

If you encounter this situation, then shut down the service, delete the .klaros folder in the Windows\System32 folder, change the user account settings of the service and finally restart the service.



Adjusting Memory Settings and Java Options

Depending on the database size and number of parallel users, Klaros-Testmanagement may require a larger amount of memory than the installer suggests with the provided default values. If you want to change the specified memory settings for an already installed installation you may supply these settings in the **tomcat9w** administration interface as shown below.

In addition, the following Java 9 options must be set to ensure smooth operation of the software:

- --add-opens=java.base/java.lang=ALL-UNNAMED
- --add-opens=java.base/java.io=ALL-UNNAMED
- --add-opens=java.base/java.util=ALL-UNNAMED
- --add-opens=java.base/java.util.concurrent=ALL-UNNAMED
- --add-opens=java.base/java.net=ALL-UNNAMED
- --add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED

To ensure that indeed the settings of the installed service are changed, add the service name as a command line argument as follows: **[path]\Klaros-Testmanagement\bin \Tomcat9w.exe //ES//Tomcat9**.

eneral Log On Logging	Java Startup	Shutdown	
Use default			
Java Virtual Machine:			
C:\Program Files\Eclipse	e Adoptium∖jdk-21.0).3.9-hotspot\bin\se	rvei
Java Classpath:			
C:\Program Files\Apach	e Software Founda	tion\Tomcat 9.0\bin	\bootstra
Java Options:			
L-Diava util logging conf	ig.file=C:\Program	Files Apache Softwa	are Fou
Java 9 Options:			~
Java 9 Options: add-opens=java.bas add-opens=java.bas add-opens=java.bas	e/java.lang=ALL-U e/java.io=ALL-UN e/java.util=ALL-UN e/java.util.concurr	NNAMED IAMED NAMED ent=ALL-UNNAMED	
Java 9 Options: add-opens=java.bas add-opens=java.bas add-opens=java.bas add-opens=java.bas add-opens=java.bas dd-opens=java.bas	e/java.lang=ALL-U e/java.io=ALL-UNN e/java.util=ALL-UN e/java.util.concurra e/java.util.concurra e/java.pet=ALL_UN	NNAMED IAMED NAMED ent=ALL-UNNAMED	
Java 9 Options: add-opens=java.bas add-opens=java.bas add-opens=java.bas add-opens=java.bas add-opens=java.bas Initial memory pool: Maximum memory pool:	e/java.lang=ALL-U e/java.io=ALL-UN e/java.util=ALL-UN e/java.util.concurro fiava.pot=ALL_UN 256	NNAMED IAMED NAMED ent=ALL-UNNAMED MB MB	
Java 9 Options: add-opens=java.bas add-opens=java.bas add-opens=java.bas add-opens=java.bas add-opens=java.bas Initial memory pool: Maximum memory pool: Thread stack size:	e/java.lang=ALL-U e/java.io=ALL-UN e/java.uti=ALL-UN e/java.uti=ALL-UN 256 256	NNAMED IAMED NAMED ent=ALL-UNNAMED MB MB KB	



Additional Settings

The content repository (see <u>Section 3.10.6, "The Content Repository"</u>) requires another setting in the **tomcat9w** administration interface. Here the parameter **-Doak.locksupport=disabled** must be added.

	e runicaci	9.0 TOILC	ica Prope	rues				\sim
General	Log On	Logging	Java	Startup	Shutdown	1		
0 Us	se default)							
Java V	/irtual Mac	hine:						
C:\U	sers\stolp\	Klaros-Te	stmanage	ement\jre\	bin\server\	jvm.dll		
Java (Classpath:							
C:\U	sers\stolp\	Klaros-Te	stmanage	ement\bin\	bootstrap.;	jar;C:\U	sers\stolp\]
Java (Options:							
-Djav -Djav -Doa Java 9	va.util.logg va.util.logg k.locksupp 9 Options: 1-opens=i/	jing.mana jing.config ort=disab ava.base/	ger=org, g,file=C:\ iled java.lang java.lang	apache, ju Users\stol]=ALL-UNI ALL-UNI	II.ClassLoad p\Klaros-Te NAMED MED	ierLogMa stmanac	anage gemen	
ado ado ado	d-opens=j d-opens=j	ava.rmi/su	un.rmi.tra	Insport=A	LL-UNNAME	.0		
ado ado ado Initial	d-opens=j; d-opens=j; memory p	ava.rmi/su	un.rmi.tra 512	insport=A		 	~	
ado ado ado Initial Maxim	d-opens=j; d-opens=j; memory p num memor	ool: [un.rmi.tra 512 1024	Insport=A		 МВ МВ	~	
ado ado ado Initial Maxim Threa	d-opens=j. d-opens=j. memory p num memor d stack siz	pool: [vy pool: [e: [512 1024	nsport=A		МВ МВ КВ	~]

3.13. Monitoring

Using the built-in JMX Support in Tomcat it is possible to monitor the Klaros-Testmanagement installation using any JMX capable Monitoring tool (like e.g. VisualVM).

To do this, Add the following lines to Klaros-Testmanagement/bin/catalina.sh:

```
# Set JMX connector
RMI_SERVER='[FQDN]'
JMXREMOTE="-Dcom.sun.management.jmxremote"
JMXREMOTE="$JMXREMOTE -Dcom.sun.management.jmxremote.port=[PORT]"
JMXREMOTE="$JMXREMOTE -Dcom.sun.management.jmxremote.ssl=false"
JMXREMOTE="$JMXREMOTE -Dcom.sun.management.jmxremote.authenticate=false"
JMXREMOTE="$JMXREMOTE -Dcom.sun.management.jmxremote.ssl=false"
JMXREMOTE="$JMXREMOTE -Dcom.sun.management.jmxremote.ssl=false"
JAVA_OPTS="$JAVA_OPTS $JMXREMOTE -Djava.rmi.server.hostname=$RMI_SERVER"
```

Where [FQDN] and [PORT] denote the fully qualified domain name and port of the remote JMX server you want to deliver messages to.

3.14. Configuring External Issue-Management-Systems

Depending on the type of issue management system, some additional configuration may be needed to connect it to Klaros-Testmanagement. This section describes the needed configuration steps for each system.

Name	Version
Bugzilla	3.0.9 or later
GitHub	Any
GitLab	API V3/V4
Jira	3.7 or later
Mantis	1.1.8 or later
Redmine	3.0 or later
Trac	0.10 or later
YouTrack	2021.3 or later

Table 3.3. Supported External Issue-Management-Systems

3.14.1. Jira Configuration

For current Jira versions, no further configuration actions are necessary.



Jira Cloud Instances

Starting April 25th 2019 Atlassian limited access to their cloud instances to token based authentication, so authenticating via username and password is no longer supported.

Klaros-Testmanagement is supporting token based authentication scheme as well. Just use your email address as username and the token as password.

3.14.1.1. Jira Legacy Server (Version 4.2 or lower)

To allow the connection to a Jira legacy server instance the Jira RPC Plugin has to be activated in your Jira installation. You find this option as *Accept remote API calls* in **General Configuration** under **Global Settings**. Then you need to enable the JIRA RPC Plugin in **Plugins** under **System** in the left-hand menu.

3.14.2. Redmine Configuration

n	_				
ľ	Ξ	-	-		
ľ.	_	-	-	- 1	

Enable REST Web Service

In order to connect to a Redmine installation, the *Authentication Required* and the *Enable REST Web Service* checkbox in the Authentication section of the Redmine administration settings must be activated (<u>Figure 3.15, "The Redmine Authentication</u> <u>Section"</u>).

Home My page Projects Administration Help
Redmine
✓ Successful update.
Settings
General Display Authentication API Projects Users
Enable REST web service 🕑
Enable JSONP support
Save

Figure 3.15. The Redmine Authentication Section

3.14.3. Trac Configuration

The Trac integration requires the installation of the TracXMLRPC Plugin. The download archives and installation notes are available at http://trac-hacks.org/wiki/XmlRpcPlugin/. After successful installation of the plugin, it has to be activated using the Trac web administration interface. Figure 3.16, "The Trac Plugin Section" shows the Trac Plugin section after the successful installation of the plugin.

Administration	Manage Plugins (2)	
General Basic Settings Logging	Install Plugin: File: [Date: auxwahen] Keine Datei ausgewählt. [Install] Upload a plugin packaged as Python epg.	
Permissions Plugins	> Trac 1.4.2	
Ticket System Components Milestones	Integrated SCM, wiki, insue tracker and project environment	
Priorities Resolutions Severities Ticket Types Versions	TracXMLRPC 1.1.9 RPC interface to Trac Autor: Aker Thomas Munitalian: C dd Smort Browsen	
Version Control Repositories	Home page: http://www.wmeporugin Liene: BS	Enabled
	▶ tracrpc.spi.* - License: BSD	
	XHLRBCSystem - Core of the RPC system.	
	▶ tracrpc.json_rpc.* = License: BSD	
	▶ JsonRportcocol - Example 'POST' request using 'curl' with 'Content-Type' header and body:	
	tracrpo.ml_rpo.* - License: BSD	
	▶ XmlRpcProtocol — There should be XML-RPC client implementations available for all popular programming languages. Example call using `curl':	
		Apply changes

Figure 3.16. The Trac Plugin Section

To make the plugin accessible, authenticated users must be able to access the XML_RPC plugin.

	amani					Sec
 Integrated SCH & Project Malia 	peners				logged in as user Logo	ut Preferences Help/Guide About T
				WIKI TIMELINE	ROADMAP VIEW TICKETS	NEW TICKET SEARCH ADMI
The subject authenticated I	ias been granted th	e permission XML_RPC.				do
Administration	Manage P	ermissions and G	roups			
Seneral Basic Settings Logging	Grant Permis Action: XML	sion: RPC V	ubject: authenticated	Add Grant permi	ssion for an action to a subject, w	which can be either a user or a group.
Permissions Plugins	Copy Permise Subject:	sions:	et	Add Copy all of subje	ct's permissions to target. Subjec	at and target can be either users or groups.
icket System Components Milestones Priorities Resolutions	Add Subject 1 Subject:	o Group:	ıp:	Add Add a user or gro	up to an existing permission grou	up.
Severities Ticket Types Versions	Permissions					
ersion Control	Subject	Action				
Repositories	anonymous	BROWSER_VIEW REPORT_SQL_VIEW TIMELINE_VIEW	CHANGESET_VIEW REPORT_VIEW WIKI_VIEW	FILE_VIEW ROADMAP_VIEW	LOG_VIEW SEARCH_VIEW	MILESTONE_VIEW TICKET_VIEW
	authenticated	TICKET_CREATE	TICKET_MODIFY	WIKI_CREATE	WIKI_MODIFY	XML_RPC

Figure 3.17. The Trac Permission Section

The following combinations have been successfully tested:

• Trac 0.12 / TracXMLRPC 1.1.0

Later versions are expected to work but currently not tested.

3.15. SSL Support

Note on self-signed certificates: If you are accessing HTTPS servers with self-signed certificates, you will need to import those certificates into your Java trusted keystore. One instruction on how to achieve this can be found at http://stackoverflow.com/questions/2893819/telling-java-to-accept-self-signed-ssl-certificate.

3.16. Upgrading from Version 4

This section describes important changes for users which are upgrading to version 5 from version 4. Please read this fully before attempting an upgrade. If you have questions about the upgrade process, please contact us at support@verit.de.

3.16.1. Required Version for Upgrade

For a successful upgrade to version 5, it is mandatory that the installed Klaros-Testmanagement version is 4.12.0 (released July 29th, 2019) or higher. If this is not the case, the application will refuse to start, and you will see a corresponding entry in the console and log. In this case, please download the latest available 4.xx version first, install it and update your installation by logging in with administrator credentials. After this, the upgrade to version 5 is possible.

3.16.2. Unattended Database Upgrade

In version 4 it was mandatory that an administrator triggered a database update after a new version has been installed. In version 5 this is no longer required, the system is now fully usable directly after an update has been made.

3.16.3. Startup Time for Initial Upgrade to Version 5

Version 5 rebuilds existing databases from previous installations on the first start. Depending on the size of the existing data and the processing power of your database installation this process can take from some minutes to up to an hour.

Please reserve a reasonable time frame for this and show some patience during startup. Each step of the rebuild process is logged on the console and/or logfile.

3.16.4. Embedded Java Runtime Environment

In contrast to previous versions it is no longer required to install a Java Runtime Environment prior to installing Klaros-Testmanagement. Klaros-Testmanagement is now packaged with a Java 11 runtime that is automatically used.

3.16.5. Automatic Database Version Detection

As of version 5, the database version used is automatically detected and should no longer be specified in the <user.home>/.klaros/hibernate.properties file. Please remove the hibernate. dialect=... line from this file before you perform the upgrade.

3.16.6. Oracle Database no Longer Supported

In version 5 we have discontinued support for Oracle databases. If you have an existing Oracle installation you like to migrate to another DBMS please contact us at support@verit.de for further instructions.

3.17. Uninstall

To uninstall Klaros-Testmanagement select *Klaros-Testmanagement Server Uninstaller* from the start menu (if running under Windows) or just delete the installation folder manually.

If you have added an installation as a service as described in <u>Section 3.12, "Installing as a System</u> <u>Service"</u> please remember to deactivate the Windows service or Linux init script.







The Home Folder

Just like during the upgrade process uninstalling will not touch your home folder (<user.home>/.klaros).

So if you uninstalled by accident re-installing will let you restore your system safely, else remove feel free to remove that folder manually.

Chapter 4. Customization

This chapter shows how to customize the displayed language and the file for the "Quote of the Day".

4.1. Languages

Klaros-Testmanagement comes with language files in German and English. Support for additional languages can be added by using additional language files if needed. The corresponding language file must be stored in the .klaros/resources/messages directory for this.

Language files are activated in the application under Configure / System / Languages. Detailed information can be found under <u>Section 10.4.5.1, "Enabling and Disabling Languages"</u>.

4.1.1. Defining Language Files

Adding a valid language file to the .klaros/resources/messages directory causes the language to become available in Klaros-Testmanagement. A valid language file has a name of the form messages_\${language}_\${COUNTRY}_\${variation}.properties where \${language} is the two-letter ISO 639-1 code for the language in lower-case, \${COUNTRY} is the two-letter ISO 3166 code for the country in upper-case and \${variation} is an optional, user-definable code for the variation of the language (e.g. dialect).



Language File Encoding

The language file must use ISO-8859-1 encoding with Unicode escapes (\uXXXX).

If you prefer editing these files as UTF-8 please use a converter like **native2ascii** or **iconv** to translate them to a proper format.

After Klaros-Testmanagement is run for the first time, the American English, German and fallback language files will be copied to the .klaros/resources/messages folder. In order to create a new language file, the text should be copied out of one of these files into an appropriately named file. The contents of the file consist of key value pairs in the form: key=This sentence is the value. None of the keys should be altered. The values (the text after the '=') should be translated into the target language.

When Klaros-Testmanagement is started, the default language files (messages.properties, messages_en_US.properties and messages_de_DE.properties) are copied to the .klaros/re-sources/messages folder, overwriting any changes made to these files. Therefore, it is advised that any user changes to the American English and German interfaces be made in a file with the variation in the filename set, e.g.messages_en_US_companyName.properties in order to persist changes.

As well as copying over the default files, Klaros-Testmanagement also merges any changes with user-defined files. If key-value pairs are added to the default files during an update, these will also be added to the end of any user-defined language files, along with a comment stating when the entries were added. These entries should be translated by the creator of the custom language file.

4.2. Quote of the Day

Klaros-Testmanagement displays a new "Quote of the day" on the login screen every day. The default quotes can be extended by own entries or deactivated. The quote file can be uploaded and activated in Klaros-Testmanagement on the page *General Settings* directly into the folder . klaros/resources in the user's home directory.

This file is a .txt document in ISO-8859-1 encoding and uses one line per quote. Section <u>Section 10.4</u>, "System" describes how to install this file. The respective quotes change daily.

Chapter 5. Functional Overview

This chapter introduces the user interface and its basic functions.

5.1. Login

<u>Figure 5.1</u> shows the login screen of Klaros-Testmanagement. To login enter the username in the *Username* field and the associated password into the *Password* field.



Figure 5.1. The "Login" Screen

Three user accounts are predefined in the database with the following roles:

Administrator	Username: admin / Password: admin
Manager	Username: manager / Password: manager
Tester	Username: tester / Password: tester

Confirm the login to Klaros-Testmanagement by clicking the Log In button.

For a description of the users roles and the permissions associated with each role, see <u>Appendix A</u>, <u>Access Permissions</u>.

enterprise edition Only available in Klaros-Testmanagement Enterprise Edition

The Klaros-Testmanagement Enterprise Edition also supports authentication using an external directory service (LDAP / Active Directory). For more information about configuring an external directory service, see <u>Section 10.5.5</u>, "LDAP".



Automated User Creation Option

If a user is authenticated for the first time using an external directory service, a user account with the *Tester* role can be created automatically for this user. The LDAP attributes for the username and the e-mail address are automatically transferred to the newly created user account in this case.



Setting the User Interface Language at Login

By default, the user interface is displayed in the language specified by the localization setting of the server operating system. Clicking on one of the flags to the right above the *username* text box also allows you to select a different language for the user interface.

After a successful login the Projects page is shown (Figure 5.2).

≡ 📌 K L A R	o s	TEST	MANAGEMENT				Financ	e Tracker 📑	8	۹ ×	0~ 1 ~
📝 Define	0	13:56:32 L	ogin successful, welcome to Klaros-Te	stmanagement.							+ ×
北 Plan	Pro	New								Save	Discard
🏟 Execute				9 Entries - Page 1	1 of 1 🖌 ┥	1 🕨 H	10 🗸	.		A a	how all 쇼 =
🕒 Evaluate	0	P00011	Description =	iterations + 0	Requirements = 0	Test Cases 🖶	Test Suites ↓ 0	Test Runs ₩ 0	about a year ago	Last Access ₹ 2 months ago	
Le Configure	0	P00010 P00009	Test PRV PRV 2	0	0	5	1	3	about a year ago about a year ago	about a year ago about a year ago	୦ ໕ 🖨 🛍 ୦ ໕ 🖨 🛍
Configure	0	P00008	PRV	0	0	3	1	1	about a year ago	about a year ago	
	0	P00007 P00005	Issue Management Integration DE Finanz-Tracker	9	6	4 24	0 7	0 8800	2 years ago 3 years ago	about a year ago 14 minutes ago	○໕╤╙
	0	P00003 8	Printer Tester	0	4	1	0	0	3 years ago	5 minutes ago	○ C° 🖨 û
	0	P00002 P00001	Finance Tracker FinanceTrackerLocalTester	9	6	24 26	7 9	338 204	3 years ago 3 years ago	less than a minute ago 12 minutes ago	⊙Ưᇢ⋓ ◯ぴᇢᄈ
		ID	Description	Iterations	Requirements	Test Cases	Test Suites	Test Runs	Created	Last Access	Action
		New								Save	Discard

Figure 5.2. The " Projects" Page

5.2. Page Overview

The individual elements of a screen page are presented below.

5.2.1. Side Menu

The entries in the menu in the left part of the screen are based on the activities in a test project. The individual sections are:

Define

In the *Define* section, all objects required for a test project are created, edited and managed. The objects managed here include the project itself, test environments, systems under test, test cases with their individual test steps, and test suites. In

	Functional Overview
	the Klaros-Testmanagement Enterprise Edition additional ob- jects are iterations, requirements and test segments.
Plan	In the <i>Plan</i> section, jobs for executing or reviewing tests are created and managed. Tasks, can be assigned to individual users, and their progress can be tracked in detail. This section is only included in the Klaros-Testmanagement Enterprise Edi- tion.
Execute	In the <i>Execute</i> section, jobs, manual test cases and test suites will be executed. Before a test can be executed, a system un- der test and a test environment must be created and select- ed. Klaros-Testmanagement guides the tester step by step through the manual test and automatically logs the results. Results and comments can be added to each test step. Tests can be interrupted and continued at a later time.
Evaluate	In the <i>Evaluate</i> section, a configurable dashboard and various reports provide an overview of the status of the tests. Both overview reports and fine-grained information are available so that each individual test execution can be tracked and visual- ized.
Configure	The <i>Configure</i> section is intended for administrative tasks, such as system settings, user management, creating report templates, integrating issue management systems, require- ments management, LDAP and the like. Data backup and the import and export of projects are also performed here.

The following chapters will describe each category in detail.

5.2.2. Page Header

5.2.2.1. Search / Quick-Select

Depending on the Klaros-Testmanagement edition, you will find either a search box in the top right corner of the page (see <u>Figure 5.3</u>) for quick navigation to objects with a specific ID or for a full text search in the currently selected project.





5.2.2.1.1. Full Text Search



The Search field allows the user to search for object ids or words or phrases contained in fields of objects. This search function uses Apache Lucene, which provides a powerful syntax for searching individual fields or with wildcards. This syntax is described on the Apache Lucene website.



Figure 5.4. The "Search Results" Page

<u>Figure 5.4</u> displays the *Search-Results* page. This page displays all occurrences per object type in a separate view. The search term is highlighted in color.

The status line displays the time needed for the search and the number of hits, see. Figure 5.5

```
♀ 14:07:37 The search finished successfully in 0.1 seconds returning 65 hits in 29 results
```

+ ×

Figure 5.5. The Status Bar

5.2.2.1.2. Quick-Navigation

The full text search is only included in Klaros-Testmanagement Enterprise Edition. In the Klaros-Testmanagement Community Edition instead of the full text search a quick navigation field for known element IDs (like TC00001) is available. The following object types are supported there:

- Project (P)
- Test Environment (ENV)
- System under Test (SUT)
- Test Case (TC)
- Test Case Result (TCR)
- Test Suite Result (TSR)
- Test Run (TR)
- Issue-Management-System (IMS)

Please note, that ids will be searched for in the current project only.

0000000

Shortcuts in Quick Navigation

The quick navigation is case-insensitive and can also be abbreviated by omitting leading zeros. So **TC00001**, **TC1** or **tc1** all refer to the same element.

5.2.2.2. The Log Panel



Figure 5.6. The "Log Panel" Icon

The log panel displays status messages such as warnings or information. By default, only the last status message is displayed. Clicking the + icon on the right side of the log panel displays the status messages since the last login. Clicking the \times icon closes the log panel.

۲ ۵	14:20:51 The search finished successfully in 0.1 seconds returning 65 hits in 29 results	- ×
Ŷ	INFORMATION 14:20:51 The search finished successfully in 0.1 seconds returning 65 hits in 29 results	
Ŷ	INFORMATION 14:20:51 The search finished successfully in 0.2 seconds returning 65 hits in 29 results	
Ŷ	INFORMATION 14:20:47 The current selected project is: Finance Tracker (P00002)	
Ŷ	INFORMATION 14:20:42 The current selected project is: Printer Tester (P00003)	



5.2.2.3. The Help Menu





The help menu provides quick links to various Klaros-Testmanagement resources. Pressing the help menu icon ② opens a menu with the following links:

Documentation	Opens the relevant page in the manual.
Tutorial	Opens up the Klaros-Testmanagement tutorial.
Message Boards	Opens up the Klaros-Testmanagement message boards.
Contact Support	Prepares an e-mail to be sent to the Klaros-Testmanagement support team. This option can be used when experiencing an issue using Klaros-Testmanagement which couldn't be solved using the documentation or the message boards.

5.2.2.4. The User Menu



Figure 5.9. The "User Menu" Button

The user menu icon on the upper right of Klaros-Testmanagement opens the user menu.



Figure 5.10. The opened user menu

The name and role of the currently logged-in user is displayed here. It is possible to change the user interface language and log out of the system.

5.2.3. Content

The content of a page changes due to navigation via the menu bar or other actions.

The pages generally have the same structure. The following two basic patterns exist: overview pages and detail pages

5.2.3.1. Overview Page

On an overview page several objects of the same type (e.g. test cases) are displayed and managed in a table.

5.2.3.1.1. Quick Table Filtering

enterprise Only available in Klaros-Testmanagement Enterprise Edition

On selected tables it is also possible to filter the listed entries by full text search as described in <u>Section 5.2.2.1.1, "Full Text Search"</u>.

By entering a search term in the input field above the table, the text fields of the displayed objects are searched for the specified search term and the list is filtered accordingly.

5.2.3.1.2. Filtering and Sorting

The objects displayed in a table can be filtered and sorted by their attributes. This includes attributes that are not directly visible in a list column. By clicking on the *Filter/Sort* \Im icon above the table, a panel with two further tables opens. The left table contains the active filters, while the right table contains the active sort criteria. If a filter is used, the icon is marked with an orange color.



Figure 5.11. Number of Active Filter in Use

OS TESTI	MANA	SEMENT				Finance Tracker			_	٩	× 0~ ≛~
Test Cases											
New									S	ave	Discard
Filter / Sort											
Fie	ld	Criterion	Value	Action		Order by				Dire	ction Action
Channed		M Faula M	07.06.2022.12:52 #	m	Changed	older by				1 4 4 4 4 4	ling M III
Changed		• Equais •	07.00.2023 13.33	<u> </u>	changed					Ascen	ning • 🖂
New					New						
								Apply	Res	set	Close
+ 0028	00		24 Entries - Page 1 of 3	M ┥ 🚺 2 3	▶ N 10 ∨			유			Q × & ≡
ID ● Q	Revision		ame 🗢		Traceability 🕈	Priority 🗘	Status 🗘	Execution © 3	Steps 🗘	Issues 🗘	Action
TC00024	1.0 Cł	reck the connection to the serve	ar	Requirement 1.3	2.4 in testplan_16.txt	High	Draft	Automated	0	2/1	CCQů
TC00023	1.0 Cł	nange a standing order.		Requirement 1.3	2.4 in testplan_16.txt	Medium	Draft	Manual	5	0/2	CO D 🖞
TC00022	1.0 Up	adate dashboard by adding a sta	anding order.	Requirement 1.3	2.4 in testplan_16.txt	Medium	Draft	Manual	6	5/0	2040
TC00021	1.0 Up	adate dashboard by cancelling a	a standing order.	Requirement 1.3	2.4 in testplan_16.txt	Medium	Draft	Manual	6	0/2	C C & Ö
TC00020	1.0 Up	odate dashboard by cancelling a	a debit mandate.	Requirement 1.3	2.4 in testplan_16.txt	Medium	Draft	Manual	6	0/0	2000
TC00019	1.0 Co	onnection to the database.		de.verit.finance	tracker.web.Connection	High	Draft	Automated	0	0/0	2000
TC00018	1.0 Co	innection to the server of the ba	ank.	de.verit.finance	tracker.web.Connection	High	Draft	Automated	0	0/0	CODÓ
TC00017	1.0 De	Hete debit mandate.		Requirement 1.3	2.4 in testplan_16.txt	Medium	Draft	Manual	5	0/0	CODO
TC00016	1.0 De	slete standing orders.		Requirement 1.3	2.4 in testplan_16.txt	Medium	Draft	Manual	5	0/0	0000
TC00015	1.0 Ac	Jd a standing order.		Requirement 1.3	2.4 in testplan_16.txt	Medium	Draft	Manual	5	0/0	2040
ID	Revision		Name		Traceability		Status	Execution	Steps	Issues	Action

Figure 5.12. The Filter and Sort Panel

Below the two tables there are three buttons:

- Apply applies the filters and orders to the table below.
 - Reset deletes all filters and sorting entries.
- Close closes the panel.

Specifying Filtering Criteria

Each row in the filter table represents a filter which is applied to the table.

Clicking the New button adds a new, empty filter to the filter table. Each row has four columns:

- The *Field* column specifies the field to filter by. The fields that can be filtered by vary from object to object.
- The *Criterion* column defines the restriction type to be applied during filtering. The restriction types that can be selected in an entry will vary from field to field.
- The *Value* column specifies the value of a constraint, e.g. the user by which a field should be filtered.
- The Action column contains a button for deleting the filter criterion.



Multiple Selection

For some entries, such as users, multiple selection is also possible. To do this, hold down the CTRL key and select the desired entries individually.

		Action		
eated By 🗙 Equals 🗸	0	Û	No entries available	
			New	
		^		
	Felix Mustermann			
	Günther Gross			
	Oliver Krams			
	Patrick Reilly			
	Cohrise Cidler	~		
New				


Specifying Sorting Criteria

Table entries can be sorted by multiple criteria. The sorting criteria are specified in the right table in the *Filter/Sort* panel.

It is possible to specify more than one sorting option. In this case, the top row of the table has the highest priority and the bottom row has the lowest priority.

Clicking the New button creates a new empty row for a sort criterion is created at the bottom of the table. To remove the row, use the Delete in icon in the Action column.

The sorting options table contains three columns:

- The Order by column indicates the field of the listed objects to sort by.
- The Direction column defines the direction of the sort order, i.e. descending or ascending.
- The find icon in the Action column removes the individual sort criteria.

5.2.3.1.3. Categorization



Only available in Klaros-Testmanagement Enterprise Edition

To manage a large number of objects, they can be categorized according to any criteria. This function is available for the following objects: *Iterations, Requirements, Systems under Test, Test Environments, Test Cases* and *Test Suites*. The categorization panel is opened by clicking the *Categories* icon $\frac{1}{24}$ above the table.

In the Categorization panel, *Category groups* and *Categories* can be created. Each category group contains any number of categories that can be arranged in a tree structure. Objects can be assigned to any category in each category group. The number of categories or category groups is not limited.

In order to apply categorization, the *Categories* panel must be open. It contains two different views: *Edit* and *Tree*. If no categories have been defined, only the edit view can be used. Each view allows selection of the active category group through the use of a dropdown menu, or selection of the blank category group to disable categorization.

Edit categories

≡ 👱 ।	KLAR	OS TEST	MANAG	BEMENT			Finance Tracker				Q :	× 0~ ±~
📝 Define		Test Cases										
📇 Plan		New								S	ave	Discard
🏟 Execute		Category Group Name	Action Group Action Group	✓ + 前								∷ ⊄
🕒 Evaluat		V III Action	late			+ 2	1) 2 11					
🔑 Configu		II Del	ete t			+ 0	20					
		+ 🕫 🗅 😕 (8 ش 🗅	4 Entries - Pag	e1of1 🕅 ┥ 🤨 🕨 🕽	4 10 ¥			Save	Disc	ard	Close
		ID ♦ 🛛 🖓	Revision	Name 🖨	Traceability	\$	Priority 🖨	Status 🖨	Execution 🗢	Steps 🗢	Issues 🖨	Action
		TC00024	A 1.0	Check the connection to the server	Requirement 1.2.4 in testpl	an_16.txt	High	Draft	Automated	0	2/1	CCQÛ
		TC00023	1.0	Change a standing order.	Requirement 1.2.4 in testpl	an_16.txt	Medium	Draft	Manual	5	0/2	ぴ∪₿₫
		TC00019	1.0	Connection to the database.	de.verit.financetracker.web	Connection	High	Draft	Automated	0	0 / 0	CCQ
		TC00018	1.0	Connection to the server of the bank.	de.verit.financetracker.web	Connection	High	Draft	Automated	0	0/0	┏┍ҿ╓
		ID	Revision	Name	Traceabili	ty	Priority	Status	Execution	Steps	Issues	Action
		New								S	ave	Discard

Figure 5.14. The "Categorization Edit" View

This view allows creating and editing category groups and categories as well as organizing category hierarchies. Clicking the + icon at the top of this view creates a new category group.

Categories can also be added, edited and deleted in this view. A category group always contains a root category that cannot be deleted. By clicking on the + a new subcategory is added to a category. Clicking on the \widehat{m} will delete a category.

Categories may be renamed by changing the value in the name field and also given a description through use of the \square edit button.

Changing the order and hierarchy of categories is also possible via drag and drop.

≡ 📌 K L A R	os test	MANAG	EMENT		Finance Tracker				Q X	0 ~ ≗ ~
📝 Define	Test Cases									
北 Plan	New Categorization]						Save	9	Discard
🕸 Execute	Action Group	24)	~							≣ Ø
🕒 Evaluate	Update Delete ((4/4) (2/2)								
🖌 Configure	Edit (14	/14)								
	4 🗆 🖄 +	8 m 🗅	4 Entries - Page 1 o	of1 🕅 🛋 🚺 🕨 🕅 10 🗸			토 기 Show all			Close
	. ID≑ Ø	+ Revision	Name 🖨	Traceability 🗢	Priority 🖨	Status 🖨	Execution 🗢	Steps 🗢 Is	sues 🖨	Action
	TC00024	▲ 1.0	Check the connection to the server	Requirement 1.2.4 in testplan_16.txt	High	Draft	Automated	0	2/1	C C 🖯 🛈
	TC00023	1.0	Change a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft	Manual	5	0/2	CCQ
	TC00019	1.0	Connection to the database.	de.verit.financetracker.web.Connection	High	Draft	Automated	0	0/0	COAD
	TC00018	1.0	Connection to the server of the bank.	de.verit.financetracker.web.Connection	High	Draft	Automated	0	0/0	C L A U
	ID	Revision	Name	Iraceability	Priority	Status	Execution	Steps	ssues	Action
	New							Save	Э	Discard

The Tree View



This view shows an overview of the category structure. The available categories are represented as a collapsible tree, with the name of each category is shown, along with the number of objects it contains. The description of each category is shown as a tooltip when its name is hovered over with the mouse pointer.

Assigning objects to categories

To assign objects to categories in a table, the *Categories* panel must be open and the *Tree* view must be enabled in it.

To assign objects first activate the category containing the objects that should be assigned and select the objects. Then click on the folder icon \Box above the table. A dialog will appear. Choose the appropriate category to which you want to assign the objects and click the Assign button. The selected objects are now assigned to the chosen category.

Please note, that you can not assign objects from multiple categories to a single category at once. In case that the desired objects belong to different categories you must repeat the step for each category.

5.2.3.1.4. Display of Deleted Objects

If an object is deleted, it gets a deletion marker and is no longer displayed on the overview page. Administrators can view deleted objects by clicking the Show all button. Clicking on the button Only active will hide them again. This function is available for the following objects: *Iterations, Jobs, Projects, Requirements, Test Cases, Test Environments, Test Runs, Test Segments, Test Suites, Test Systems* and *Users*.



Automatic Saving of the Setting

The selection whether deleted objects should be displayed is automatically saved for each overview page and administrator.

5.2.3.1.5. Bulk Actions



In tables, bulk actions are often also available for the displayed objects. This means that several entries in the list can be selected and processed together. This applies, for example, to deleting, duplicating, editing or assigning to categories.

In order to use a bulk action, the user must first select one or more objects in the list using the check-boxes to the left of the table. The buttons above and to the left of the table then become active. <u>Figure 5.16</u> shows the bulk action buttons on the test case overview page.

Test Cases							
New						Save	Discard
+៤០រទ្យុក	24 Entries - Page 1 c	of 1 📕	▲ 1 ►	M 40 ∨	品⑦ Show a	11	Q × & Ξ
ID 🗘 🖓 Revision	Name 🖨		Traceability 🖨	Priority 🖨	Status 🖨 Execution 🖨	Steps 🖨 Issues 🖨	Action



Before a mass action can be executed, all changes in the table must be saved. If there are still unsaved changes, a dialog will appear.

A confirmation dialog is displayed once a bulk action icon has been clicked. These dialogs describe the action which will be carried out and sometimes contain input fields, e.g. to enter the revision comment for the bulk new revision action. All changes are written to the database as soon as this dialog is confirmed.

🔳 👱 K L A R	o s test management	Finance Tracker 🖬 🖹 🚺 🔍 🕹 🗸
📝 Define	Systems under Test	
Plan	New	Save Discard
🏟 Execute	ID ◆ Q 0 Version ◆	10 V At a state of the state o
	✓ SUT00006 Finance Tracker 2.1.0 ✓ SUT00005 Finance Tracker 2.0.1	1/2 2 2 日前
Lvaluate	✓ SUT00004 Finance Tracker 2.0.0 ✓ SUT00003 Finance Tracker 1.1.1	
De Configure	✓ SU100002 Finance Tracker 1.1.0 ✓ SUT00001 Finance Tracker 1.0.0	0/0 译合世 0/0 译合前
	New	Save Discard
	Bulk edit	
	Version	
	6 SUTs will be edited	
		Next Cancel

Figure 5.17. Bulk Editing Objects

5.2.3.2. Detail Page

A single object is displayed and managed on a detail page.

5.2.3.2.1. Print Pages



Objects can be displayed in a print-optimized representation on their own page by clicking on the icon. The icon can be found in the *Action* column in the table, in the list of mass operations at the top left of tables and as a large icon at the top right of *Detail pages*.

The display can be modified by the following parameters.

- The Suppress empty fields option hides fields content in the view.
- If the *Display diagrams* option is selected, diagrams are embedded in the print preview.
- With *Test environments* and *Systems under Test* the test environments and test systems can be selected whose results should be displayed on the page.
- The *Details* setting determines how detailed the data is displayed.

The selected print options will be preserved during a user session.

Figure 5.18 shows the print view for a test suite.



Browser print settings

The option *Print Backgrounds and Images* must be activated to achieve the best output quality.





5.2.3.2.2. Bookmarks

The detail page of any object can be accessed directly via a special URL generated by the system. By clicking on the \square icon this URL to the current page is copied to the clipboard and can be shared with any other application.

≡ 👱	FinanceTracker Selenide	-Tests 🖬 📕 🔍 😲 V 💄 V
🗹 Define	TC00022 - Update dashboard by adding a standing order.	◈읍♫Ġᢒ
an Plan	Overview Properties User Defined Steps (6) Attachments (1) Revisions Issues (1)	Save Discard Back Jobs (1) Results (8) Changes
🏚 Execute	Status Draft Execution Manual	
🕒 Evaluate	Priority Medium Latest Executor Felix Mustermann	
🗲 Configure	System under Test FinanceTracker 2.1.0 × × Test Environment Android 4.5 Smartphone × ×	

Figure 5.19. Creating Bookmarks

<u>Section 6.1.2.5.1, "Assigning Project Specific Roles"</u> explains the configuration of controlling access to bookmarked pages.

5.2.3.2.3. Overview



For every type of object in Klaros-Testmanagement there is a dedicated overview tab which shows the most important data for this object at a glance. In addition to plain data, many overview pages contain various tables and diagrams, like for example the success history for test suites, which helps to visualize the test progress.



Figure 5.20. The "Overview" Tab

5.2.3.2.4. User Defined Properties



Only available in Klaros-Testmanagement Enterprise Edition

To customize projects to specific requirements, objects can be extended with user-defined fields.

These are created separately for each project in the *Project* section of the *User-Defined* view, see Figure 5.21.



Figure 5.21. The "User Defined" Tab

Pressing the New button adds a new user defined property to the list. After adding a new property, the field type must be chosen.

User-defined fields can be created for the following objects, these are each managed in their own tab: Test Case, Test Step, Test Segment, Test Suite, Test Environment, Test System, Test Run, Requirement, Iteration.

The four possible property types are:

Text	The property will be a single-line text field.
Text Field	The property will be a multi-line text field.
Boolean	The property will be a check box.
Enumeration	The property will be a drop-down box to select a single entry from a predefined list of values.

The default value of the property can also be set here.

After the new property has been saved, the property type can no longer be changed. The name of the property and - if available - the list of enumeration values can be changed at any time.

Enumeration Values

The \square icon next to the enumeration name opens the dialog for editing the enumeration values, see <u>Figure 5.22</u>.

= 👱 K L A R	o s test management	Finance Tracker 👕 🗮	५ × 0 × ≛ ×
📝 Define	P00002 - Finance Tracker		🖨 😋 ᢒ
Plan 😫	Properties User Defined (6) Copy Objects Access Integration Results Change	Save	Discard Back
Execute	Iteration Job Requirement System under Test (1) Test Case (1) Test Case Re Test Suite	sult Test Case Step <u>Test Environment (4)</u> T	est Run Test Segment
🕒 Evaluate	Name	Type Values	Default Action
🖋 Configure	Checked by Text Additional informations or remarks Text Text Laboratory experiment Boolean	- Id -	1 ⑪ 1 ⑪ 1 ⑪
	Edit the elements of the enumeration		1
	Name	Action	
	Temperature < 0°C	1 🛍	
	Cre 0°C < Temperature < 20°C	1 🛈	d 3 months ago by Felix Mustermann
	Room temperature	Į U	Discard Back
	40°C < Temperature	↓ Ш ‡ Ш	
	New	OK Cancel	

Figure 5.22. Editing an Enumeration Property

Once custom properties are created, they can be edited in the *Custom* view of the associated object (Figure 5.23).

≡ 📌 K L A R	o s test managem				Finance Tracker 🛅 🚍		Q X	Ø~ ≗ ~
📝 Define	ENV00005 - Android 8 Smartpho	ne					(🖹 🗋 😋 😜
🏝 Plan	Overview Properties User Defi	ned Attachments	Iterations (5) Jo	bs Results (111)	Changes	Save	Discard	Back
🏟 Execute	Checked by							
🕒 Evaluate	Additional informations or remarks							
🖋 Configure								
	Temperature	Temperature < 0°C						~
	Created 3 years ago by Sabrina Gidley					Last c	hanged 2 years ago by	y Felix Mustermann
						Save	Discard	Back



For test runs, the values of the custom properties are captured in the respective execution dialog, see <u>Figure 8.3</u>.

5.2.3.2.5. Revisions

Requirements, Test Cases, Test Segments and Test Suites are revisionable objects.

A new revision of an object should be created when a major edit is taking place. For example, an older revision of a test case can only be executed when used with older versions of a system under test as newly added test steps are only applicable to never versions of the system under test.

Revisions of objects can be managed from the *Revisions* tab. This tab shows the revision history for the object, allows the user to change the revision comment associated with each revision and

also allows creating new revisions. The revision history table may be used to select the revision to display and edit.



Figure 5.24. The "Revisions" Tab

Relationships between object revisions

Many objects have a direct relationship to other objects. Objects like e.g. Test Cases or Requirements may exist in different revisions, so one revision of an object can be referenced by any revision of other objects. Every revision of an object may contain its own references to other objects, regardless of the relationships of the other revisions. In addition, every revision of an object can be related to one or more specific revisions of another object.

Example: Given is the test case TC0001 and the requirement R0001. Requirement R0001 is covered by the test case TC0001. Both are still in the initial revision 1.0.



Now a new Revision 1.1 of Requirement R0001 is created which is also covered by Test Case TC0001. So TC0001 is now covering the Requirement R0001 in both revisions.



Next, a new revision of test case TC0001 is created, since revision 1.0 of requirement R0001 has become obsolete. Therefore, revision 1.1 of the test case will now reference the newest revision of the requirement.



After creating the new revision of test case TC0001, we discover that requirement R0002 is also covered by this test case revision. Therefore, we add further coverage between R0002 and the test case TC0001 in Revision 1.1.



This example shows, that although the Test Case and the Requirement are the same objects, their relationship can change over their revision history. References are defined per revision, and so can be different with each revision. This does not only apply to test cases and requirements, but also to other revisionable objects like test suites or test segments.

5.2.3.2.6. Attachments

For many objects file attachments can be uploaded and linked to this object. This can be done within the *Attachments* view on the corresponding detail page of the object or during the execution of a test case.

In the table on the attachment detail page, the *Name*, the *Size*, the *File Type*, the *Version* and the name of the creator. are displayed. The row contents can be edited directly in the table.

Click on the button Upload attachment to open a dialog. The button Q Select opens a dialog to select the attachment. With OK the selected file is uploaded to the application. If you want to upload more files, repeat the steps. Click Save to save the file permanently.



Clipboard Support

On browsers of the Chrome family (Chrome, Brave Microsoft Edge), an upload via **Ctrl+V** supported. Such, e.g. screenshots can be transferred without saving them in the file system.

= 🞐 K L A R	o s test management	Finance Tracker 📑 📑	२ × छ ≁ ≛×
📝 Define	TC00020 - Update dashboard by cancelling a debit mandate.		● 🛱 🏾 😋 🗲
Plan	Overview Properties User Defined Steps (6) <u>Attachments (2)</u> Revisions Issues	Save Jobs (1) Results (18) Changes	Discard Back
🏟 Execute	Name \$	Size ♦ File Type ♦ Ve	ersion Created By Action Falix Mustermann
🕒 Evaluate	IssuesPDF.PNG	32 KB image/png 1.	1 Felix Mustermann 🕹 🗓
🖌 Configure	Upload Attachments		
	Created 4 years ago by Talal Arif	Last ch	nanged 4 years ago by Felix Mustermann
		Save	Discard Back

Figure 5.25. The "Upload Attachments" Page

In the action column saved attachments can be downloaded (\underline{A}) and deleted ($\overline{\mathbb{II}}$).



File Size Limitation

The file size of an attachment can be limited to prevent the upload of excessively large files. This can be defined under *Configure -> System -> Other*.

If a file with the same name is uploaded multiple times, a new version of the attachment is created. This version contains the newly uploaded file.

5.2.3.2.7. Test Runs and Results



Some detail pages show a tab with test runs and test results which are related to the object being displayed, e.g. executed in the displayed test environment or executed by the displayed user. An example of a *Test Results* tab is shown in <u>Figure 5.26</u>.

🗏 👱 K L A R	OS TEST MANAGEMENT	Finance Tracker 🗧 🗮	ς x @γ ≛γ
📝 Define	SUT00005 - Finance Tracker 2.0.1		🖶 🛛 😋 Đ
🚑 Plan	Overview Properties User Defined Attachments Issues (1) Iterations	Jobs <u>Results (61)</u> Changes	Discard Back
🏟 Execute	Test Case Results (61) Test Suite Results (6) Test Runs (39)		
🕒 Evaluate	61 Entries-Page 1 of 7 K ◀ 1 2 3 4	5 🕨 🕅 🔽 🗸 🖉 🖉	Q × 丞 ≡ Duration ≑ Besult ≑ Action
€ Configure	CR0001520 4 years ago TRU0001115 C0000 TCR0001528 4 years ago TRU0001115 TC0000 TCR0000325 4 years ago TRU0000207 TC0000 TCR0000326 4 years ago TRU0000207 TC0000 TCR0000329 4 years ago TRU0000207 TC0000 TCR0000329 4 years ago TRU0000200 TC0000 TCR0000324 4 years ago TRU0000200 TC0000 TCR0000324 4 years ago TRU0000200 TC0000 TCR0000324 4 years ago 1.0.*Sprint 01-Basic functions TRU0000193 TCR0000324 4 years ago 1.0.*Sprint 01-Basic functions TRU0000193 TC0000 TCR0000321 4 years ago 1.0.*Sprint 01-Basic functions TRU0000193 TC0000	1 Android 8 Smartphone Felix Mustermann 1 Android 8 Smartphone Felix Mustermann 7 Android 10 Smartphone Tim Thiel 7 Android 10 Smartphone Markus Meyer 0 Android 10 Smartphone Tim Thiel 3 Android 10 Smartphone Tim Thiel 2 Android 10 Smartphone Thomas Tafel 2 Android 10 Smartphone Thomas Tafel 2 Android 10 Smartphone Timo Tunklik 95 Android 10 Smartphone Timo Tunklik	00:00:06 ▲ ▲ ▲ ● Q 00:00:11 ▲ ▲ ● Q 00:00:25 ▲ ● Q 0 ● ● Q 0 0:00:20 ▲ ● Q 0:00:22 ● ● Q 0:00:02 ● ▲ ● Q 0:00:02 ● ● Q 0:00:02 ● ▲ ● Q 0:00:02 ● ▲ ● ● Q 0:00:02 ● ● ● Q 0:00:02 ● ● ● ● ● ●
	Created 4 years ago by Talal Arif	Last ch Save	anged 4 years ago by Felix Mustermann Discard Back



In both of these tabs, the names of objects are links to the *Details Pages* of the objects. The Q icon displays the *Details Page* of the corresponding test run or test result.

In the *Test Runs* tab (<u>Figure 5.26</u>), reports may also be generated, as described in <u>Section 9.3.4</u>, <u>"The Test Run Report"</u>.

5.2.3.2.8. Change History



This tab displays the entire change history of the object since its creation.

Figure 5.27 shows the *Changes* tab for a test case.



Figure 5.27. The "Changes" Tab

All changes are marked in color: green for newly added entries, red for deleted entries. Changes for non-text values are marked in the form *Old value -> New value*.

Furthermore, it is indicated which user has carried out a change and at what time it was made. If several changes were made at once, they are grouped together per save operation.

5.3. Main Functions

In this section, general cross-cutting functions of Klaros-Testmanagement are described in detail.

5.3.1. Resolving save conflicts

If an object is edited by two users at the same time, these changes may collide. The following measures are provided for this in the application.

When saving an object, an attempt is made to automatically resolve possible conflicts with concurrent changes as far as possible. Any changes made by other users that do not affect fields that the current user has changed are merged without additional user input.

If field contents have been changed that were also changed by another user in the meantime, a dialog is displayed. There the user can choose how the conflict should be resolved.

Figure 5.28 shows the *Conflict Resolution Dialog* for a conflicting test case change.

≡ 👱	Finance Tracker 🖀 🚍	۹ 🚱 ۲ 🛓 ۲
📝 Define	Test Suites	Dura
📇 Plan	New 10 2 2 10 7 Entries-Page 1 of 1 M 1 M 10 2 2 5 7 Show all	Save Discard
🌣 Execute	ID+ Q Revision Name \$ System under Test \$ TS00008 1.0 Check the connection to the server Finance Tracker 2.1.0	Test Cases ♦ Action
Evaluate	TS00007 1.0 Edit stading orders Tmane House 2.1.0 TS00007 1.0 Edit stading orders	
	TS00004 1.0 Consisting Concurrent modification conflict	
🔑 Configure	Your changes are conflicting with concurrent modifications done by other users. Please select the data that should finally be stored.	* CCQU * CCQU
	ar Test	Test Cases Action
	New Overviewpage (User) Overview (Database)	Save Discard
	Save Discard	

Figure 5.28. The "Conflict Resolution" Dialog

For each conflict, the user has the choice of saving the object with the value they entered or keep the value currently present in the database. For lists, e.g. the list of test steps in a test case, the user has the option to *Merge* the changes or to keep the version which is currently in the database. This is to prevent the inadvertent loss of data through the deletion of objects which were added by other users.

The *Merge* option builds a list containing all the elements of both lists (user input and database) in an approximation of the correct order.

Clicking the Save button persists the selected changes to the database, while clicking the Discard button discards them.

5.3.2. Deleting, Purging and Restoring Objects

5.3.2.1. Deleting Objects

Objects can be deactivated or deleted by clicking on the icon $\widehat{\mathbf{m}}$ so that they are no longer included in reports and evaluations. If the object to be deleted is linked to other objects (e.g. a requirement linked to a test case) or is contained in other objects (e.g. a test case contained in an execution definition), a dialog is displayed (see Figure 5.29).

😑 📌 KLAR	o s test management	Finance Tracker 🖀 🛢	९ छ~
📝 Define	Requirements		
	New		Save Discard
📇 Plan	6 Entries - Page 1 of 1 🖌 🚽 1 🕨 🕅 10 🗸	品又	< × & =
🛱 Evecute	■ ID	Priority 🗢 Status 🗢 Te	st Cases 🗢 🛛 Action
	R00006 🔗 1.0 The dashboard updates correctly after something is changed.	Medium Draft	4 🖉 🗋 🛱
Evaluate	Are you sure?	Lieb Deaft	
	Do you really want to delete this requirement?		
Le Configure			2 12日日前
Conngure	ID\$ Revision Name\$		2 2000
	R00005 1.2 The application is compatible with Android smartphones, tablets and smartwatches	Т. Т	est Cases Action
	The objects listed above will be removed from the following iterations		Save Discard
	ID = Name =		
	TR00005 1.1.4-Sprint 02-Extending debboard		
	TRU0008 2.0.x-Sprint 02-Extending dashboard		
	TRUUUU7 2.0.4 Sprint 01 Revie functione		
	ITRUDUOT 1.0.4-Sprint 01-Datate functions		
	TRUDU04 1.1.4 Sprint 02 Extending basis functions		
	TR00002 1.0.4-Sprint 02-Excertaining basic functions		
	TRUDUD 21 1. Sprint 01-Overviews		
	TRUDUUg 2.1.4 Optimizing for tablete		
	TR00006 Transpirit de-optimizing for labers		
		OK Cancel	

Figure 5.29. The "Delete Objects" Dialog

In the dialog window all dependencies, relations and occurrences of the object to be deleted are displayed. With OK the object is deleted, with Cancel the window is closed, and the object is not deleted.

5.3.2.2. Purging Objects

Objects can be irretrievably removed by clicking on the icon &. For this, it is necessary that these objects have already been deleted or deactivated. By default deleted or deactivated objects are not displayed. By clicking on Show all you can show them. After clicking on the icon &, a dialog appears (see Figure 5.30). Clicking on OK restores the object, clicking on Cancel closes the window and the object remains deactivated.

= 📌 K L A R	OS TEST MANAGEMENT Finance Tracker		વ× છ~ ≛~
📝 Define	Requirements		
📇 Plan	New		Save Discard
	+ 🖸 🗋 🖗 🛱 🕺 7 Entries - Page 1 of 1 🔣 ┥ 🚺 🕨 🕅 10 🗸	유 ⑦ Only active	Q × & ≡
🏚 Execute	ID ¢ 🛛 Revision Name ¢	Priority 🗢 Status 🗢	Test Cases 🗢 🛛 Action
	R00155 C 1.0 No connection error	Draft	0 🛛 🖨 🖉 🖻
A Further	R00006 🔗 1.0 The dashboard updates correctly after something is changed.	Medium Draft	4 200
C Evaluate	R00005 🔗 1.2 The application is compatible with Android smartphones, tablets and smartwatches	High Draft	7 ピロ母曲
	R00004 🔗 1.0 It is required that the database is always in a consistent state.	High Draft	5 COQ
差 Configure	R00003 🔗 1.3 At least 100 users should be able to use the application simultaneously without any performance losses.	High Draft	· CD 🛱 🗓
	R00002 🔗 1.0 No connection error.	Low Draft	2 2000
	R00001 🔗 1.0 User input results takes less than 4 seconds to process.	Medium Draft	2 ┏┖╘面
	ID Revision Name	Priority Status	Test Cases Action
	Are you sure?		Save Discard
	Are you sure that you want to irrevocably purge the requirement from the database? This process can not be reverted.		Distard
	ID♦ Description ♦		
	R00155 1.0 - R00155, No connection error		
	0	Cancel	

Figure 5.30. The "Purge Objects" Dialog

5.3.2.3. Restoring Objects

Deleted or deactivated objects can be restored by clicking on the icon 🗊 . After clicking on the icon, a dialog appears (see Figure 5.31). Clicking on OK restores the object, clicking on Cancel closes the window and the object remains deactivated. Restored objects are included in reports and evaluations.

= 👱 K L A R	OS TEST MANAGEMENT Finance Tracker			۹	× @~ & ~
📝 Define	Requirements				
® Dian	New			Save	Discard
	+ 🕐 🗋 🖗 🖉 🛱 7 Entries - Page 1 of 1 🔣 ┥ 🚺 🕨 🕅 10 🗸	₽ Torr To	ly active		Q × & ≡
🔹 Execute	■ ID PRevision Name P	Priority 🗢	Status 🖨	Test Cases 🕯	Action
	R00778 C 1.0 No connection error		Draft	0	
	R00006 Ø 1.0 The dashboard updates correctly after something is changed.	Medium	Draft	4	C C C C C
	R00005 Ø 1.2 The application is compatible with Android smartphones, tablets and smartwatches	High	Draft	7	C C C C C
	R00004 Ø 1.0 It is required that the database is always in a consistent state.	High	Draft	5	C L O U
🔎 Configure	R00003 Ø 1.3 At least 100 users should be able to use the application simultaneously without any performance losses.	High	Draft	6	
	R00002 Ø 1.0 No connection error.	Low	Draft	2	
	R00001 Ø 1.0 User input results takes less than 4 seconds to process.	Medium	Draft	2	
	ID Revision Name	Priority	Status	Test Cases	Action
	De unu unat le restere this requirement?			Save	Discard
	O bo you want to restore this requirement?				
	ID ¢ Description \$				
	R00778 Revision: 1.0 - ID: R00778, Name: No connection error				
	OH	Car	ncel		

Figure 5.31. The "Restore Objects" Dialog

5.3.3. Referencing Object Properties

Properties and user defined properties (<u>Section 5.2.3.2.4, "User Defined Properties</u>") can be referenced and resolved in the displayed text fields while executing jobs, test cases or test suites. For example, the following test case precondition references the system under test in which it is executed:

"Please ensure that the operating system version in /config/os matches %sut:buildnumber% before executing this test case."

When executing this test case, *%sut:buildnumber%* gets replaced by the value of the user defined property *buildnumber* of the system under test for which the test case is executed. So assuming the value for the property is 1.44_alpha3, the text will be output as follows:

"Please ensure that the operating system version in /config/os matches 1.44_alpha3 before executing this test case."



Figure 5.32. Referenced Object Property

Property substitution is supported both for the common pre-defined attributes of an object (e.g. the name or id) and the user defined properties. The name of the common attributes can be derived from the corresponding bean methods defined in the <u>de.verit.klaros.core.model</u>.

Properties of the following objects are currently supported:

Iteration	itr
Job	job
System under Test	sut
Testcase	tc
Test Environment	env
Test Run	tr
Test Segment	seg
Test Suite	ts

Assume that the following example text is used in the test case precondition:

"Please make sure that your system under test is using version %sut:productversion%."

The attribute name productversion is found in the class documentation for the SUT-Implementation class in <u>de.verit.klaros.core.model.KlarosSUTImplementation</u>. Given that the ver-

sion defined there has a value of 1.0.1, the following output will be rendered when displaying the precondition:

"Please make sure that your system under test is using version 1.0.1."



Property References are Case Insensitive!

The left-hand side of a reference is case-insensitive. %sut:productversion%, %SU-T:productversion% and %Sut:productversion% will yield the same result.

5.3.4. Referencing Attachments

Similar to referencing user defined properties in test case and test case steps (<u>Section 5.3.3</u>, <u>"Referencing Object Properties"</u>), values from binary attachments can be referenced as well. Currently, CSV and XLS files are supported.

Binary attachments of the following objects are currently supported:

Iteration	itr-att
Job	job-att
System under Test	sut-att
Test Case	tc-att
Test Environment	env-att
Test Run	tr-att
Test Segment	seg-att
Test Suite	ts-att

In addition to the above, the global wildcard att can be used. The position of the attachment is thus not specified. In this case, all objects involved in the test execution are searched for the attachment.

A placeholder can be defined in the form %[object type]:[file name]:[column]:[row]% using the following parameters:

Placeholder Parameters

[object type]	The type of the object that is holding the binary attachment.
[file name]	The name of the binary attachment.
[column]	The column number of the value of the binary attachment.
[row]	The row number of the value of the binary attachment.



Referencing Excel Files

When referencing Excel files, cells can be addressed using Excel-Coordinates (e.g. C1 or AB24). This shortens the placeholder to %[object type]:[file name]:[coordinate]%.

The file name, column and row parameters can also be defined using user defined properties. So *%itr-att:customers.csv:%job:Customer%:%job:CustomerNr.%%* references the attachment customers.csv of the current iteration, using the value in the column defined in the attribute *Customer* of the current job and the row defined in the attribute *CustomerNr.*



Attachment References are Case Insensitive!

The left-hand side of an attachment reference is case-insensitive. itr-att:cus-tomers.csv:A1%, ITR-att:customers.csv:A1% and ITR-Att:customers.csv:A1% will yield the same result.

Chapter 6. Define

In the *Define* section, <u>Projects</u>, <u>Iterations</u>, <u>Requirements</u>, <u>Test Environments</u>, <u>Systems under Test</u>, <u>Test Segments</u>, <u>Test Cases</u> and <u>Test Suites</u> can be created and edited. There is a menu entry on the left-hand side for each of these object types.

6.1. Projects

A *project* collects all objects needed and created in a test project, such as test cases, test runs and test results. This chapter shows how to create and edit *projects*.

6.1.1. Overview Page

The overview page displays all existing projects in the selected project in a table. New projects are created here.

≡ 📌 K L A R	os test					Finance Tra	iker 🖬 📑		Q x	Ø~ ≗ ~
📝 Define	Projects									
• • • Dian	New								Save	Discard
			9 Entries - Page 1 of 1	K 🖪 🚺 I	▶ N 10 ∨				√ Only	active 🖧 🗏
💏 Evecute	ID 🗘 🖓	Description 🖨	Iterations 🗢	Requirements 🗢	Test Cases 🖨	Test Suites 🖨	Test Runs 🗢	Created 🗢	Last Access 🖨	Action
	O P00011		0	0	1	0	0	2 years ago	5 months ago	○ 🗗 🖨 🛍
	O P00010	Test PRV	0	0	5	1	3	2 years ago	9 months ago	○ 🗗 🖨 🛍
	O P00009	PRV 2	0	0	1	0	0	2 years ago	2 years ago	○ 🖒 🖨 🛍
	O P00008	PRV	0	0	3	1	1	2 years ago	2 years ago	○ ピ 🖨 🛍
🏓 Configure	O P00007	Issue Management Integration	0	0	4	0	0	3 years ago	7 months ago	○ ໕ 🖨 🛍
	O P00005	DE Finanz-Tracker	9	6	24	7	340	4 years ago	about 7 hours ago	o ピ 🖨 🛍
	O P00003 &	Printer Tester	0	4	1	0	0	4 years ago	about 7 hours ago	○ 🖉 🖨 🛍
	P00002 P0002 P00 P0002 P0002 P0002 P0002 P00 P	Finance Tracker	9	6	24	7	338	4 years ago	less than a minute ago	⊙ ໕暮 🛍
	O P00001	FinanceTrackerLocalTester	9	6	25	7	204	4 years ago	about 7 hours ago	○ ໕ 🖨 🏛
	ID	Description	Iterations	Requirements	Test Cases	Test Suites	Test Runs	Created	Last Access	Action
	New								Save	Discard

Figure 6.1. The "Projects" Page

The table shows the following values:

ID	Assigned automatically.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Description	The description of the project.
Iterations	The number of iterations in this project.
Requirements	The number of requirements in this project.
Test Cases	The number of test cases in this project.
Test Suites	The number of test suites in this project.
Test Runs	The number of test runs in this project.
Created	The date on which the project was created.

Actions

The executable actions.

The Description entry can be edited directly in the table rows with one click.

6.1.1.1. Create a new Project

By clicking on the button New a a new empty table row will appear. Now, the *Description* can be defined.

With <u>Save</u> the new project is created and saved. The ID of the project (P00001) is automatically assigned by Klaros-Testmanagement. Click on the ID *P00001* to get to the detail page of the project.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



Red IDs

All rows with red IDs have been changed and are not yet saved!

6.1.1.2. Actions

The following actions can be performed in the action column:

- Activate
- 🕜 Edit
- Open print view
- ரி Delete

If a project has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted projects, the following actions are available instead of *Activate* and *Delete*:

- Restore (only Administrator)
- ☆ Irrecoverably remove the project from the database

6.1.1.3. Table Operations

The following operations can be performed in the line above the table on the right:

√ Filter / Sort

Show all / Only active (only Administrator)

- 凸 Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

6.1.1.4. Quick Selection

To quickly switch between projects, a quick selection is also available in the page header (see Figure 6.2, "Quick Selection of Projects").

≡ 🖞	ITR00001 - 1.0.x-Sprint 01-Basic functions V Finance Tracker	▶ 🖬 📰	۹ 🚱 ۲ 💄 ۲

Figure 6.2. Quick Selection of Projects

6.1.2. Details Page

Each project has its own detail page with several additional tabs. Clicking on the ID of the respective project or on the icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Overview* tab.

The following tabs are available: *Properties*, *User Defined*, *Copy Objects*, *Access*, *Integration*, *Results* and *Changes*.

≡ 📌 K L A R	o s test management	Finance Tracker 🖬 🗮 🔍 🔍 🕹 V
📝 Define	P00002 - Finance Tracker	음 C
📇 Plan	Properties User Defined (6) Copy Objects Access Integration Results Changes	Save Discard Back
🕸 Execute	ID P00002 Description Finance Tracker	
🕒 Evaluate	Textfield Format HTML V	Last changed 10 months ago by Felix Mustermann
		Save Discard Back



6.1.2.1. Actions

Auf den Detailseiten lassen sich jeweils folgende Aktionen vornehmen:

Open print view
A print-ready view of the project can be created here. With a click on the icon
this opens in a new browser tab.

Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u>.

Browse Use the green arrows at the very top right to switch between the projects present on the previous page.

6.1.2.2. Properties

This tab (Figure 6.3) allows the user to view or change the following attributes of the project:

Description	The description of the project.
Textfield Format	Multiline text fields can contain HTML content or plain text.
	This is set here for the entire project.

6.1.2.3. User-Defined



A project can be adapted to specific requirements by defining additional fields for objects such as test cases. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

= 📌 K L A R	OS TEST MANAGEMENT	Finance Tracker 🔳 📑	९ ७ ~ . .
📝 Define	P00002 - Finance Tracker		a 🔁 🔁 🔊
n Plan	Properties User Defined (6) Copy Objects Access Integration Results Changes	Save	Discard Back
🔅 Execute	Iteration Job Requirement System under Test (1) Test Case (1) Test Case Step	Test Environment (4) Test Run T	est Segment Test Suite
욙 Evaluate	Name Additional information or remarks	Type Value Text Field -	es Default Action
🗲 Configure	Checked by Temperature Laboratory experiment New	Text - Enumeration (5)	L III 24°C < Temperature < 40°C ↓ III ↓ III
	Created 10 months ago by Felix Mustermann	La	st changed 2 minutes ago by Felix Mustermann
		Save	Discard Back



6.1.2.4. Copy Objects



Objects can be copied or synchronized between projects. The *Copy objects* tab displays a dropdown list of all source projects that the user can access to and that contain at least one copyable object. The *Transfer user defined fields* checkbox determines whether user defined properties should be copied as well. Define



Figure 6.5. The "Copy Objects" tab

Once a source project has been selected, the lists of objects that can be copied (Requirements, Test Environments, test systems, test cases, test segments and test suites) appear, as well as a radio button to select the mode (Copy or Synchronize).

Two modes are available.

- Copy
- Synchronize

Copied objects have no reference to their source object, so subsequent changes to the source object are not propagated to the copies.

Synchronized objects have a reference to their source object and can *not* be edited. Changes to the source objects can be transferred to the copies at any time.



What are synchronized objects needed for?

Synchronized objects allow you to manage frequently used test cases in a reference project and reuse them in other projects.

Already synchronized objects are hidden and can be displayed by clicking the Show all button. If there are changes to the source object for a synchronized object, the **①** icon is displayed in the *Additional notes* column. Clicking on it opens a dialog with which it can be synchronized.



Synchronization Mode

Currently only test cases and test suites are supported in synchronization mode.

After one or more objects have been selected, clicking the Copy (Synchronize) button opens a dialog in which all objects are shown in a list. Clicking the Ok starts the copy process (see Figure 6.6, "The "Copy Objects" Dialog ").

= 👱 K L A R	o s test management	Finance Tracker 🖬 🗮	Q x Ø≁ ≛*
📝 Define	P00003 - Printer Tester		🖨 😋 😜
🏩 Plan	Properties User Defined Copy Objects Access (3) Integration Results Changes	Save	Discard Back
🔅 Execute	Transfer user defined fields Source Project P00002 - Finance Tracker ([6 R], [5 ENV], [6 SUT], [24 TC], [1 SEG], [7 TS])		
L'addate	Are you sure?		
🖋 Configure	The following objects will be transferred to the selected project		ready existing objects.
	Test Environments		
	ID Description		
	ENV00005 Android 8 Smartphone		Q × & =
	ENV00004 Android 9 Smartphone		
	ENV00003 Android Smartwatch		
	ENV00002 Android Tablet		
	ENV00001 Android 10 Smartphone		
	Do you want to copy these objects to project P00003 - Printer Tester?		
	Keep test case states (e.g. Draft/Approved)?		
		OK Cancel	Сору
	Created 3 years ago by Felix Mustermann	<u>ــــــــــــــــــــــــــــــــــــ</u>	ast changed 2 years ago by selen2 selen2
		Save	Discard Back

Figure 6.6. The "Copy Objects" Dialog

The *Keep test case status* checkbox determines whether the defined test case statuses should be copied as well or whether the *Draft* status should be set for all test cases instead.



Note

When a test suite is copied, all contained test cases are automatically copied as well.

6.1.2.5. Access

A project can be secured against outside access by enabling the *Disable anonymous project access* option.

When this option is enabled, a user must always be logged in to Klaros-Testmanagement to view pages referenced via bookmarks. For more information on bookmarking, see <u>Section 5.2.3.2.2</u>, <u>"Bookmarks"</u>.

If this option is disabled, a page referenced via bookmarks is accessible to everyone even without logging in. As soon as the user then tries to switch to another page, he is redirected to the login screen. If a user is not yet logged in, this is indicated by a corresponding icon in the <u>Section 5.2.2.4</u>, <u>"The User Menu"</u> at the top right.

6.1.2.5.1. Assigning Project Specific Roles



Only available in Klaros-Testmanagement Enterprise Edition

Define

= 📌 K L A R (S TEST MANAGEMENT	Finance Tracker 🏾 📑 📑	Q x 0⁄ ≛×
📝 Define	P00003 - Printer Tester		🖨 😋 😜
🏝 Plan	Properties User Defined Copy Objects Access (3) Integration	Results Changes	Save Discard Back
🏟 Execute	Anonymous project access disabled		
🕒 Evaluate	A 3 Entries - Pa	je1of1 🕅 ┥ 🚺 🕨 🕅 10 🗸	7 & ≡
	Username 🗢 🛛 🗧 🖉 Full Name 🗢	Global Role 🗢	Project Role 🗢 🛛 Action
🔑 Configure	manager Max Mustermann	Test Manager	Test Manager 🗸 –
, , , , , , , , , , , , , , , , , , ,	olli Oliver Krams	Tester	Tester V -
	Tegen Till Tegen	Tester	Guest 🗸 –
			Action
	Created 4 years ago by Felix Mustermann		Last changed 3 years ago by selen2 selen2
	Assign		Save Discard Back

Figure 6.7. The "Access" Tab

Klaros-Testmanagement Enterprise Edition features a global role based access system (see <u>Section 10.3, "Users"</u>). In the Access tab, administrators and test managers can change the global role of a user to an individual project role. For example, a user with the global role tester can be assigned to one project as a tester and to another project as a test manager.

Initially, every project is accessible by every user. If at least one test manager has been assigned to a project, access to this project is restricted for all other users not explicitly assigned.

Pressing the Assign button opens up a dialog (see <u>Figure 6.8</u>, "The "Assign Project Role" Dialog ") where new users can be added to the project.

Define P00003 - Printer Tester		BAA
Properties User Defined Copy Objects Access (3) Integration Results	Save	Discard Back
Execute Anonymous project access disabled		
Evaluate Assign Project Role Assign Project Role The following users are assigned the role selected below in this project.		t Role \$ Action ✓ -
8 of 8 selected 8 Entries - Page 1 of 1 K 4	▶ ₩ 10 ▼ 🖧 ☰	× -
Usemame 🕈 🛛 🛛 🖓 Full Name 🗘	Global Role 🗢	Action
✓ gross Günther Gross	Guest	
C ✓ Meyer Markus Meyer	Test Manager	anged 3 years ago by selen2 selen2
✓ Jania Sania Bergen	Tester	Discard Back
✓ tester Erika Mustermann	Tester	Distant
✓ Thiel Tim Thiel	Tester	
✓ Toppler Tanja Toppler	Tester	
Tunklik Timo Tunklik	Tester	
Add users as Test Manager 🗸	OK Cancel	





Every project needs at least one test manager!

A project needs at least one assigned test manager. When trying to save a project which has no test manager assigned, a warning message will be displayed in the log panel (see <u>Section 5.2.2.2, "The Log Panel"</u>).

The following collection actions can be performed here:

- & Change user role in project
- 🗊 Delete
- 6.1.2.6. Integration

In this view, external systems such as issue management and requirements management can be linked with Klaros-Testmanagement.

6.1.2.6.1. Issue Management

The issue management systems that are in use in the project can be edited here.



Creating an Issue-Management-System

Before an issue management system is available on this page, it must first be created by an administrator.

For instructions on the creation of issue management systems, see <u>Section 10.5.1</u>, <u>"Issue Management"</u>.

The tab shows two tables: The left table contains the issue management systems that are available, while the right table shows the issue management systems that are already assigned to the project. Pressing the > icon adds the corresponding issue management system to the project. To add all issue management systems press the > icon. The < and \ll are used in the same way to unassign issue management systems. It is also possible to drag and drop an issue management system to a list.

The icons in the center can be used to perform the following actions:

- > Assign the selected issue management system.
- » Assign all issue management systems.
- « Unassign all issue management systems.
- < Unassign the selected issue management system.

Define



Figure 6.9. The "Integration/Issue Management" Tab

6.1.2.6.2. Requirements Management



See <u>Section 10.5.2, "Requirements Management"</u> for more information about how to connect remote requirement management systems to Klaros-Testmanagement. ≡ 🎐 Finance Tracker 🛅 📑 Q × 00∨ **.**... P00003 - Printer Tester **₿ € €** 📝 Define Save Discard Back 📇 Plan Properties User Defined Copy Objects Access (3) Integration Results Changes 🏚 Execute Issue Management Requirements Manag Evaluate Sync Requirements from RM00001 (SYNC) V Jira (SYNC) 🔑 Configure Synchronized Types Task X Sub-task X ~ Field Mappings 11 Entries - Page 1 of 2 🛛 🖌 🚺 🚽 🚺 2 🕨 🕅 10 🗸 7 & ≡ ~ Attachmen Enum ~ BUGTESt String Text ~ Description ~ Fix Version/s Enum Issue Type Text ~ KASSEL_TEST Text Parent Text Enum ~ R Priority Priority Text ~ Project × Status String Created 4 years ago by Felix Mustermann Last changed 3 years ago by selen2 selen2 Save Discard Back

Connecting a Requirements Management System

Figure 6.10. The "Integration/Requirements Management" Tab

A project can be connected to a requirement management system via the *Requirements Management* tab on the project detail page. This tab will show a dropdown list of all available requirement management systems.



Important

In order to connect an RMS to a project, the RMS has to be configured in the *Configure* section first (see <u>Section 10.5.2, "Requirements Management"</u>).

After pressing the Save button, a background synchronization will start that will load all requirements from the configured RMS and stores them locally.



Note

Synchronizing requirements can take up a huge amount of time depending on the number of requirements on the remote side. A built-in scheduler will synchronize all requirements of all connected requirement management systems for all enabled projects in specified intervals.



Warning

When enabling a remote RMS any local requirements that have already been created will be disabled. Switching back to local requirement management will enable those requirements again.

Linking Types

After selecting a requirement management system from the dropdown list, a list of issue types appears. Per default all types are selected, so a background synchronization will synchronize requirements of any type.

Issue Management	Requirements Management	
Sync Requirements fro Synchronized Type	om RM00001 (SYNC) V Des Task X Sub-task X V	Jira (SYNC)

Figure 6.11. The Synchronized Types of the Requirements Management System

Linking Fields

Administrators and project managers can link individual fields between the remote requirement management system and Klaros-Testmanagement. When using Jira, as per default, the *Description, Priority* and *Summary* fields from the remote RMS (if they are available) are linked to the *Description, Priority* and *Name* fields in Klaros-Testmanagement. In addition to that, the Jira *Status* field is automatically mapped to a dedicated *Status* field that is only displayed if RMS synchronization is enabled.

Fields can be one out of three types: *String*, *Enum* and *Boolean*.



Note

An enum field in Klaros-Testmanagement can only be linked to an enum field of the remote RMS, but an enum field of the RMS can be linked to either a string or boolean field in Klaros-Testmanagement.

Linking Enum Values

When linking enum fields, it is necessary to indicate which enum values should be linked together. Otherwise, all values of the RMS will be mapped to the default enum value of the Klaros-Test-management field.

Clicking the 😰 icon of an enum linking opens up a dialog where individual values can be linked.

Define

≡ 📌 KLAR	0 S TEST MANAG	EMENT			Finance Tracker			ଦ ଡ~ ≗~
📝 Define	P00003 - Printer Tester							a 🔁 🔁 Đ
						Save	Discard	Back
📇 Plan	Properties User Defined (Copy Objects Access (3) Integration Results	Changes				
🔅 Execute	Issue Management Requi	rements Management						
♥ Evaluate ✓ Configure	Sync Requirements from RMC Synchronized Types Ta	00001 (SYNC) V sk X Sub-task X V						🔷 Jira (SYNC)
	Link enum values - Priority/Pric	rity						7 & =
	Remote Enum	Value 🗢		Local Enum Value 🗢				ta Suna 📤 Action
	Blocker	Hig	h			~		ito sync 🗣 Action
	Critical	Hig	h			~		
	Major	Me	dium			~	•	
	Minor	Low	i			~	•	
	Trivial	Low	1			~	×	
					Save	Cancel	~	
	Parent	Text					~	
	Priority	Enum	Priority				~	Ľ
	Project	Text					~	
	Status	String					~	
	Created 10 months ago by Felix Muste	mann				Last chang Save	ed about 10 hours Discard	s ago by selen2 selen2 Back

Figure 6.12. Linking Enumeration Values



Note

Multiple enum values of a remote enum field can be linked to the same enum value of a field in Klaros-Testmanagement.

Synchronizing Enum Values

When linking enum fields, it is often tedious to keep the list of selectable values in sync. For this, an automated synchronization mechanism for enumeration values is available using the *Auto Sync* column.

Using AutoSync requires that the remote field is an enum field. In addition, the local field must be set to a user defined requirement enum property. Once this is satisfied, the *Auto Sync* field will be displayed with a check mark in the list.



Note

It is not necessary to specify all the enum field values. Once the *Auto Sync* flag is set, the remote system is queried for available values and values for the local field are automatically changed accordingly.

Switching the Assigned Requirements Management System

The requirement management system (RMS) of a project can be changed any time. All currently existing requirements of the project will be disabled, as long as another RMS is used.

6.1.2.7. Results

The results tab is further divided into a *Test Runs, Test Case Results* and a *Test Suite Results* tab, showing the test results related to this project as described in <u>Section 5.2.3.2.7, "Test Runs and Results"</u>.

6.1.2.8. Changes

The tab Changes shows the change history of this project.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

6.2. Iterations



A *Iteration* represents a test cycle in a project and allows to synchronize the test process with an agile development process (e.g. Scrum). This chapter shows how to create and edit *iterations*.

6.2.1. Overview Page

The overview page displays all existing iterations in the selected project in a table. New iterations are created here.

≡	👱 KLAR	OS TEST M.	ANAGEMENT	Finance Tracker 🧧 🗮	२ × 0 × ≛ ×
Ľ	Define	Iterations			
	Di	New			Save Discard
· 🕋	Pidii	C C A Ó	9 Entries - Page 1 of 1 🛛 🖌 📕 🚺 🔽	品 ⑦ Show all	Q × & Ξ
à	Execute	□ ID \$ Q	Name 🗢	Start ♦ Due ♦ [†]	Fest Runs 🗢 🛛 Action
		O ITR00009	2.1.x-Sprint 01-Optimizing for use on smartwatches	Jun 30, 2020 Jul 30, 2020	46 0 🖉 🗋 🛱
¢	Evaluate	O ITR00008	2.0.x-Sprint 02-Extending dashboard	Jun 16, 2020 Jun 30, 2020	5 000000
		O ITR00007	2.0.x-Sprint 01-Integrating dashboard	May 29, 2020 Jun 18, 2020	
ىر	Configure		1.1.x-Sprint 04-Optimizing for tablets	May 6, 2020 May 21, 2020	
			1.1.x-sprint 03-Cancelling debit mandates	Apr 30, 2020 May 7, 2020	
			1.1.x-sprint 02-edit standing orders	Mar 10, 2020 Apr 29, 2020	
			1.0 x-Sprint 02-Extending basic functions	Eeb 7 2020 Feb 29 2020	
			1.0 x-Sprint 02-Basic functions	Jan 27, 2020 Feb 12, 2020	37 0 C2
		ID	Name	Start Due	Test Runs Action
		New			Save Discard

Figure 6.13. The "Iterations" Page

The table shows the following values:

ID	Assigned automatically.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Name	The editable name of the iteration.

Start Date	The date on which this iteration starts.
Due Date	The date on which this iteration is due to be finished.
Test Runs	The number of test runs.
Actions	The executable actions.

Name, Start Date and Due Date can be edited directly in the table rows.

6.2.1.1. Creating a new Iteration

By clicking on the button New a a new empty table row will appear. Now, *Name, Start Date* and *Due Date* can be defined.

With <u>Save</u> the new iteration is created and saved. The ID of the iteration (ITR00001) is automatically assigned by Klaros-Testmanagement. Click on the ID *ITR00001* to get to the detail page of the iteration.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.



Red IDs

All rows with red IDs have been changed and are not yet saved!

With Discard all changes are undone.

6.2.1.2. Actions

The following actions can be performed in the action column:

⊙; Activate

🖉 Edit

- Duplicate
- Open print view
- 前 Delete

If an iteration has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted iterations, the following actions are available instead of *Delete*:

- Restore (only Administrator)
- ${\displaystyle \bigotimes}\;$ Irrecoverably remove the iteration from the database

6.2.1.3. Bulk Actions

On the *Iterations* page, one or more iterations can be selected for bulk actions. Bulk actions are described in <u>Section 5.2.3.1.5, "Bulk Actions"</u>.

The following bulk actions are supported for iterations:

🕜 Edit

Duplicate

Open print view

前 Delete

- Restore (only Administrator)
- ☆ Irrecoverably remove the iteration from the database (only Administrator).

□ Assign to a category (appears only after a category has been created).

Bulk actions are described in detail in Section 5.2.3.1.5, "Bulk Actions"

6.2.1.4. Table Operations

The following operations can be performed in the line above the table on the right:

- 品 Categorize
- √ Filter / Sort

Show all / Only active

- Q Search
- Constant Appendix Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

6.2.1.5. Quick Selection

To quickly switch between iterations, a quick selection is also available in the page header (see Figure 6.14, "Quick Selection of Projects and Iterations").

≡ 👱	ITR00001 - 1.0.x-Sprint 01-Basic functions V Finance Tracker	✓ Ξ Ξ	۹ 🚱 ۲ 💄 ۲

Figure 6.14. Quick Selection of Projects and Iterations

When an iteration is active, only test objects relevant to the test cycle represented by the iteration are visible. If a test run is executed while an iteration is active it will be linked to the iteration automatically. This helps to separate the test results of a given test cycle from other activity.

6.2.2. Details Page

Each iteration has its own detail page with several additional tabs. Clicking on the ID of the respective iteration or on the icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Overview* tab.

The following tabs are available: Overview, Properties, User Defined, Attachments, Test Environments, Systems under Test, Requirements, Jobs, Results and Changes. Define





6.2.2.1. Actions

Auf den Detailseiten lassen sich jeweils folgende Aktionen vornehmen:

🖨 Open print view	A print-ready view of the iteration can be created here. With a click on the icon 🖨 this opens in a new browser tab.
	Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .
☐ Create Bookmarks	Each individual detail page can also be reached directly via a hyperlink. By clicking on the icon \Box this link is copied to the clipboard.
	The creation of bookmarks is described in detail in <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".
G 🖨 Browse	Use the green arrows at the very top right to switch between the iterations present on the previous page.

6.2.2.2. Overview

All data displayed here always refer to the currently displayed iteration.

The following values are displayed on the overview tab:

SuccessThe overall success rate of all test cases. A success rate of
100% means that the latest test run for every test case has
been successful. Even if a test case has been successfully ex-
ecuted in the past, only the latest result is considered for the
success rate.





Progress	The progress rate shows how many test cases have been ex- ecuted. In contrast to the success rate, the progress rate only takes into account whether a test case was executed at least once in the iteration, regardless of its result.
Compliance	The compliance rate shows how many test cases assigned to requirements have been successfully executed.
Coverage	The coverage rate shows how many test cases assigned to requirements have been executed at least once. In contrast to the compliance rate, the coverage rate considers only if a test case has been executed at least once in the iteration, regard- less of its result.
Start Date	The date on which this iteration starts.
Due Date	The date on which this iteration is due to be finished.
Execution Time (manual) and Execution Time (automated)	The overall execution times from manual and automated test executions of the iteration.
System under Test Radar Chart	This chart shows the success and progress rate of this itera- tion for every combination of test environment and system un- der test which have been configured for this iteration.

System under Test Finance Tracker 1.1.0 Project Finance Tracker, Iteration 1.1.x-Sprint 01-Overviews






Note

All test cases that are part of the iteration (via requirements) contribute to the metrics of this tab. If none are explicitly registered for this iteration, then all test cases of the project are considered.

6.2.2.3. Properties

This tab (Figure 6.18) allows the user to view or change the following attributes of the iteration:

Description	A description of the iteration.
Success Criteria	A description of the conditions which need to be met for this iteration to be complete.
Start Date	The date on which this iteration starts.
Due Date	The date on which this iteration is due to be finished.

Finance Tracker 👕 📑 द x 🚱 × 💄 × ITR00009 - 2.1.x-Sprint 01-Optimizing for use on smartwatches 📝 Define Save Discard Back 📇 Plan Overview Properties User Defined Attachments Test Environments (4) Systems under Test (1) Requirements (4) Jobs Results Changes 🏚 Execute ID ITR00009 🕒 Evaluate Name 2.1.x-Sprint 01-Optimizing for use on smartwatches escription Optimizing for use on Android smartwatches. Some functions are possible to do on a smartwatch. 👂 Configure Success Criteria Optimizing for use on smartwatches: The User can install an App for a smartwatch. The User can see the last 3 bank statements on the smartwatch · The User gets notifications when a new transaction is made Start 30.06.2020 🗰 Due 30.07.2020 🗰

Figure 6.18. The "Properties" Tab - Iterations

6.2.2.4. User-Defined

You can create your own fields to meet individual requirements. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

6.2.2.5. Attachments

You can add any files as attachments to an iteration. For more information, see <u>Section 5.2.3.2.6</u>, <u>"Attachments"</u>.

6.2.2.6. Test Environments

If you want to associate test environments with the iteration, click the Assign button and select one or more test environments.

6.2.2.7. Systems under Test

The page lists all systems under test assigned to the iteration, with their respective states:

ID	Assigned automatically.
Version	The Version of the system under test.
Success	The percentage of passed tests for this system under test that were executed in this iteration.
Progress	The percentage of tests executed in this iteration for this system under test.
Compliance *	The percentage of tests executed for this system under test that cover the requirements assigned to this iteration.
Coverage*	The percentage of tests executed for this system under test that cover the requirements assigned to this iteration and last had the result <i>Pass</i> .

* Only present if at least one requirement is associated with the iteration.

If you want to associate systems under test with the iteration, click the Assign button and select one or more test environments.

6.2.2.8. Requirements

The *Requirements* tab shows the requirements which have been assigned to this iteration, as well as their *Compliance* and *Coverage* rates. These metrics refer to the test cases linked with the requirements and the last results of these executions. The following values are displayed:

ID	Assigned automatically.
Revision	The revision of the requirement.
Name	The name of the requirement
Compliance	The percentage of executed test cases, which cover require- ments of this iteration.
Coverage	The percentage of executed test cases, which cover require- ments of this iteration and which were last executed with the result <i>Passed</i> .

If you want to associate requirements with the iteration, click the Assign button and select one or more requirements.

6.2.2.9. Jobs

The page displays a list of jobs assigned to the selected iteration.

The following values are displayed in the table:

ID	Assigned automatically.
Summary	The summary of the job
Priority	The priority of the job. Possible values are <i>Trivial, Minor, Major, Critical, Blocker</i>
State	The state of the job. Possible values are New, Reopened, In Progress, Resolved, Closed, Rejected.
Progress	The percentage of executed test cases of this job and its sub- jobs.
Success	The success rate of the executed test cases of this job and its subjobs.
Due Date	The date on which this job is due to be finished.
Assigned	The user to whom the job is assigned.
Action	Available actions are: C Edit, Open Print View und Execute. For the possible representations for Execute, see Section F.2.2, "Execution Actions".

6.2.2.10. Results

The results tab is further divided into a *Test Runs, Test Case Results* and a *Test Suite Results* tab, showing the test results related to this iteration as described in <u>Section 5.2.3.2.7, "Test Runs and Results"</u>.

6.2.2.11. Changes

The tab *Changes* shows the change history of this iteration.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

6.3. Requirements



A *requirement* is a predefined condition to be fulfilled that specifies and describes the properties or performance to be achieved by a product, a process or a system. This chapter shows how to create and edit requirements.

6.3.1. Overview Page

The overview page displays all existing requirements in the selected project in a table. New requirements are created here.

≡ 👱	KLAR	os⊺	ES	SΤ	MAN	I A G E M E N T Finance Tracker 🖬			۵ ،	< 0× ±×
📝 Defi	ine	Requi	reme	ents	5					
191 Dian	n	N	ew						Save	Discard
		+ 🖉	0	p d	ə û	6 Entries - Page 1 of 1 🛛 🚽 🚺 🕨 🕅 10 🗸	# 7	Show all		Q × & ≡
📩 Ever	cute							🕈 Status 🖨		Action
	oute	RO	0006	0	1.0	The dashboard updates correctly after something is changed.	Medium	Draft	4	┏┍₽₫
		R00	0005	Θ	1.2	The application is compatible with Android smartphones, tablets and smartwatches	High	Draft	7	嫦健₿₫
C, Eval	luate	RO	0004	O	1.0	It is required that the database is always in a consistent state.	High	Draft	5	嫦健₿₫
		RO	0003	S	1.3	At least 100 users should be able to use the application simultaneously without any performance losses.	High	Draft	6	嫦健₿₫
🖉 🔑 Con	nfigure	RO	0002	O	1.0	No connection error.	Low	Draft	2	嫦健ᇢ◍
		R00	0001	O	1.0	User input results takes less than 4 seconds to process.	Medium	Draft	2	┏┖◓◍
			D		Revision	Name	Priority	Status	Test Cases	Action
		N	ew						Save	Discard

Figure 6.19. The "Requirements" Page

The table shows the following values:

ID	Assigned automatically.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Revision	The revision of the requirement.
Name	The name of the requirement.
Priority	The priority of the requirement. Possible values are <i>Low</i> , <i>Medium</i> , <i>High</i> .
State	The state of the requirement. Possible values are <i>Locked</i> , <i>Approved</i> , <i>Draft</i> and <i>Skip</i> .
Test Cases	The number of test cases assigned to this requirement.
Action	The executable actions.

Name, Priority and Status can be edited directly in the table rows.

6.3.1.1. Creating a new Requirement

By clicking on the button New a new empty table row will appear. Now, *Name*, *Priority* and *Status* can be defined.

With <u>Save</u> the new requirement is created and saved. The ID of the requirement (R00001) is automatically assigned by Klaros-Testmanagement. Click on the ID *R00001* to get to the detail page of the requirement.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



Unsaved Changes

All rows marked with red IDs contain changes that have not yet been saved.

6.3.1.2. Action

The following actions can be performed in the action column:

🕜 Edit

- Duplicate
- Open print view
- 🗊 Delete

If a requirement has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted requirements, the following actions are available instead of *Delete*:

- Restore (only Administrator)
- ${\displaystyle \bigotimes} \,$ Irrecoverably remove the requirement from the database

6.3.1.3. Bulk Actions

On the *Requirement* page, one or more requirements can be selected for bulk actions. Bulk actions are described in <u>Section 5.2.3.1.5, "Bulk Actions"</u>.

The following bulk actions are supported for requirements:

- 🕜 Edit
- Duplicate
- Create new Revision
- Open print view
- 🗊 Delete
- Restore (only Administrator)
- ⊘ Irrecoverably remove the requirement from the database (only Administrator).

For additional information, see Section 5.2.3.1.5, "Bulk Actions".

6.3.1.4. Table Operations

The following operations can be performed in the line above the table on the right:

- 品 Categorize
- √ Filter / Sort

Show all / Only active (only Administrator)

- Q Search
- 옰 Export
- ⊟ Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

6.3.1.5. Remote Synchronization

If the active project is connected to a remote requirement management system, editing of requirements is disabled for all users. An \mathcal{C} icon will be shown on the upper right side of the requirements list screen. Clicking on it will start a background synchronization of all requirements from the requirement management system to the project.

≡ 📌 K L A R (OS TES'	t mai	NAGEME	N T Printe	er Tester 🔳 📑			૧×છ ∼	* ~
📝 Define	13:51:51	The curren	nt selected proje	ct is: Printer Tester (P00003)					+ ×
	Requiremen	ıts							C
Plan 🚬							n 🖂	0	
🏟 Execute	+ 😄	2 Revision	External ID 🖨	4 Entries - Page For T	F	Priority ≑	品 Ⅴ Test Cases ≑	External State 🖨	Action
	R00004	1.0	SYNC-4	It is required that the database is always in a consistent state.	F	ligh	0	In Progress	QB
🕒 Evaluate	R00003	1.0	SYNC-3	At least 100 users should be able to use the application simultaneously without any plosses.	performance N	/ledium	0	To Do	0₽
	R00002	1.0	SYNC-2	No connection error.	N	√ledium	0	To Do	Q 🖨
🔑 Configure	R00001	1.0	SYNC-1	User input results takes less than 4 seconds to process.	N	Лedium	0	To Do	< 🖶
	ID	Revision	n External ID	Name		Priority	Test Cases	External State	Action

Figure 6.20. The "Requirements Page" Using Remote Synchronization

6.3.2. Details Page

Each requirement has its own detail page with several additional tabs. Clicking on the ID of the respective requirement or on the \square icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Overview* tab.

The following tabs are available: Overview, Properties, User Defined, Attachments, Iterations, Test Cases, Revisions, Results and Changes.

6.3.2.1. Actions

On the detail pages there are more icons in the upper right corner. The following actions can be performed here:

Synchronize	Synchronizes the local requirement with the external source.
[∠ [™] Open external link	Shows the requirement in the external source.
읍 Open print view	A print-ready view of the requirement can be created here. With a click on the icon \bigoplus this opens in a new browser tab.
	Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .
☐ Create Bookmarks	Requirements may be linked to from outside of Klaros-Test- management using the link on the \square icon. By clicking on the \square icon this link will be copied to the clipboard.

Weitere Informationen hierzu finden Sie unter <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".

G S Browse

Use the green arrows at the very top right to switch between the requirements present on the previous page.

6.3.2.2. Overview

≡ 📌 K L A R	d s test management	Finance Tracker	∎ ≣ Q ×	Ø~ ≗ ~
📝 Define	R00005 - 1.2 - The application is compatible with Android sma	rtphones, tablets and smartwatches	Ę	🖹 🗋 😋 😜
📇 Plan	Overview Properties User Defined Attachments Iterations (1) Test Cases (7) Revisions (3) Results	Save Discard Changes	Back
🔯 Execute	Compliance 83% Coverage 85%	Priority High Status Draft		
🕒 Evaluate	Automation Rate 14%			
🖋 Configure	Revision 1.2 V System under Test Finance Tracker 1.0.0 V Test E	vironment Android 10 Smartphone 💙		
	Compliance History (requirement The appli	cation is compatible with Android smartphones 30 days) Project Finance Tracker cker 1.0.0, Test Environment Android 10 Smartp	s, tablets and smartwatches, last phone	
	100	Compliance Coverage		
	90			
	80			
	70			
	60			
	40			
	30 -			
	20			
	10			
	2020 2021	2022		2023
	Latest executed Test Cases	8. O KAN		0
	6 Entries - Page 1 of 1	▲ ▶ ▶ 10 ▼	T	Q × & ≡
	ID ♦ 🕴 Name ♦ Start	♦ Executor ♦ Test Run ♦ Test Environmer	nt ♦ System under Test ♦ Duration ♦ R	tesult 🗢 Action
	TC00022 Update dashboard by adding a standing order. 3 years	ago Felix Mustermann TRU0001674 Android 10 Smartpl	hone Finance Tracker 1.0.0 00:00:11	0 Q
	TC00002 Money transfer with insufficient funds. 3 years	ago Felix Mustermann TRU0001649 Android 10 Smartpl	hone Finance Tracker 1.0.0 00:00:08	0 Q
	TC00021 Update dashboard by cancelling a standing order. 3 years	ago Felix Mustermann TRU0001642 Android 10 Smartpl	hone Finance Tracker 1.0.0 00:00:10	0 Q
	TC00001 Transfer money to a local bank account. 3 years	ago Tim Thiel TRU0000015 Android 10 Smartpl	hone Finance Tracker 1.0.0 00:00:04	0 Q
	TC00003 Transfer money to a different bank account. 3 years	ago Tim Thiel TRU0000015 Android 10 Smartpl	hone Finance Tracker 1.0.0 00:00:04	0 Q
	TC00024 Check the connection to the server 3 years	ago Sabrina Gidley TRU0000013 Android 10 Smartpl	hone Finance Tracker 1.0.0 16 ms	× Q

Figure 6.21. The "Overview" Tab

The following values are displayed on the overview tab:

Compliance	The conformance rate shows how many test cases assigned to this requirement were performed with the result <i>Passed</i> .
Coverage	The coverage rate shows how many test cases assigned to this requirement have already been executed.
Automation Rate	The automation level indicates the percentage of test cases that can be executed automatically.
Priority	The priority of the requirement. Possible values are <i>Low, Medi-um</i> and <i>High</i> .
Status / External Status	The status of the requirement. If the requirement is synchro- nized from an external system, the externally managed status is displayed here.

Otherwise, the possible values are *Locked*, *Approved*, *Draft* and *Skip*.

Compliance History ChartThis diagram shows the compliance and coverage rate of this
requirement for the combination of revision, system under test
and test environment set here.

This graph shows how the conformance and coverage for this requirement has changed since it was defined. The timeline is automatically adjusted to reflect the available information.

In the lower area there is a table with the overview of the last executed test cases.

6.3.2.3. Properties

In this tab (<u>Figure 6.22</u>) the following attributes of the requirements are displayed and can be edited if the user has the appropriate permission:

ID	Assigned automatically.
Name	The name of the requirement.
Priority	Die Priorität der Anforderung. Possible values are Low, Medi- um and High
Status / External Status	The status of the requirement. If the requirement is synchro- nized from an external system, the externally managed status is displayed here.
	Otherwise, the possible values are <i>Locked</i> , <i>Approved</i> , <i>Draft</i> and <i>Skip</i> .
Summary	The brief summary of the requirement.
Description	The detailed description of the requirement.

≡ 🎐 Finance Tracker 🛅 📑 द × 0° ≜ × R00005 - 1.2 - The application is compatible with Android smartphones, tablets and smartwatches 📝 Define Save Discard Back 📇 Plan Overview Properties User Defined Attachments Iterations (1) Test Cases (7) Revisions (3) Results Changes 🏚 Execute ID R00005 Name The application is compatible with Android smartphones, tablets and smartwatches Priority High 🗸 🕒 Evaluate Status Draft ~ Summary 🔑 Configure It is required, that the application runs without causing an exception on smartphones, tablets and smartwatches Description The application should not freeze. The image has to be always sharp and correct. The application should run smoothly on following devices: Android smartphones -Android tablets -Android smartwatches Created 4 years ago by Talal Arif Last changed 4 years ago by Felix Mustermann Save Discard Back

Figure 6.22. The "Properties" Tab

6.3.2.4. User-Defined

You can create your own fields to meet individual requirements. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

6.3.2.5. Attachments

You can add any files as attachments to a requirement. For more information, see <u>Section 5.2.3.2.6, "Attachments"</u>.

6.3.2.6. Iterations

If you want to associate the requirement with an iteration, click the Assign button and select one or more requirements.

6.3.2.7. Test Cases

The coverage rate of requirements is tested by test cases. Test cases can be linked to requirements in the *Test Cases* tab. The test results of linked test cases count towards the coverage and compliance rates of the requirement.

🚍 学 KLAROS TEST MANAGEMENT 🛛 Finance Tracker 🖬 🖹	२ × 0 × ≛ ×
C Define R00006 - The dashboard updates correctly after something is changed.	🖨 🛛 😋 ᢒ
Save Overview Properties User Defined Attachments Iterations (1) Test Cases (4) Revisions Results Changes	Discard Back
¢ Execute 4 Entries - Page 1 of 1 M 1 ▶ N 10 7	Q × & Ξ
Evaluate Comparison Vide revision Vide revision	Manual &
Configure Configure Description Descripti Description Description Description	Manual 🛓
ID Revision Name Traceability Created 3 years ago by Sabrina Gidley Last chi Last chi	Execution Action
Assign	Discard Back

Figure 6.23. The "Test Cases" Tab

Test cases can be assigned to the requirement by clicking the Assign button. This opens up a dialog, where multiple test cases can be selected at once.

The following bulk actions are supported for requirements:

– Remove

The detail page of a test case shows a list of all requirements that are covered by it, as shown in section <u>Section 6.7.2.7, "Revisions"</u>.

6.3.2.8. Revisions

Requirements may exist in different revisions. For more information, see <u>Section 5.2.3.2.5</u>, "Revisions". Define



Figure 6.24. The "Revisions" Tab

6.3.2.9. Results

The results tab is further divided into a *Test Runs, Test Case Results* and a *Test Suite Results* tab, showing the test results related to this requirement as described in <u>Section 5.2.3.2.7, "Test Runs and Results"</u>.

6.3.2.10. Changes

The tab Changes shows the change history of this requirement.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

6.4. Test Environments

A *test environment* summarizes the external circumstances that can influence a test result. This can be the operating system, the physical environment, or the version of a client system, web browser, or application server. This chapter shows how to create and edit test environments.

6.4.1. Overview Page

The overview page displays all existing test environments in the selected project in a table. New test environments are created here.

≡ 👱 K L A R	OS TEST	MANAGEMENT				Finance Trac	ker 🖬 😅	Q X	@ ∗ ≗ ×
📝 Define	Test Environm	ients							
±®≛ Dian	New							Save	Discard
	C 🖨 û		5 Entries - Page 1 of 1	₩ ◀	1 🕨 🕅 1		쁆 🏹 Show all	0	λ × & Ξ
🏟 Execute	🔲 id 🗘 🤇	2			Descr	ption 🗢			Action
	ENV00005	Android 8 Smartphone							♂ 🖨 🛍
4	ENV00004	Android 9 Smartphone							☞ 🖨 🛍
C Evaluate	ENV00003	Android Smartwatch							☞ 🖨 🛍
	ENV00002	Android Tablet							☞ 🖨 🛍
🔑 Configure	ENV00001	Android 10 Smartphone							₫ 🖨 🛍
•	ID				Desc	ription			Action
	blass							0	Diseased
	New							Save	Discard

Figure 6.25. The "Test Environments" Page

The table shows the following values:

ID	Assigned automatically.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Description	The description of the test environment.
Actions	The executable actions.

The Description entry can be edited directly in the table rows by clicking in the corresponding field.

6.4.1.1. Creating a new Test Environment

By clicking on the button New a a new empty table row will appear. Now, a *Description* can be defined.

With Save the new test environment is created and saved. The ID of the test environment (ENV00001) is automatically assigned by Klaros-Testmanagement. Click on the ID *ENV00001* to get to the detail page of the test environment.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



Red IDs

All rows with red IDs have been changed and are not yet saved!

6.4.1.2. Actions

The action column is located on the far right of the table. The following actions can be performed here:

🖉 Edit

- Open print view
- 🗊 Delete

If a test environment has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted test environments, the following actions are available instead of *Delete*:

Restore (only Administrator)



6.4.1.3. Bulk Actions

Some actions can be applied to multiple test environments at the same time. To do this, select the test environments to which the action is to be applied in the leftmost column.

The following bulk actions are supported for test environments:

- 🕜 Edit
- Open print view
- 🗊 Delete
- Restore (only Administrator)
- ☆ Irrecoverably remove the test environment from the database (only Administrator).
- □ Assign to a category (appears only after a category has been created).

Bulk actions are described in detail in Section 5.2.3.1.5, "Bulk Actions"

6.4.1.4. Table operations

The following operations can be performed in the line above the table on the right:

品	Categorize				
7	Filter / Sort				
S	how all / Only active (only Administrator)				
Q	Search				
凸	Export				
≡	Column selection				

All operations are described in detail in Section 5.2.3.1, "Overview Page".

6.4.2. Details Page

Each test environment has its own detail page with several additional tabs. Clicking on the ID of the respective test environment or on the \square icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Overview* tab.

The following tabs are available: *Overview*, *Properties*, *User Defined*, *Attachments*, *Jobs*, *Results* and *Changes*.

Define

≡ 👱 K L A R	ROS TEST MANAGEMENT Finance Tracker 🖬 🗮	५ ४ 0 ° ≜ ४
📝 Define	ENV00001 - Android 10 Smartphone	🖨 🗋 😋 ᢒ
rean Plan	Overview Properties User Defined Attachments Iterations (9) Jobs Results (226) Changes	iscard Back
🔹 Execute	Success 13%	
🕒 Evaluate	Compliance 14% Coverage 50%	
🗲 Configure	System under Test Finance Tracker 1.1.0	
	Success History (last 30 days) Project Finance Tracker SUT Finance Tracker 1.1.0, Test Environment Android 10 Smartphone	
	100 . Coverage	
	90 -	
	80	
	70	
	80	
	40	
	30	
	20	
	10	•
	0 2020 2021 2022	2023
	O stars and T st	
	Systems under Test	mpliance Coverage Action
	SUT00002 Finance Tracker 1.1.0 13 13% 62%	14% 50% O
	SUT00003 Finance Tracker 1.1.1 13 61% 75%	55% <u>64%</u> O
	SUT00001 Finance Tracker 1.0.0 48 73% 79%	66% <u>64%</u> O

Figure 6.26. The "Overview" Tab

6.4.2.1. Actions

On the detail pages, there are additional icons in the top right corner of the header. The following actions can be performed here:

🖨 Open print view	A print-ready view of the test environment can be created here. With a click on the icon $$ this opens in a new browser tab.
	Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .
☐ Create Bookmarks	Each individual detail page can also be reached directly via a hyperlink. By clicking on the icon \square this link is copied to the clipboard.
	The creation of bookmarks is described in detail in <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".
G S Browse	Use the green arrows at the very top right to switch between the test environments present on the previous page.

6.4.2.2. Overview

= 👱 κlar	os test management	Finance Tracker 🖀 🚆	२ × 0 ≁ ≛×
📝 Define	ENV00001 - Android 10 Smartphone		🖨 🛛 😋 ᢒ
📇 Plan	Overview Properties User Defined Attachments Iteration	s (9) Jobs Results (226) Changes	Discard Back
🏟 Execute	Success 13%		
🕒 Evaluate	Progress 62% Compliance 14% Coverage 50%		
🔑 Configure	System under Test Finance Tracker 1.1.0		
	SUT Finance 1	Success History (last 30 days) Project Finance Tracker Fracker 1.1.0, Test Environment Android 10 Smartphone	
	100 Su	Ccess Progress Compliance Coverage	
	90		
	80		
	70		
	60		
	50		
	40		
	30		
	20 -		•
	10		•
	2020 2021	2022	2023
	Systems under Test		
	ID System under	er Test Test Runs Success F	rogress Compliance Coverage Action
	SUT00002 Finance Tracker 1.1.0	13 13%	62% 14% 50% O
	SUT00003 Finance Tracker 1.1.1	13 61%	75% 55% 64% 0
	Softwoor Finance tracker 1.0.0	48 73%	79% 66% 64% O



All data displayed here always refer to the currently displayed test environment.

The following values are displayed on the overview tab:

Success	The overall success rate of all test cases. A success rate of 100% means that the latest test run for every test case has been successful. Even if a test case has been successfully executed in the past, only the latest result is considered for the success rate.
Progress	The progress rate shows how many test cases have been ex- ecuted. In contrast to the success rate, the progress rate only takes into account whether a test case was executed at least once, regardless of its result.
Compliance	The compliance rate shows how many test cases assigned to requirements have been successfully executed.
Coverage	The coverage rate shows how many test cases assigned to requirements have been executed at least once. In contrast to the compliance rate, the coverage rate considers only if a test case has been executed at least once in the iteration, regard- less of its result.
Automation Rate	The automation level indicates the percentage of test cases that can be executed automatically.

Success History Diagram	This chart shows the historical evolution of the success,
	progress, conformance, and coverage rate of this test environ-
	ment for the selected system under test.

The timeline is automatically adjusted to the existing data.

In the lower area you find a table with the key figures of this test environment broken down by system under test.

6.4.2.3. Properties

This tab (<u>Figure 6.28</u>) allows the user to view or change the following attributes of the test environment:

Description

A description of the Testumgebung.



Figure 6.28. The "Properties" Tab

6.4.2.4. User-Defined

You can create your own fields to meet individual requirements. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

6.4.2.5. Attachments

You can add any files as attachments to a test environment. For more information, see <u>Section 5.2.3.2.6, "Attachments"</u>.

6.4.2.6. Iterations

This page lists all iterations to which this test environment is assigned.

Define

≡ 👱 K L A R	OS TEST MANAGEMENT Finance Tracker 🖬 🖬	≣Q× 2 ×
📝 Define	ENV00005 - Android 8 Smartphone	🖨 🗋 😋 😜
📇 Plan	Overview Properties User Defined Attachments Iterations (5) Jobs Results (114) Changes	Save Discard Back
🏟 Execute		
	■ ID ¢ 🛛	Start ≑ Due ≑
🕒 Evaluate	ITR00005 1.1.x-Sprint 03-Cancelling debit mandates	Apr 30, 2020 May 7, 2020
	ITR00001 1.0.x-Sprint 01-Basic functions	Jan 27, 2020 Feb 12, 2020
🔑 Configure	ITR00004 1.1.x-Sprint 02-Edit standing orders	Mar 31, 2020 Apr 29, 2020
v v	ITR00002 1.0.x-Sprint 02-Extending basic functions	Feb 7, 2020 Feb 29, 2020
	ITR00003 1.1.x-Sprint 01-Overviews	Mar 10, 2020 Mar 31, 2020
	ID Name	Start Due
	Created 4 years ago by Sabrina Gidley Assign	Last changed 3 years ago by Felix Mustermann Save Discard Back

Figure 6.29. The "Iterations" Tab

This test environment can be assigned to one or more iterations by clicking the Assign button. This opens up a dialog, where multiple iterations can be selected at once.

The following bulk actions are supported for iterations:

- Remove

The detail page of an iteration shows a list of all test environments that are assigned to it, as shown in section <u>Section 6.2.2.6, "Test Environments"</u>.

6.4.2.7. Changes

The tab Changes shows the change history of this test environment.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

6.5. Systems under Test

With *System under Test* the system/object is designated, which is to be tested. Other common terms for this are *test item* or *test object*. This can be identified, for example, by the version number of the tested software. This chapter shows how to create and edit systems under test.

6.5.1. Overview Page

The overview page displays all existing systems under test in the selected project in a table. New systems under test are created here.

≡	y KLAR (DS TEST	MANAGEMEN	Т			Finance Tracker 🔳	801	Q X	Ø~ ≗ ~
ľ	Define	Systems unde	er Test							
	Plan	New							Save	Discard
· • • •	Fiall	ư đ ũ		6 Entries - Page 1 of 1	₩ ◀	1 🕨 🎽 10 🗸		品 ⑦ Show all	Q	× & ≡
~	Execute	🔲 ID 🗘 🤇	2			Version 🗢			Issues 🕯	Action
	Execute	SUT00006	Finance Tracker 2.1.0						1/2	♂ 🖨 🛍
	F	SUT00005	Finance Tracker 2.0.1						1/0	♂ 🖨 🛍
C.	Evaluate	SUT00004	Finance Tracker 2.0.0						0/0	₫ 🖨 🛍
		SUT00003	Finance Tracker 1.1.1						1/0	☞ 🖨 🛍
يو ا	Configure	SUT00002	Finance Tracker 1.1.0						0/0	☞ 🖨 🛍
		SUT00001	Finance Tracker 1.0.0						0 / 0	☞ 🖨 🛍
		ID								Action
		New							Save	Discard

Figure 6.30. The "System under Test" Page

The table shows the following values:

ID	Assigned automatically.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Description	The description of the system under test.
Actions	The executable actions.

The Description entry can be edited directly in the table rows by clicking in the corresponding field.

6.5.1.1. Creating a new System under Test

By clicking on the button New a new empty table row will appear. Now, the *Version* of the system under test can be defined.

With <u>Save</u> the new system under test is created and saved. The ID of the system under test (SUT00001) is automatically assigned by Klaros-Testmanagement. Click on the ID *SUT00001* to get to the detail page of the system under test

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



Red IDs

All rows with red IDs have been changed and are not yet saved!

6.5.1.2. Actions

The following actions can be performed in the action column:

- 🕜 Edit
- Open print view

🗊 Delete

If a system under test has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted systems under test, the following actions are available instead of *Delete*:

- Restore (only Administrator)
- ${\displaystyle \bigotimes}\$ Irrecoverably remove the system under test from the database

6.5.1.3. Bulk Actions

Certain actions can also be applied to several systems under test at the same time. To do this, select the systems under test to which the action is to be applied in the leftmost column.

The following bulk actions are supported for systems under test:

- 🕜 Edit
- Open print view
- 🗊 Delete
- Restore (only Administrator)
- ⊘ Irrecoverably remove the system under test from the database (only Administrator).
- ☐ Assign to a category (appears only after a category has been created).

Bulk actions are described in detail in Section 5.2.3.1.5, "Bulk Actions"

6.5.1.4. Table Operationen

The following operations can be performed in the line above the table on the right:

- 옮 Categorize
- √ Filter / Sort
- Show all / Only active (only Administrator)
- Q Search
- 곬 Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

6.5.2. Details Page

Each system under test has its own detail page with several additional tabs. Clicking on the ID of a system under test or on the 😰 icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Overview* tab.

The following tabs are available: *Overview*, *Properties*, *User Defined*, *Attachments*, *Jobs*, *Results* and *Changes*.



Figure 6.31. The "Overview" Tab

6.5.2.1. Actions

On the detail pages, there are additional icons in the top right corner of the header. The following actions can be performed here:

읍 Open print view	A print-ready view of the system under test can be created here. With a click on the icon 🖨 this opens in a new browser tab.
	Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .
☐ Create Bookmarks	Each individual detail page can also be reached directly via a hyperlink. By clicking on the icon \Box this link is copied to the clipboard.
	The creation of bookmarks is described in detail in <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".
G D Browse	Use the green arrows at the very top right to switch between the systems under test present on the previous page.

6.5.2.2. Overview

≡ 👱 K L A R	o s test management	Finance Tracker 🖀 📑	۹ × 0 × ≗×
📝 Define	SUT00002 - Finance Tracker 1.1.0		🖨 🗋 🔂 ᢒ
res Plan	Overview Properties User Defined Attachments Issues Iterations (1) Jobs Results (50)	Sav	e Discard Back
🕸 Execute	Success 13%		
🔥 Evaluate	Compliance 13% Coverage 50%		
🖌 Configure	Test Environment Android 10 Smartphone		
	Success History (last 30 days) Project Finance Tracker SUT Finance Tracker 1.1.0, Test Environment Andro	oid 10 Smartphone	
	Progress Compliance	Coverage	
	100		
	90		
	70		
	70- 60		
	50		
	40		
	30		
	30		
	10		•
			•
	2020 2021	2022	2023
	Test Environments		
	ID Test Environment	Test Runs Success	Progress Compliance Coverage Action
	ENV00003 Android Smartwatch	0 0%	
	ENV00002 Android 10 Smartphone	13 13%	62% 14% 50% O
	ENV00005 Android 8 Smartphone	5 75%	33% 80% 35% O
	ENV00004 Android 9 Smartphone	7 90%	45% 80% 35% O
	Show all		
	Latest executed lest Cases 5 Entries - Page 1 of 1 M 4 1 M 10 V		

Figure 6.32. The "Overview" Tab

All data displayed here always refer to the currently displayed system under test.

The following values are displayed on the overview tab:

Success	The overall success rate of all test cases. A success rate of 100% means that the latest test run for every test case has been successful. Even if a test case has been successfully executed in the past, only the latest result is considered for the success rate.
Progress	The progress rate shows how many test cases have been ex- ecuted. In contrast to the success rate, the progress rate only takes into account whether a test case was executed at least once, regardless of its result.
Compliance	The compliance rate shows how many test cases assigned to requirements have been successfully executed.
Coverage	The coverage rate shows how many test cases assigned to requirements have been executed at least once. In contrast to the compliance rate, the coverage rate considers only if a test case has been executed at least once in the iteration, regard- less of its result.

Automation Rate	The automation level indicates the percentage of test cases that can be executed automatically.
Success history diagram	This chart shows the historical evolution of the success, progress, conformance, and coverage rate of this system under test for the selected test environment.

The timeline is automatically adjusted to the existing data.

In the lower area there is a table with the overview of the results for this test system broken down by test environment. In the *Action* column, the test environment for the above diagram can be selected.

6.5.2.3. Properties

This tab (<u>Figure 6.33</u>) allows the user to view or change the following attributes of the system under test:

Version

The version of the system under test.

E 🔮 KLAROS TEST MANAGEMENT Finance Track			
📝 Define	SUT00002 - Finance Tracker 1.1.0		
📽 Plan	Overview Properties User Defined Attachments Issues Iterations (1) Jobs Results (50) Changes		
🏟 Execute	ID SUT00002 Version Finance Tracker 1.1.0		
🕒 Evaluate	Created 4 years ago by Talal Arif		
🗲 Configure			

Figure 6.33. The "Properties" Tab

6.5.2.4. User-Defined

You can create your own fields to meet individual requirements. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

6.5.2.5. Attachments

You can add any files as attachments to a system under test. For more information, see <u>Section 5.2.3.2.6</u>, "Attachments".

6.5.2.6. Issues

This view lists all issues that are associated with this system under test.

Define



Figure 6.34. The "Issues" Tab

The table shows the following values:

ID	The ID of the issue in the external issue management system.
System	The external issue management system of this issue.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Summary	The summary of the issue.
Created	The date when the issue was created.
Created by	The user who created the issue.
Assigned	The user to whom the issue is assigned.
Priority	The priority of the issue. The possible values are specified by the external issue management system.
Status	The status of the issue. The possible values are specified by the external issue management system.
Actions	The executable actions.

6.5.2.6.1. Creating a new Issue

Clicking New opens the *Issue Details* page (<u>Section 9.6.6, "Issue Details (Creating a new Issue</u>)"), where new issues can be created and linked to the system under test. Clicking Link links existing issues to the system under test (<u>Section 9.6.7, "Link Issue</u>").

6.5.2.6.2. Action

The following actions can be performed in the action column:

🕜 Edit

- 🖸 Browse
- Delete

6.5.2.7. Iterations

This page lists all iterations to which this system under test is assigned.

Define

≡ 🞐 K L A R (DS TEST MANAGEMENT Finance Tracker 🖬 🛢	۹ × 0 × ⊾×
📝 Define	SUT00003 - Finance Tracker 1.1.1	🖨 🛛 😋 ᢒ
📇 Plan	Overview Properties User Defined Attachments Issues (1) Iterations (3) Jobs Results (113) Changes	Save Discard Back
🏟 Execute	- 3 Entries-Page 1 of 1 N ◀ 1 ▶ N 10 ♥	\[\] \[
🕒 Evaluate	ID+ V Name Introduct Introduct	Apr 30, 2020 May 7, 2020 Mar 31, 2020 Apr 29, 2020
🖌 Configure	ITTROOOD 1.1.1x-Sprint 04-Optimizing for tablets ID Name	May 6, 2020 May 21, 2020 Start Due
	Created 4 years ago by Talal Arif	Last changed 4 years ago by Felix Mustermann
	Assign	Save Discard Back



This system under test can be assigned to one or more iterations by clicking the Assign button. This opens up a dialog, where multiple iterations can be selected at once.

The following bulk actions are supported for iterations:

- Remove

The detail page of an iteration shows a list of all systems under test that are assigned to it, as shown in section <u>Section 6.2.2.7</u>, "Systems under Test".

6.5.2.8. Jobs

The page displays the list of jobs assigned to the selected system under test.

The following values are displayed in the table:

ID	Assigned automatically.	
Summary	The summary of the job	
Priority	The priority of the job. Possible values are <i>Trivial</i> , <i>Minor</i> , <i>Major</i> , <i>Critical</i> , <i>Blocker</i>	
State	The status of the job. Possible values are New, Reopened, In Progress, Resolved, Closed, Rejected.	
Progress	The percentage of executed test cases of this job and its sub- jobs.	
Success	The success rate of the executed test cases of this job and its subjobs.	
Due Date	The date on which this job is due to be finished.	
Assigned	The user to whom the job is assigned.	
Action	Available actions are: \Box Edit, \Box Open Print View und Execute For the possible representations for Execute, see Section F.2.2, "Execution Actions".	

6.5.2.9. Results

The results tab is further divided into a *Test Runs, Test Case Results* and a *Test Suite Results* tab, showing the test results related to this system under test as described in <u>Section 5.2.3.2.7, "Test Runs and Results"</u>.

6.5.2.10. Changes

The tab Changes shows the change history of this system under test.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

6.6. Test Segments



A *test segment* defines and encapsulates a sequence of test steps. In test cases, these test segments can be shown as test steps. The content of a test segment is edited centrally here and the changes are automatically applied in the test cases. This chapter shows how test segments are created and edited.

= 📌 K L A R	o s test management		Finance Tracker 🖀 😫	५ × 0 × ≗×
📝 Define	Test Segments			
🙁 Plan	New		Sa	Discard
	C C ? A û C	1 Entries - Page 1 of 1 🛛 🖌 🔰 🕨 🕺 10 🗸	品 ⑦ Show all	Q × & ≡
🔹 Execute	□ ID PRevision	Name 🗢	Steps 🗢 Test	Cases 🗢 🛛 Action
Litterate	SEG00005 1.0 Login		1	18 🕜 🖓 🛍
Evaluata	ID Revision	Name	Steps Tes	t Cases Action
	New		Sa	ve Discard
🔑 Configure				

Figure 6.36. The "Test Segments" Page

6.6.1. Overview Page

The overview page displays all existing test segments in the selected project in a table. New test segments are created here.

The table shows the following values:

ID	Assigned automatically.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Revision	The revision of the test segment.
Name	The name of the test segment.
Steps	The number of defined test steps.

Test Cases

The number of test cases in which this test segment occurs.

The Name entry can be edited directly in the table rows by clicking in the corresponding field.

6.6.1.1. Creating a new Test Segment

By clicking on the button New a a new empty table row will appear. Now, a Name can be defined.

With <u>Save</u> the new test segment is created and saved. The ID of the test segment (SEG00001) is automatically assigned. Clicking on the ID *SEG00001* takes you to the detail page of the test segment.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



Red IDs

All rows with red IDs have been changed and are not yet saved!

6.6.1.2. Actions

The following actions can be performed in the action column:

- 🖉 Edit
- Duplicate
- 🔒 Open print view
- 🗊 Delete

If a test segment has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted test segments, the following actions are available instead of *Delete*:

- Restore (only Administrator)
- ${\displaystyle \bigotimes} \,$ Irrecoverably remove the test segment from the database

6.6.1.3. Bulk Actions

Certain actions can also be applied to several test segments simultaneously. To do this, select the test segments to which the action is to be applied in the leftmost column.

The following bulk actions are supported for System under Test:

- 🕜 Edit
- Duplicate
- Create new Revision
- Open print view
- 🗊 Delete

Restore (only Administrator)

☆ Irrecoverably remove the test segment from the database (only Administrator).

Bulk actions are described in detail in Section 5.2.3.1.5, "Bulk Actions"

6.6.1.4. Table operations

The following operations can be performed in the line above the table on the right:

√ Filter / Sort

Show all / Only active (only Administrator)

Q Search

& Export

 \equiv Column selection

All operations are described in detail in <u>Section 5.2.3.1, "Overview Page"</u>.

6.6.2. Details Page

Each test segment has its own detail page with several additional tabs. Clicking on the ID of the relevant test segment or on the \mathcal{D} icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Properties* tab.

The following tabs are available: *Properties*, *User Defined*, *Steps*, *Attachments*, *Revisions*, *Test Cases*, and *Changes*.

6.6.2.1. Actions

On the detail pages, there are additional icons in the top right corner of the header. The following actions can be performed here:

읍 Open print view	A print-ready view of the test segment can be created here. With a click on the icon \bigoplus this opens in a new browser tab.	
	Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .	
☐ Create Bookmarks	Each individual detail page can also be reached directly via a hyperlink. By clicking on the icon \square this link is copied to the clipboard.	
	The creation of bookmarks is described in detail in <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".	
G S Browse	Use the green arrows at the very top right to switch between the test segments present on the previous page.	

6.6.2.2. Properties

In this view (Figure 6.37). the following attributes of the test segment can be displayed and can be edited:

= 坐 K L A R (os test m	ANAGEMENT	Finance Tracker 🖬 🛢	Q × 0 × ≛×
📝 Define	SEG00005 - Logi	1		🖨 🏹 😋 ᢒ
n Plan	Properties User	Defined Steps (1) Attachments Revisions Changes	Save	Discard Back
🏟 Execute	ID Name	SEG00005		
🕒 Evaluate	Description	These steps describe how to log into the app.		
🖌 Configure				
	Precondition	Make sure the app is installed befor testing!		
		The application is installed on the device. If not, follow directions: https://finanztracker/l Connection to the server exists.	install	
	Expected Result	The user can log into the app.		
	Postcondition	The start screen appears.		
	Note	Make sure, that an internetconnection is available.		
	Created 4 years ago by	/ Felix Mustermann		

Figure 6.37. The "Properties" Tab

ID	Assigned automatically.
Name	The name of the test segment.
Description	The textual description of the test segment.
Precondition	The precondition of the test segment.
Expected Result	The expected result of the test segment.
Postcondition	The postcondition of the test segment.
Note	An optional note.



Editor Settings

Under Define --> Project --> Properties you can set whether the editor displays only text or HTML. With the HTML setting, a bar with formatting options appears when you click in the text field.

Clicking the Save button saves the selected changes to the database, Discard discards the changes.

6.6.2.3. User-Defined

You can create your own fields to meet individual requirements. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

6.6.2.4. Test Steps

The test steps contain the following information:

Actions	The description of the action to take, e.g. Enter name and password, and click the Login button.
Expected Result	The (visible) behavior of the system predicted while carrying out the test step e.g. The user receives a message that the login succeeded.
Precondition	The Condition that must be fulfilled before the execution of the test step, e.g. The user is not yet logged in.
Postcondition	The environmental and state conditions that must be fulfilled after the execution of the test case, e.g. The user is authen- ticated and has access to the system



Figure 6.38. The "Steps" Tab

6.6.2.5. Attachments

You can add any files as attachments to a test segment. For more information, see <u>Section 5.2.3.2.6, "Attachments"</u>.

6.6.2.6. Revisions

Test segments may exist in different revisions. For more information, see <u>Section 5.2.3.2.5, "Revisions"</u>.

6.6.2.7. Test Cases

The page displays a list of test cases assigned to the selected system under test.

The following values are displayed in the table:

ID	Assigned automatically.		
Revision	The revision of the test segment.		
Name	The name of the test segment.		
Traceability	A reference to the corresponding requirement, use case or work package.		
State	The state of the test case. Possible values are <i>Locked</i> , <i>Approved</i> , <i>Draft</i> and <i>Skip</i> .		
Steps	The number of test steps.		
For more information see Section 6.7, "Test Cases".			

6.6.2.8. Changes

The tab Changes shows the change history of this test segment.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

6.7. Test Cases

A test case is a set of input values, execution preconditions, expected results and execution postconditions, developed for a particular objective or test condition, in order to determine whether an application or software system meets its specifications. This chapter shows how to create and edit test cases.

6.7.1. Overview Page

The overview page displays all existing test cases in the selected project in a table. New test cases are created here.

= 👱 K L A R	os test				Finance Tracker				Q >	< 0× ±×
📝 Define	Test Cases									
+®+ Dian	New							Sa	ave	Discard
	+ @ @ %	8 ŵ C	24 Entries - Page 1 of 3			6	퉒			Q × & ≡
🛧 Evocuto	□ ID¢ (2 🗢 Revision	Name 🗢	Traceability 🗢	Priority 🖨	Status 🖨	Execution 🗘 S	teps 🖨	Issues 🗢	Action
	TC00024	A 1.0	Check the connection to the server	Requirement 1.2.4 in testplan_16.txt	High	Draft	Automated	0	2/1	┏┖ᇢ╓
	TC00023	1.0	Change a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft	Manual	5	0/2	┏┍₽₫
🕒 Evaluate	TC00022	🛕 1.0	Update dashboard by adding a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft	Manual	6	5/0	┏┖₽₫
6 0 - 6	TC00021	S 1.0	Update dashboard by cancelling a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft	Manual	6	0/2	ぴ₽₽₫
Configure	TC00020	8 1.0	Update dashboard by cancelling a debit mandate.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft	Manual	6	0/0	┎╻ᇢ╓
	TC00019	1.0	Connection to the database.	de.verit.financetracker.web.Connection	High	Draft	Automated	0	0/0	CCQÛ
	TC00018	1.0	Connection to the server of the bank.	de.verit.financetracker.web.Connection	High	Draft	Automated	0	0/0	┏┖◓◍
	TC00017	A 1.0	Delete debit mandate.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft	Manual	5	0/0	┏┍₽₫
	TC00016	1.0	Delete standing orders.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft	Manual	5	0/0	┏┍₽₫
	TC00015	>\$ 1.0	Add a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft	Manual	5	0/0	┏┍₽₫
	ID	Revision	Name	Traceability	Priority	Status	Execution	Steps	Issues	Action
	New							Sa	ave	Discard

Figure 6.39. The "Test Cases" Page

The table shows the following values:

ID	Assigned automatically.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Revision	Die Revision des Testfalles.
Name	The name of the test case.
Traceability	The requirement assigned to this test case.
Priority	The priority of the test case. Possible values are <i>Low</i> , <i>Medium</i> , <i>High</i> .
State	The test case state determines whether it is editable or exe- cutable. Only test cases with the state Draft or Skip are ed- itable. Only test cases with the state Draft or Approved are ex- ecutable.

		Status: Te	st Case	
	Draft	Approved	Skip	Locked
Editable	\bigcirc	8	\bigcirc	8
Executable	\bigcirc	0	8	8

Execution	Manual oder Automated.
Steps	The number of associated test steps.
Issues	Die Anzahl der offenen und behobenen Issues.
Action	The executable actions.

Name, Traceability, Priority, Status and Execution can be edited directly in the table rows.

6.7.1.1. Creating a new Test Case

By clicking on the button New a new empty table row will appear. Now, *Name, Traceability, Priority* and *Execution* can be defined.

With <u>Save</u> the new test case is created and saved. The ID of the test case (TC00001) is automatically assigned by Klaros-Testmanagement. Click on the ID *TC00001* to get to the detail page of the test case.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



Unsaved Changes

All rows marked with red IDs contain changes that have not yet been saved.

6.7.1.2. Action

The following actions can be performed in the action column:

- 🕜 Edit
- Duplicate
- Open print view
- 🗊 Delete

If a test case has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted test cases, the following actions are available instead of *Delete*:

- Restore (only Administrator)
- ${\displaystyle \bigotimes}\;$ Irrecoverably remove the test case from the database

6.7.1.3. Bulk Actions

On the *Test Cases* page, one or more test case can be selected for bulk actions. Bulk actions are described in <u>Section 5.2.3.1.5, "Bulk Actions"</u>.

The following bulk actions are supported for test cases:

- + Assign to a new test suite
- 🕜 Edit
- Duplicate
- Create new Revision
- Open print view
- 前 Delete
- Restore (only Administrator)
- ☆ Irrecoverably remove the test case from the database (only Administrator).

For additional information, see Section 5.2.3.1.5, "Bulk Actions".

6.7.1.4. Table Operations

The following operations can be performed in the line above the table on the right:

- 윪 Categorize
- √ Filter / Sort

Show all / Only active (only Administrator)

Q Search

& Export

 \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

6.7.2. Details Page

Each test case has its own detail page with several additional tabs. Clicking on the ID of the respective test case or on the \square icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Overview* tab.

The following tabs are available: Overview, Properties, User Defined, Steps, Attachments, Revisions, Issues, Jobs, Results and Changes.



Figure 6.40. The "Overview" Tab

6.7.2.1. Actions

On the detail pages, there are additional icons in the top right corner of the header. The following actions can be performed here:

۲	Review	Creates a review job.
₿	Open print view	A print-ready view of the test case can be created here. With a click on the icon \bigoplus this opens in a new browser tab.

	Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .
☐ Create Bookmarks	Each individual detail page can also be reached directly via a hyperlink. By clicking on the icon 🔲 this link is copied to the clipboard.
	The creation of bookmarks is described in detail in <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".
G D Browse	Use the green arrows at the very top right to switch between the test environments present on the previous page.

6.7.2.2. Overview

The following values are displayed on the overview tab:

The state of the test case. Possible values are Locked, Approved, Draft and Skip.
The type of test case execution: Manual or Automated.
The priority of the test case: Low, Medium or High.
The user who last executed this test case.
This graph shows the minimum, maximum and average exe- cution times for this test case. This allows a better estimate of the time required for future executions of this test case.
This graph shows all test steps of this test case and which steps have been marked as failed or error during a test run. This graph can be used for determining possible misconfigu- rations, if errors happen more frequently at one specific step, for example.
This table shows the latest result of this test case for every system under test and test environment this test case has been executed in. Pressing the result icon will open up the ap- pertaining test case result.
This table shows the latest results for this test case. This table is limited to five entries. The other results can be viewed by pressing the \square icon.
This table shows the latest issues that are linked to this test case. Click on the 🔄 icon to display the issue directly in the Issue-Management-System

6.7.2.3. Properties

This tab (Figure 6.41) allows the user to view or change the following attributes of the test case:



Editor Settings

Under Define --> Project --> Properties you can set whether the editor displays only text or HTML. With the HTML setting, a bar with formatting options appears when you click in the text field.

ID	Assigned automatically.
Name	The name of the test case.
Description	The textual description of the test case.
Precondition	The precondition for this test case, describing the condition for executing the test case, e.g It is required that the database is initialized
Expected Result	The visible behavior of the system predicted while carrying out the test step, e.g. The user receives a message that the login succeeded.
Postcondition	The environmental and state conditions that must be fulfilled after the execution of the test case, e.g. The user is authen- ticated and has access to the system.
Estimated Duration	The estimated duration required to complete this test case
Note	An optional comment, such as notes to the tester.
Test Type	A group of test activities based on specific test objectives with the purpose of testing a component or system for specif- ic characteristics (Functional, Non Functional, Structural, Re- gression, Retest).
Variety	The expected outcome of the test: positive or negative.
Priority	The priority of the test case: Low, Medium or High.
Team	The team that is responsible for the test case.
Docbase	A reference to the document this test case is based on, which may e.g. contain the requirements this test case is related to.
Evaluation	The type of test result evaluation: Manual or Automated.

= 坐 K L A R (OS TEST MANAGEMENT Finance Tracker 🖬 🗮 🔍 🔍 🗙	0 ∼ ≗ ×
📝 Define	TC00022 - Update dashboard by adding a standing order.	⊇ 🗆 😋 €)
🏩 Plan	Save Discard Overview Properties User Defined Steps (6) Attachments Revisions Issues (5) Jobs (1) Results (24) Changes	Back
🏟 Execute	ID TC00022 Name Undate dashboard by adding a standing order	0
€ Evaluate ✓ Configure	Description Tests the automatic update of the dashboard by pressing the Add standing order	0
Comgue	button to add a new standing order. The standing order should have following information: Beneficiary of payment:Max Mustermann IBAN: DE19 1234 1234 1234 1234 12 Amount: € 100;- Invoice:38745 Time intervalmonthly	
	Precondition Make sure the app is installed befor testing! The application is installed on the device. If not, follow directions: https://finanztracker/install Connection to the server exists.	0
	Expected Result An new standing order is listed on the dashboard.	0
	Postcondition The dashboard has been updated.	0
	Estimated Duration 0 Dave 00:00:00 Clear	

Figure 6.41. The "Properties" Tab

6.7.2.4. User-Defined

You can create your own fields to meet individual requirements. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

6.7.2.5. Test Steps

In the *Steps* tab, test steps may be added to or removed from the test case. The following attributes of single test steps may also be changed:

New Creates a new test step.

Insert Test Segment Inserts an existing test segment.

Toggle View Mode Toggle the view mode between a tabbed and a non tabbed display.

- 4₽ Minimizes the display of all steps.
- [] Maximizes the display of all steps.

6.7.2.5.1. Creating a new Test Step

Clicking on the New button opens the step editor for creating a new test step.

1 Moves the entry to another position.

- ▶ Inserts a new step before.
- KX Inserts a new test segment before.
- ▶ Inserts a new test segment after that.
- 1 Inserts a new step after it.
- Duplicates the step.
- Deletes the step.
- and + Opens and closes the field view.

As soon as the input field is activated, a toolbar appears in the text field. The texts can be formatted with this as well as be equipped with links and pictures.

Actions	The description of the action to take, e.g. Enter name and password, and click the Login button.
Expected Result	The (visible) behavior of the system predicted while carrying out the test step e.g. The user receives a message that the login succeeded.
Precondition	Condition that must be fulfilled before the execution of the test step, e.g. The user is not yet logged in.
Postcondition	The environmental and state conditions that must be fulfilled after the execution of the test case e.g. The user is authen- ticated and has access to the system.

😑 🎐 KLAROS TEST MANAGEMENT 🛛 Finance Tracker 🖬 🗮 🔷 🔍 🖓 × 🚱 × 😫 🖤			
📝 Define	TC00022 - Update dashboard by adding a standing order.	60	
📇 Plan	Save Discard E Overview Properties User Defined Steps (6) Attachments Revisions Issues (5) Jobs (1) Results (24) Changes	Back	
🌣 Execute	New Insert Test Segment		
🕒 Evaluate	+ □ □ 6 Entries - Page 1 of 1 M ■ 10 ✓	∎ ╬ 0	
🗲 Configure	Action Expected Result Precondition Postcondition User Defined Attachments		
	Start the application by pressing the blue	4	
	icon button.		
	Step 2: Select the button that is found in the main menu. If a KA (X) P (] 🗊 🗕	
	Action Expected Result Precondition Postcondition User Defined Attachments	4	
	Select the		
	button that is found in the main menu.		

Figure 6.42. The "Steps" Tab

Pressing one of the Insert Attachment buttons next to the Description, Expected Result, Precondition and Postcondition text fields opens up a dialog which allows to embed an attachment into the text.
Define

		-
🗏 🏆 KLAR	OS TEST MANAGEMENT Finance Tracker ■ ■ Q× ??~	•~
📝 Define	TC00021 - Update dashboard by cancelling a standing order.	Ð
🏩 Plan	Save Discard Back Overview Properties User Defined Steps (6) Attachments (1) Revisions Issues (2) Jobs Results Changes	
🏟 Execute	New Insert Test Segment	
🔥 Evaluate		12 CO 11 — 11
🗲 Configure	Step 1: Start the application by pressing the blue icon button. Image: Constraint of the blue icon button. Action Expected Result Precondition User Defined Attachments	
	Name Size File Type Version Created By Activ No entries available Upload Attachments	n
	Step 2: Select the button that is found in the main menu. ¹	<u>) –</u>
	Select the (Standing Order) button that is found in the main menu.	4
	Step 3: Select the delete button. 2 4 K1 DX P [] [Action Expected Result Precondition Postcondition User Defined Attachments	<u>) –</u>
	Select the delete	÷

Figure 6.43. The "Insert Attachment" Dialog



Note

The attachments to be selected must have been previously uploaded in the *Attachments* view.

Pressing the Add button, while having an attachment selected, will insert a reference in the form of %inline-att:[attachment-identifier]%, e.g. %inline-att:9e3df155-359f-40e7-8c78-7ae9aac6a8ab%.

≡ 👱 K L A R	ROS TEST MANAGEMENT France Tracker 🖬 🗮 🔍	Ø~ ≛ ~
📝 Define	TC00022 - Update dashboard by adding a standing order.	≩∏ G€
n Plan	Overview Properties User Defined Steps (6) Attachments Revisions Issues (5) Jobs (1) Results (27) Changes	Back
🔅 Execute	New Insert Test Segment + ⊠ □ 6 Entries - Page 1 of 1	II # C)
民 Evaluate	Step 1: Start the application by pressing the blue icon button.) 🗋 💼 🗕
🌽 Configure	Action Start the application by pressing the blue Construction.	4
	Expected Result The start screen appears. %INLINE-ATT:ea752534-18da-4f56-9bd4-c35bd3e703e9%	4
	Precondition The application is installed on the device. If not, follow directions: https://finanztracker/install	e e
	Postcondition The application has started successfully.	4
	Attachments	£
	Step 2: Select the button that is found in the main menu.) 🗋 💼 🗕
	Action Select the Standing order button that is found in the main menu.	e e
	Expected Result A list of all standing orders are shown.	4

Figure 6.44. A Reference to an Attachment in the Expected Result Field

Depending on whether the attachment is an image, this reference will be replaced by a scalable preview image during the execution of the test case. Otherwise, the reference will be replaced by a hyperlink that will open the attachment in a new browser tab.

Tabular View?		Ŷ
+ TC00020 - U	Update dashboard by cancelling a debit mandate.	Step 1 of 6
Step Precondition Action	1 of 6 The application is installed on the device. If not, follow directions: <u>https://finanz</u> Start the application by pressing the blue icon button. The 1 fact the application by pressing the blue information by the application by	<u>tracker/install</u> ে ৫ ৩৪ চে ৬ ০ে জ্লা ৰ
	thtp://infosys.veri.de/titeos.come/seam/sesuscahest/download?uud=203601+ed/46a24611-66601af/t/56.kersion= Expected Result The start screen appears.	4
	Precondition The application is installed on the device. If not, follow directions: <u>https://finaextooksel/install</u>	a
	Postcondition The application has started successfully.	-a
	Attachments	1
Postcondition	The application has started successfully.	
	2 🖉 👁 🌗 🕕 💷 🔗 🦉	A 🛛 🛪

Figure 6.45. An Attachment Reference Replaced by a Preview of the Attachment



Figure 6.46. An Attachment Reference Replaced by a Hyperlink to the Attachment

6.7.2.6. Attachments

You can add any files as attachments to a test case. For more information, see <u>Section 5.2.3.2.6</u>, <u>"Attachments"</u>.

6.7.2.7. Revisions

Test cases may exist in different revisions. For more information, see <u>Section 5.2.3.2.5, "Revisions"</u>.

6.7.2.8. Issues

This view lists all issues that are associated with this test case.

😑 👱 K L A R	OS TEST MANAGEMENT	Finance Tracker 🖬 🗮 🚺 🔍 🕹 🗸
📝 Define	TC00023 - Change a standing order.	④ 合 口 G O
🏩 Plan	Overview Properties User Defined Steps (5) Attachments Revisions Issues (2) Jobs	Save Discard Back s Results (33) Changes
🔯 Execute	New Link	
🕓 Evaluate		Systems under Test Assigned To ♦ Priority ♦ Status ♦ Action
🖌 Configure	PLAYGROUND-22963 Y IM00008 Closed Issue for Test Data PLAYGROUND-22947 Y IM00008 Closed Issue for Test Data, more then one Test Case	hudson Major Closed 🗭 🗹 🖨 — gidley Major Closed 🕼 🖓 🖨 —
	New Link	Systems under Test Assigned To Priority Status Action
	Created 4 years ago by Talal Artf	Last changed 4 years ago by Felix Mustermann
		Save Discard Back

Figure 6.47. The "Issues" Tab

The table shows the following values:

ID	The ID of the issue in the external issue management system.
System	The external issue management system of this issue.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Summary	The summary of the issue.
Created	The date when the issue was created.
Created by	The user who created the issue.
Assigned	The user to whom the issue is assigned.
Priority	The priority of the issue. The possible values are specified by the external issue management system.
Status	The status of the issue. The possible values are specified by the external issue management system.

Actions

The executable actions.

6.7.2.8.1. Creating a new Issue

Clicking New opens the *Issue Details* page (<u>Section 9.6.6</u>, <u>"Issue Details (Creating a new Issue)</u>"), where new issues can be created and linked to the test case. Clicking Link links existing issues to the test case (<u>Section 9.6.7</u>, <u>"Link Issue"</u>).

6.7.2.8.2. Action

The following actions can be performed in the action column:

- 🖉 Edit
- 🖸 Browse
- 前 Delete

6.7.2.9. Jobs





This view lists all the jobs in which this test case is included. These jobs can be executed directly from this view. For more information on managing jobs, see <u>Section 7.1, "Jobs"</u>.

6.7.2.10. Results

The results tab is further divided into a *Test Runs, Test Case Results* and a *Test Suite Results* tab, showing the test results related to this test suite as described in <u>Section 5.2.3.2.7, "Test Runs and Results"</u>.

6.7.2.11. Changes

The tab *Changes* shows the change history of this test case.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

6.8. Test Suites

A *test suite* is a collection of test cases that are to be executed together. The results of a test suite execution can be used to verify and ensure that a system meets certain specifications or requirements.

≡ 📌 K L A R	OS TEST	MAN	AGEMENT						Finance Tracker 🛛 📑		۹ ،	< 0~ ≛~
📝 Define	Test Suites											
±®≛ Dian	New										Save	Discard
	+ @ @ %	8 Û		7 Entries - Page 1 of 1	М	ا 🕨 🕨	▶ ₩ [10 🗸	品又	Show all		Q × & ≡
💏 Evecute	□ ID\$ (Action
LACOULC	TS00008	× 1.0	Check the connectio	n to the server				Fir	nance Tracker 2.1.0		1	┏┖◓◍
A	TS00007	× 1.0	Edit standing orders								3	₡₽₽₫
C Evaluate	TS00006	🥥 1.0	Overview								7	┏┍₽ΰ
	TS00004	1.0	Connection					Fir	nance Tracker 1.0.0		2	₡₽₽₫
🔑 Configure	TS00003	🔺 1.0	Display activities of o	different time periods							4	┏┍₽ΰ
	TS00002	8 1.0	Bank Transfer								8	┏┍₽ΰ
	TS00001	8 1.0	Update Dashboard								4	ⅆⅅÅ℔
	ID	Revis	ion	Name					System under Test		Test Cases	Action
	b laure										0	Discust.
	New										Save	Discard

Figure 6.49. The "Test Suites" Page

6.8.1. Overview Page

The overview page displays all existing test suites in the selected project in a table. New test suites are created here.

The table shows the following values:

ID	Assigned automatically.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Revision	The revision of the test suite.
Name	The name of the test suite.
System under Test	The systems under test assigned to this test suite.
Test Cases	The number of test cases that this test suite contains.
Action	The actions that can be performed.

Name and System under Test can be edited directly in the table rows.

6.8.1.1. Creating a new Test Suite

By clicking on the button New a new empty table row will appear. Now, *Name* and *System under Test* can be defined.

With <u>Save</u> the new test suite is created and saved. The ID of the test suite (TS00001) is automatically assigned by Klaros-Testmanagement. Click on the ID *TS00001* to get to the detail page of the test suite.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



Unsaved Changes

All rows marked with red IDs contain changes that have not yet been saved.

6.8.1.2. Actions

The following actions can be performed in the action column:

- 🕜 Edit
- Duplicate
- Open print view
- fi Delete

If a test suite has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted test suites, the following actions are available instead of *Delete*:

- Restore (only Administrator)
- ${\displaystyle \bigotimes} \,$ Irrecoverably remove the test suite from the database

6.8.1.3. Bulk Actions

On the *Test Suite* page, one or more test suites can be selected for bulk actions. Bulk actions are described in <u>Section 5.2.3.1.5, "Bulk Actions"</u>.

The following bulk actions are supported for test suites:

- + Assign Test Cases
- 🕜 Edit
- Duplicate
- Create new Revision
- Open print view
- 前 Delete
- Restore (only Administrator)
- ⊘ Irrecoverably remove the test suite from the database (only Administrator).

For additional information, see Section 5.2.3.1.5, "Bulk Actions".

6.8.1.4. Table Operations

The following operations can be performed in the line above the table on the right:

- 品 Categorize
 マ Filter / Sort
 Show all / Only active (only Administrator)
 Q Search
 公 Export
 - \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

6.8.2. Detail Page

Each test suite has its own detail page with several additional tabs. Clicking on the ID of the relevant execution definition or on the \square icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Overview* tab.

The following tabs are available: Overview, Properties, User Defined, Attachments, Revisions, Jobs, Results and Changes.



Warning Sign in the ID Column

A manual test case which that contains no steps is not executable. To indicate this, a warning sign (\triangle) is displayed in the ID column.

6.8.2.1. Actions

On the detail pages there are additional icons in the upper right corner. The following actions can be performed here:

Review	Creates a review job.
읍 Open print view	Here, a print-ready view of all entries of the respective test suite can be created. Click on the 🕒 icon to open it in a new browser tab.
	For more information, see <u>Section 5.2.3.2.1, "Print Pages"</u> .
☐ Create Bookmarks	Each individual detail page can also be reached directly via a hyperlink. By clicking on the icon 🔲 this link is copied to the clipboard.
	The creation of bookmarks is described in detail in <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".



Use the green arrows at the very top right to switch between the test suites present on the previous page.

6.8.2.2. Overview

The following values are displayed on the overview tab:



Figure 6.50. The "Overview" Tab - Test Suite



Assigning a System Under Test

The assignment of a test suite to a system under test only indicates this to the user. Assigned test-suites can still be executed with other systems under test.

Latest test runs	This graph shows the number of passed, failed, erroneous, in- conclusive and skipped test case results for the latest five test runs for this test suite.
Execution Times of this Test Suite	This graph shows the minimum, maximum and average exe- cution times for this test suite.
System under Test Overview	This table shows the latest result of this test suite for every system under test and test environment this test suite has been executed in. Hovering the mouse cursor over a single re- sult will show a more detailed overview of the individual test

case results of this test suite. Clicking on a result takes you to the test case result detail page.

6.8.2.3. Properties

In this tab (Figure 6.22) the following attributes of the test suite are displayed:



Figure 6.51. The "Properties" Tab

In this view, test cases can be added to or removed from the test suite. The upper table displays the test cases contained in the test suite, it supports reordering via drag and drop operations. The lower table shows the available test cases.

The following bulk actions are supported for the list of test cases in the test-suite:

- ₩ Remove multiple occurrences
- Remove test cases

This tab allows the user to view following attributes of the test cases:

ID	Assigned automatically.
Name	The name of the test case.
Traceability	Die Anforderung, die dieser Testsuite zugeordnet ist.
Execution	Manual oder Automated.
State	The state of the test case. Possible values are Locked, Approved, Draft and Skip.

Steps

The number of associated test steps.

Action

Entfernt oder fügt den Testfall zur Testsuite hinzu.

You can create your own fields to meet individual requirements. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

6.8.2.5. Attachments

You can add any files as attachments to a test suite. For more information, see <u>Section 5.2.3.2.6,</u> <u>"Attachments"</u>.

6.8.2.6. Revisions

Test suites may exist in different revisions. For more information, see <u>Section 5.2.3.2.5, "Revi</u>sions".

6.8.2.7. Jobs

= 📌 K L A R (OS TEST MANAGEMENT	Finance Tracker 🗧 📑	વ× 0/ ≗ ×
📝 Define	TS00007 - Edit standing orders		④ 🖨 🗋 😌 ᢒ
📇 Plan	Overview Properties User Defined Attachments Revisions Jobs (11)	Save Results (23) Changes	Discard Back
🔹 Execute	11 Entries - Page 1 of 2 📕 🖣 🧎 2	▶ N 10 ▼	∇ Q × & Ξ
	ID ♦ Summary ♦	Priority ♦ Status ♦ Progress ♦ Success ♦	Due ♦ Assignee ♦ Action
🕒 Evaluate	JOB00349 Edit standing orders	 Major Resolved 100% 100% 	25.10.2020 Tanja Toppler 🕜 🖨 🌲
,	JOB00326 Edit standing orders	Major Resolved 100% 66%	25.10.2020 Tanja Toppler
Configure	-IOR00263 Edit standing orders	Major Resolved 100% 66%	25.10.2020 Thomas Tafel
	JOB00241 Test editing standing orders	▲ Major Reopened 0% 0%	25.10.2020 Thomas Tafel 📝 🖨 🍰
	JOB00219 Edit standing orders	Major Resolved 100% 100%	25.10.2020 Thomas Tafel 📝 🖨 🏖
	JOB00192 Test editing standing orders	 Major Resolved 100% 0 % 	25.10.2020 Thomas Tafel 📝 🖨 🌲
	JOB00169 Edit standing orders	Major Resolved 100% 0 %	25.10.2020 Timo Tunklik 🛛 🖉 🖨 🌲
	JOB00146 Edit standing orders	Major Resolved 100% 100%	25.10.2020 Tanja Toppler 🕜 🖨 🌲
	JOB00104 edit standing orders	 Major Resolved 100% 100% 	25.10.2020 Tanja Toppler 📝 🖨 🏖
	Created 4 years ago by Talal Arif		Last changed 8 years ago by Talal Arif
		Save	Discard Back
		0010	Diodard Duon

Figure 6.52. Jobs

This view lists all the jobs in which this test suite is included. These jobs can be executed directly from this view. For more information on managing jobs, see <u>Section 7.1, "Jobs"</u>.

6.8.2.8. Results

The results tab is further divided into a *Test Runs, Test Case Results* and a *Test Suite Results* tab, showing the test results related to this test suite as described in <u>Section 5.2.3.2.7, "Test Runs and Results"</u>.

^{6.8.2.4.} User-Defined

6.8.2.9. Changes

The tab Changes shows the change history of this test suite.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

Chapter 7. Plan



Only available in Klaros-Testmanagement Enterprise Edition

This chapter describes the *Plan* section in Klaros-Testmanagement. Here <u>Jobs</u> can be created and edited.

7.1. Maintain Jobs

A *job* is used to plan, organize and track a test activity. A job may consist of the execution or review of a test case or test suite or contain the textual description of any other possible activity. This chapter shows how to create and edit jobs.

7.1.1. Overview Page

The overview page displays all the jobs present in the selected project in a table. New jobs are created here.

≡ 👱 K L A R	OS TEST I	MANAGEMENT			Finance Tracker		५ ४ 0 ४ ≗ ४
🕑 Define	Maintain Jobs						
😕 Plan	New					Save	Discard
	> ℃D A D	0	22 Entries - Page 1 of 3	3 🕅 ┥ 🚺 2	3 🕨 🕅 10 🗸	Show all	Q × ♣ ≡
🌣 Execute	🔲 ID 🗘	🔉 🛛 Summary 🗢	System under Test 🗢	Priority 🗢 Status 🗢	Test Cases 🗢 Progress 🗢 Succes	ss 🗢 🛛 Due 🗢 🔹 Assignee 🗢	Action
🕒 Evaluate	JOB00360	D Execution TC00022 - Update dashboard by adding a standing order.	Finance Tracker 2.0.0	 Major Reopened 	1 0%	8/25/2020 Markus Meyer	• ໕ຏຨຓ 🏜
🖌 Configure	JOB00359	Execution TC00021 - Update dashboard by cancelling a standing order.	Finance Tracker 2.0.0	▲ Major New	1 0%	8/7/2020 Markus Meyer	▷ Ư ़ ≙ û 4
	JOB00358	Execution TC00020 - Update dashboard by cancelling a debit mandate.	Finance Tracker 2.0.0	▲ Major New	1 0%	10/26/2020 Sandra Selen	> C C Ə û 🎝
	JOB00357	Execution TC00017 - Delete debit mandate.	Finance Tracker 2.0.0	 Major New 	1 0%	10/26/2020 Sandra Selen	D C C 🖯 🛍 🏞
	JOB00356	Execution TC00016 - Delete standing orders.	Finance Tracker 2.0.0	▲ Major New	1 0%	8/30/2020 Markus Meyer	D C C 🛱 û 🏜
	JOB00355	Execution TC00015 - Add a standing order.	Finance Tracker 2.0.0	 Major Resolved 	1 100%	10/26/2020 Felix Mustermann	D C C 🛱 û 🌡
	JOB00354	Review TC00017 - Delete debit mandate.	Finance Tracker 2.1.0	ጵ Critical Reopened	1 0%	10/26/2020 Sabrina Gidley	D C C 🛱 🛍 🛓
	JOB00240 (65)	Test all functions for release2.1.0	Finance Tracker 2.1.0	☆ Critical In Progress	79 84% 76%	10/26/2020 Markus Meyer	> C C 🛱 🛱 🌣
	JOB00218 (21)	Test all functions on a smartwatch	Finance Tracker 2.1.0	Solution In Progress	29 86% 80%	10/26/2020 Markus Meyer	▸┏╘╘ฃѻ
	JOB00172 (43)	Test all basic functions for the next release SUT	Finance Tracker 2.0.1	☆ Critical In Progress	54 90% 80%	10/26/2020 Markus Meyer	▷ ໕∁⋛⋔‡

Figure 7.1. The "Maintain Jobs" Page

The table shows the following values:

ID	Assigned automatically.
Summary	The summary of the job.
System under Test	The test system assigned to this job.
Priority	The priority of the job. Possible values are <i>Trivial</i> , <i>Low</i> , <i>High-Critical</i> und <i>Blocker</i> .
Status	The status of the job. Possible values are New, Reopened, In Progress, Resolved, Closed and Rejected.
Test Cases	The number of test cases assigned to this job and its sub-jobs.

Progress	The percentage of executed test cases of this job and its sub- jobs.
Success	The success rate of the executed test cases of this job and its sub-jobs in percent.
Due	The date on which this job is due to be finished.
Assigned	The user to whom the kob is assigned.
Action	The actions that can be performed.

7.1.1.1. Creating a new Job

By clicking on the button New a new empty table row will appear. Now, a *Summary* can be defined.

With <u>Save</u> the new job is created and saved. The ID of the job (JOB00001) is automatically assigned by Klaros-Testmanagement. Click on the ID *JOB00001* to get to the detail page of the job.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



Red IDs

All rows with red IDs have been changed and are not yet saved!

7.1.1.2. Actions

The action column is located on the far right of the table. The following actions can be performed here:

- Subordinate job
- Edit, (see <u>Section 7.1.2, "Job Details"</u>)
- 🖸 Clone
- Open print view
- 前 Delete

✿ Execute, ♣ Execute manually, ▲ Execute Review, ♣ Import Test Results, ♣ Reopen or execute (see Section 7.1.1.6, "Executing Jobs")

If a job has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted jobs, the following actions are available instead of *Delete*:

- Restore (only Administrator)
- \bigtriangleup Irrecoverably remove the job from the database

7.1.1.3. Bulk Actions

Some actions can be applied to multiple jobs at the same time. To do this, select the jobs to which the action is to be applied in the leftmost column.

The following bulk actions are supported for jobs:

- Subordinate jobs to another job
- 🕜 Edit
- 🖸 Clone
- Open print view
- 🗊 Delete
- Restore (only Administrator).
- ☆ Irrecoverably remove the job from the database (only Administrator).
- O Assign jobs to an iteration.

Bulk actions are described in detail in Section 5.2.3.1.5, "Bulk Actions"

7.1.1.4. Table operations

The following operations can be performed in the line above the table on the right:

- √ Filter / Sort
- Q Search
- & Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

7.1.1.5. Arranging Sub-Jobs

Sub jobs can be rearranged using the arrow icons in the *Action* column. The \uparrow icon opens a dialog that allows the user to move the job to a new position. The \triangleleft and \triangleright icons change the job hierarchy. The \triangleleft icon moves the job to the hierarchy level of the parent job. The \flat icon opens a dialog where a new parent job can be selected.

Parent jobs have the following characteristics:

- A parent job cannot be assigned any test case or test suite.
- A parent job can contain other parent jobs with additional sub jobs.
- If the *b* icon of a job is grayed out, no target parent job is available for subordination.

To subordinate a job to another job, click on the \diamond icon. Now you can select which job this job should be subordinated to. The number of subordinate jobs is then displayed next to the ID of the

parent job. The displayed success and progress rate of the parent job is automatically derived from the test results of its sub jobs.



Warning

Jobs that already have a test case or test suite assigned to them cannot contain sub jobs and are therefore not shown as possible targets in the *New parent job* dialog.



Changes to parent jobs

Changes to parent jobs regarding test environment, system under test and iteration are automatically propagated to their sub jobs.

7.1.1.6. Executing Jobs

For a detailed description on how to execute jobs, see Section 8.1.3, "Executing a Job".

7.1.2. Job - Details

Each job has its own detail page with several additional tabs. By clicking on the ID of the relevant job or on the \Box icon on the right in the action column, you can access the detail view that was selected last. When called for the first time, this is the *Overview* view.

The following tabs are available: Overview, Properties, User Defined, Dependencies Attachments, Results, Comments, Work Log and Changes.

7.1.2.1. Actions

On the detail pages there are additional icons in the upper right corner. The following actions can be performed here:

\$	2 0	.	-5	20	Execute	Click on the displayed icon to execute the job.	
						The execution of jobs is described in detail in <u>Section 7.1.1.6,</u> <u>"Executing Jobs"</u>).	
8	Оре	n prin	ıt viev	N		A print-ready view of the job can be created here. With a click on the icon \bigoplus this opens in a new browser tab.	
						Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .	
	Crea	ate Bo	okma	arks		Each individual detail page can also be reached directly via a hyperlink. By clicking on the icon \Box this link is copied to the clipboard.	
						The creation of bookmarks is described in detail in <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".	
G	Ð	Brov	vse			Use the green arrows at the very top right to switch between the jobs present on the previous page.	

7.1.2.2. Job Overview



Figure 7.2. The "Overview" Tab

Туре	The job type (<i>Test Execution - Automated, Test Execution - Manual, Review</i> or <i>Task</i> (if neither a test case nor a test suite is assigned to the job).
Status	The status of the job.
Priority	The priority of the job.
Assignee	The person responsible for this job.
Start Date	The earliest point in time this job can be executed by a tester.
Due Date	The point in time when the execution of the job has to be com- pleted at the latest.
Test Cases	The total number of test cases of this job or its subjobs.
Test Case / Test Suite	The test case or test-suite assigned to this job.
System under Test	The system under test which should be used when executing this job.
Test Environment	The test environment which should be used when executing this job.
Estimated Duration	The estimated duration needed to complete this job.
Progress	The progress rate shows how many test cases which are cov- ered by this job have been executed, regardless the result.

Success	The success rate shows how many of the test cases which are covered by this job have been successfully executed in the latest test run.
Total Time spent	The time spent on the test execution. This value results from the sum of the execution times from the work log.
Latest Executor	The user who executed the job most recently.
Latest Test Runs	This graph displays the accumulated test case results of the latest test runs for this job in descending order.
Execution Time	This graph shows the minimum, average and maximum exe- cution times for this job.
Latest Test Runs	This table shows the latest completed test runs for this job.
Note	

By default, this table shows only up to five entries. Pressing the Show all button opens up the results tab in which all test runs are listed.

7.1.2.3. Properties

In the *Properties* tab, the following attributes can be changed:

Туре	The job type (<i>Test Execution - Automated, Test Execution - Manual, Review</i> or <i>Task</i> (if neither a test case nor a test suite is assigned to the job).
Summary	The summary of the job.
Description	The detailed description of the job.
Priority	The priority of the job. Possible values are, in ascending order of severity, <i>Trivial</i> , <i>Minor</i> , <i>Major</i> , <i>Critical</i> and <i>Blocker</i> .
Status	The status of the job. Possible values are New, In Progress, Resolved, Closed, Reopened and Rejected.



Figure 7.3. Status Graph of a Job

Progress	The current percentage of completion of the job.
Estimated Duration	The estimated time in days, hours, minutes and seconds (For- mat: Days HH:MM:SS) that the job will take.
Start Date	The earliest time at which this job can be performed by a tester.
Due Date	The time at which the execution of the job must be completed at the latest.
Assignee	The user responsible for the job.
Iteration	The iteration to which this job is assigned. A job can only be in one iteration at a time. Changing the iteration is possible at any time.
Test Case	The test case to be executed in this job. A job can contain either a test case or a test-suite. This field is not available if the job contains subjobs or is of the type <i>Task</i> .
Test Suite	The test suite to be executed with this job. A job can contain either a test case or a test suite. This field is not available if the job contains subjobs or is of type <i>Task</i> .
System under Test	The system under test for which this job should be executed.

Test Environment

The test environment in which this job should be executed.

≡ 🞐 K L A R (OS TEST MANAGEMENT Finance Tracker 🖬 🖬	≅Q× ₽ ×
🕑 Define	JOB00360 - Execution TC00022 - Update dashboard by adding a standing order.	🛃 🖨 🗋 🤤 🕄
😤 Plan	Overview Properties User Defined Dependencies (2) Attachments Results Comments Work Log Changes	Save Discard Back
🔅 Execute	ID JOB00360 Type Test Execution V	
🕒 Evaluate	Execution Manual Summary Execution TC00022 - Update dashboard by adding a standing order.	
€ Configure	Description Tests the automatic update of the dashboard by pressing the Add standing order button to add a new standing order. The standing order should have following information: • Beneficiary of payment:Max Mustermann • IBAN: DE19 1234 1234 1234 1234 1234 12 • Amount € 100,- • Time intervalmonthly	
	mane sure une app is instance union resulty.	
	Priority Major V A Status Reopened V	

Figure 7.4. The "Properties" Tab

A test case or test suite can be assigned to a job, provided that the job does not contain any subjobs. A click on the icon opens a dialog in which a test case or test suite can be selected. Only one item can be selected at the same time. Clicking the icon removes the element from the job.

7.1.2.4. User Defined

You can create your own fields to meet individual requirements. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

7.1.2.5. Dependencies

In the *Dependencies* view, dependencies can be defined between jobs so that certain jobs can only be executed when previously defined conditions are met. For each job, any number of dependencies can be defined to one or more other jobs that must be fulfilled before the job can be executed. In this view you will find an overview of all prerequisite and dependent jobs of the currently selected jobs.

Plan

≡ 🞐 K L A R C) S TEST MANAGEMENT Finance Tracker 🖬 🗮	२ × 0 × ≛ ×
📝 Define	JOB00360 - Execution TC00022 - Update dashboard by adding a standing order.	🛃 🖓 🕄 🖓
📇 Plan	Overview Properties User Defined Dependencies (2) Attachments Results Comments Work Log Changes	Save Discard Back
🏟 Execute	Job dependencies	
	2 Entries - Page 1 of 1 🔰 🕨 🕅 🚺 🗸	7 & ≡
C Evaluate	Job V Summary Pro IOR00259 ID Execution TC00020 Update declapated by concelling a debit mandate	pgress ≑ Success ≑ Result ≑ Status ≑ Action
🖌 Configure	JOB00359 00 Execution TC00021 - Update dashboard by cancelling a standing order.	100% 0% Resolved 🖉 -
	Add Dependency Created 4 years ago by Felix Mustermann	Last changed 4 years ago by Felix Mustermann
		Save Discard Back

Figure 7.5. The "Dependencies" Tab

= 👱 K L A R	o s test managem	ENT	Finance Tracker 🖬 🚆 🔷 🗘 🖓 🗸	.
📝 Define	JOB00360 - Execution TC00022	- Update dashboard by adding a standing order.	20 음 지	60
😤 Plan	Overview Properties User Defi	ined Dependencies (2) Attachments Results Commen	Save Discard B	ack
🕸 Execute	Job dependencies	Job JOB00357 - JOB00357 Execution TC00017 V		
C Evaluate	Job ◆ ♀ JOB00358 00 Execution TC00020 JOB00359 00 Execution TC00021	Progress 0%	Progress € Success € Result € Status € 100% 100% Pased Resolved 100% 0% Resolved	Action C C C C C C C C C C C C
Conligure	Add Dependency Created 3 years ago by Felix Mustermann	Success 0%	Last changed 3 years ago by Felix M	ustermann
		Skipped Falled Error Inconclusive	Save Discard B	ack
		Status New Reopened In Progress Resolved Closed Rejected		
		Add	Cancel	

Figure 7.6. The "Add Dependency" Dialog

Dependency Criteria

Job	The prerequisite job.
Progress	The minimum progress rate of the prerequisite job (0-100%).
Success	The minimum success rate of the prerequisite job (0-100%).
Latest Result	The latest result of the prerequisite job (<i>Passed, Failed, Error, Inconclusive, Skipped</i> or <i>Unknown</i>). Multiple values can be selected.

Status

The status of the prerequisite job. Possible values are *New*, *In Progress*, *Resolved*, *Closed*, *Reopened* and *Rejected*.

If a job cannot be executed because the dependency criteria of its prerequisite jobs are not fulfilled, the \mathcal{Q}_{\bullet} icon is displayed instead of the execute icon.

7.1.2.6. Attachments

You can add any files as attachments to a job. For more information, see <u>Section 5.2.3.2.6, "At-tachments"</u>.

7.1.2.7. Results

The results tab is further divided into a *Test Runs*, *Test Case Results* and a *Test Suite Results* tab, showing the test results related to this job as described in <u>Section 5.2.3.2.7</u>, "<u>Test Runs and Results</u>".

7.1.2.8. Comments

After clicking the Comment button, comments can be added here.

\equiv 👱 KLAR	OS TEST MANAGEMENT Finance Tracker 🖬 🗮	Q × Q × ≗ ×
📝 Define	JOB00360 - Execution TC00022 - Update dashboard by adding a standing order.	🛃 🖨 🛛 😋 🗲
😤 Plan	Overview Properties User Defined Dependencies (2) Attachments Results Comments Work Log Changes	Save Discard Back
🏟 Execute	0 Entries-Page 1 of 1 📕 📕 🕅 🔽	7 & ≡
🕒 Evaluate	Created By ¢ Description \$ No entries available	Changed 🗢 Action
🖋 Configure	Add Comment	
	Created 4 years ago by Felix Mustermann	Last changed 4 years ago by Felix Mustermann
		Save Discard Back
	Add a comment	
	Comment This job needs an update	
	OK Cancel	

Figure 7.7. The "Add a Comment" Dialog

The text entered in this dialog will be saved as a comment on the job once the Add Comment button is clicked and saved. To modify the comment, use the 😰 icon, to remove it, click the 前 icon.

7.1.2.9. Work Log

Each test run related to this job is displayed in the *Work log* tab. It is also possible to add your own additional work logs to the job. To do this, click on the Book time button. A dialog for entering the details will then be displayed.

Plan

= 🔶 K L A R (DS TEST MANAGEMENT Finance Tracker 🖬 🗮	α × 0 γ ≜γ
📝 Define	JOB00360 - Execution TC00022 - Update dashboard by adding a standing order.	🛃 🖓 🕄 🕄
📇 Plan	Overview Properties User Defined Dependencies (2) Attachments Results Comments Work Log Changes	Save Discard Back
🏟 Execute	Total Time spent 0 ms	マ み 一
🕒 Evaluate	Created By Finished Duration Estimated Duration Left Test Run Work description Work description Work description	Changed 🗢 Action
🔑 Configure	Log work	
	Created 4 years ago by Felix Mustermann	Last changed 4 years ago by Felix Mustermann
	Work description Prepared test host	Save Discard Back
	Finished 5/7/2024 🗰	
	Estimated Duration Left 00:00.00	
	Progress 0%	
	Jaire Laiver	

Figure 7.8. The "Work Log" Dialog

A duration must be entered in the *Duration* field in order for the activity to be logged. Then click Save to save your logged work. To modify a work log entry, use the *C* icon to remove it, click the *C* icon.

7.1.2.10. Changes

The page Changes shows the change history for this job.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

7.2. Jobs from Test Cases

On this page, jobs can be created to run or review selected test cases.

= 📌 K L A R	os test ma	NAGEMENT	Finance Tracke		વ × છ	× ≗×
📝 Define	Jobs from Test Cas	ses				
🙁 Plan				Schedule Execution	Schedule Revi	BW
_		24 Entries - Page 1 of 3 🛛 🖊 🚽 1 2 3	▶ N 10 ▼	#1	Q >	< & ≡
🏟 Execute	ID 🗘 Revision	Name 🗢	Traceability 🗢	Priority 🗢	Status 🗢 Execution 🖨	Steps 🖨
•	TC00024 1.0	Check the connection to the server	Requirement 1.2.4 in testplan_16.txt	High	Draft Automated	0
Evaluate	TC00023 1.0	Change a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft Manual	5
	TC00022 1.0	Update dashboard by adding a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft Manual	6
	TC00021 1.0	Update dashboard by cancelling a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft Manual	6
Configure	TC00020 1.0	Opdate dashboard by cancelling a debit mandate.	Requirement 1.2.4 in testplan_16.txt	Medium	Draft Manual	0
	TC00019 1.0	Connection to the database.	de.verit.financetracker.web.Connection	High	Draft Automated	0
	TC00018 1.0	Delete debit mendate	Beguirement 1.2.4 in testplan 16 txt	Modium	Draft Manual	5
	TC00017 1.0	Delete standing orders	Requirement 1.2.4 in testplan_10.txt	Medium	Draft Manual	5
	TC00015 1.0	Add a standing order	Requirement 1.2.4 in testplan_16.txt	Medium	Draft Manual	5
	ID Bevision	Name	Traceability	Priority	Status Execution	Stens
	ib neusion	- Northe	Huccubinty	Thomy	Orardo Execution	oteps
				Schedule Execution	Schedule Revi	ew



149

The jobs for the selected test cases can be created by clicking the Schedule execution or Schedule review button. A dialog will open where a description can be specified. In addition, it can be selected to whom this job is assigned and information about the priority, the test system to be used, the test environment, and the start and due dates.

≡ 9	KLAR	OS TES	τ ΜΑ	NA	GEMENT				Finance 1	fracker 📔				५ × 😯	~ ≗ ~
2 D	efine	Jobs from	Test Ca	ses											
										Sc	chedule Execution	n		Schedule Revie	w
- 📇 P	Plan	24 of 24 selecte	•d			24 Entries - Page 1 of 3	I 1 2	3	N 10 V			₽ \		0	- 4 =
-			Revision			Name 🖨			Traceability	÷	P	riority 🗢	Status	Execution	Stens 🖨
Ф 5	xecute	✓ TC00024	10	Check	the connection to the se	erver		Requireme	ent 1 2 4 in testolan 16 t	tyt	н	liah	Draft	Automated	0
		✓ TC00023	1.0	Chang	e a standing order.			Requirem	ent 1.2.4 in testplan 16.1	txt	M	1edium	Draft	Manual	5
🛟 E	valuate	✓ TC00022	1.0	Update	e dashboard by adding a	standing order.		Requireme	ent 1.2.4 in testplan_16.1	txt	м	ledium	Draft	Manual	6
		✓ TC00021	1.0	Update	e dashboard by cancellin	a standing order.		Requireme	ent 1.2.4 in testolan 16.1	txt	M	ledium	Draft	Manual	6
/ c	Configure	✓ TC00020	1.0	Upda	Schedule Execution Job	s					м	ledium	Draft	Manual	6
, in the second se	Ŭ	✓ TC00019	1.0	Conn	Description						н	ligh	Draft	Automated	0
		✓ TC00018	1.0	Conn							н	ligh	Draft	Automated	0
		✓ TC00017	1.0	Delet							M	ledium	Draft	Manual	5
		TC00016	1.0	Delet	Assignee	Sandra Selen		~			м	ledium	Draft	Manual	5
		TC00015	1.0	Add a	Priority	Major 🗸					M	ledium	Draft	Manual	5
		ID	Revision		,	inajoi						Priority	Status	Execution	Steps
					System under Test		~				adula Execution			Schodulo Povic	NAZ
					Test Environment		~				ledule Execution			Schedule Nevie	**
					Start	曲									
					Due	曲									
					Parent . Joh					v					
					T di citt 000					•					
									OK Cance	el					



≡ 👱 K L A R	o s test manag		Finance Tracker 🔤 🗮	α × 0γ ≗γ
📝 Define	Jobs from Test Cases			
📇 Plan			Schedule Exect	ution Schedule Review
	24 of 24 selected	24 Entries - Page 1 of 3 🔰 🖣 1 2	3 🕨 🕅 10 🗸	옮♡ <
🏚 Execute	ID 🗘 Revision	Name 🗢	Traceability 🗢	Priority 🗢 Status 🗢 Execution 🗢 Steps 🗢
-	✓ TC00024 1.0 Check t	the connection to the server	Requirement 1.2.4 in testplan_16.txt	High Draft Automated 0
Evaluate	✓ TC00023 1.0 Change	e a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium Draft Manual 5
	✓ TC00022 1.0 Update	e dashboard by adding a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium Draft Manual 6
	✓ TC00021 1.0 Update	e dashboard by cancelling a standing order.	Requirement 1.2.4 in testplan_16.txt	Medium Draft Manual 6
🎾 Configure	✓ TC00020 1.0 Update	e dashboard by cancelling a debit mandate.	Requirement 1.2.4 in testolan 16.txt	Medium Draft Manual 6
	✓ TC00019 1.0 Connec	ction to	tion	High Draft Automated 0
	✓ 1000018 1.0 Connec	ction to Description	stion	High Draft Automated 0
	✓ 1C0001/ 1.0 Delete d	debit m	ĸt	Medium Draft Manual 5
	✓ TC00016 1.0 Delete :	standin	kt	Medium Draft Manual 5
	✓ TC00015 1.0 Add a s	standing	xt	Medium Draft Manual 5
	ID Revision	Assignee Sandra Salan	~	Priority Status Execution Steps
		Janua Selen	Cohodulo Even	cabadula Daviau
		Priority Major V	Schedule Exect	Schedule Review
		Start #		
		Lock lest case 🗸		
			OK Cancel	

Figure 7.11. The "Create a review job" Dialog

If the jobs are to be subordinated to another job, it can be selected here using the drop-down menu.

The newly created jobs appear on the Manage Jobs and My Jobs pages and can be executed there.

7.3. Jobs from Test Suites

On this page, the user can generate jobs to run or review selected test suites.

= 👱 K L A R	o s test management	Finance Tracker 📔 🚆	५ × 0 × ≛×
📝 Define	Jobs from Test Suites		
😤 Plan		Schedule Execution	Schedule Review
	7 Entries - Page 1 of 1 N N N Development	Gusten under Test 🌢	Q × 25 ≡
🏚 Execute	TS00008 1.0 Check the connection to the connection	System under Fest -	Test Cases 🚽
	TS00007 1.0 Edit standing orders	Pinance tracker 2.1.0	3
🕒 Evaluate	TS00006 1.0 Overview		7
	TS00004 1.0 Connection	Finance Tracker 1.0.0	2
🔑 Configure	TS00003 1.0 Display activities of different time periods		4
	TS00002 1.0 Bank Transfer		8
	TS00001 1.0 Update Dashboard		4
	ID Revision Description	System under Test	Test Cases
		Schedule Execution	Schedule Review

Figure 7.12. The "Jobs from Test Suites" Page

The jobs for the selected test suites can be created by clicking the <u>Schedule execution</u> or <u>Schedule review</u> button. A dialog will open where a description can be specified. In addition, it can be selected to whom this job is assigned and information about the priority, the system under test to be used, the test environment, and the start and due dates.

≡ 📌 KLAR	os test m <i>a</i>	A N A G E M E N T		Finance Tracker 📓 🗮	Q x Ø~ &~
📝 Define	Jobs from Test Su	ites			
🙁 Plan				Schedule Execution	Schedule Review
	7 of 7 selected	7 Entries - Pag	e1of1 🕅 ┥ 🚺 🕨 🕅 🔟		ዱ∇ Q × 쇼 =
💏 Evecute	🗹 ID 🗢 Revision	De	escription 🗢	System unde	er Test ♦ Test Cases ♦
	✓ TS00008 1.0	Check the connection to the server		Finance Tracker 2.1.0	1
4	✓ TS00007 1.0	Edit standing orders			3
C Evaluate	✓ TS00006 1.0	Overview			7
	✓ TS00004 1.0	Connection Schedule Execution Jobs		5° 7 1 100	2
🔑 Configure	✓ TS00003 1.0	Display activiti			4
	✓ TS00002 1.0	Bank Transfer Description			8
	✓ TS00001 1.0	Update Dashb			4
	ID Revision	Assigned	Condro Colon		der Test Cases
		Assignee	Sandra Selen	•	Sahadula Paviaw
		Priority	Major 🗸		Schedule Review
		System under Test	~		
		-,			
		Test Environment	~		
		Start	曲		
		Due	莆		
		Parent Joh			
		Talent oob		•	
		Individual job per test case			
				OK Cancel	

Figure 7.13. The "Schedule Execution Jobs" Dialog

Plan

≡ 👱 K L A R	os test manag	BEMENT		Finance Tracker 🖬 📑	વ× છ~ ≛*
🗹 Define	Jobs from Test Suites				
📇 Plan				Schedule Execution	Schedule Review
🏚 Execute	7 of 7 selected ID \$ Revision	7 Entries - Page 1 Des	cription 🗢	System under Test ≑	C × ⅔ ≡ Test Cases ≎
	 ✓ TS00008 1.0 Check t ✓ TS00007 1.0 Edit sta 	he connection to the server nding orders		Finance Tracker 2.1.0	1 3
Evaluate	✓ TS00006 1.0 Overvier ✓ TS00004 1.0 Connect	w		Finance Tracker 1.0.0	7 2
🔑 Configure	✓ TS00003 1.0 Display ✓ TS00002 1.0 Bank Tr	activities of different time periods ansfer Create a review job			4
	ID Revision	Description		System under Test	4 Test Cases
				Schedule Execution	Schedule Review
		Assignee Sandra Selen	~		
		Priority Major V			
		Due 🗮			
			ОК	Cancel	

Figure 7.14. The "Create a review job" Dialog

The *Individual job per test case* selection can be used to specify, whether a separate job should be created for each test case in the test suite. These are then combined under a parent job. Splitting them into individual jobs makes it possible to distribute the execution of the test cases individually among several users.

If the jobs are to be subordinated to another job, this can be selected here using the drop-down menu.

The newly created jobs appear on the Manage Jobs and My Jobs pages and can be executed there.

7.4. Jobs by User

This section is used to track the workload and progress of individual users in the selected project.

On the Jobs by User page the workload, work duration and progress of every user for the current active project is shown. Clicking the Q icon displays the details page (Figure 7.16) for the selected user.

Plan

≡ 📌 K L A R (d s test managemen	ΙT					Finance Tracker 📑		م	× 0~	* ~
🕑 Define	Jobs by User										
🐣 Plan		Name 🖨	19 Entries - Page 1 of 2	K	1 2	► ►	10 ▼ Estimated Duration ♣	Busy from ≜	Busvito. ≜	Progress #	Action
🏟 Execute	Felix Mustermann Claudia Könnnecke					1	00:30:00	8/1/20, 7:00 AM	10/26/20, 12:00 AM	1/1 0/0	Q
民 Evaluate	Max Mustermann Markus Meyer Oliver Krams					40 0	5 Days 13:52:00	8/1/20, 7:00 AM	10/26/20, 12:00 AM	0/0 33/40 0/0	Q
🔑 Configure	Patrick Reilly Sabrina Gidley Sahra Berger					0 18 0	18:00:00	8/1/20, 7:00 AM	10/26/20, 12:00 AM	0/0 17/18 0/0	Q
	Sandra Selen Selen Deutsch	Name				2 0 Jobs	Estimated Duration	8/1/20, 7:00 AM Busy from	10/26/20, 12:00 AM Busy to	0/2 0/0 Progress	Q



7.4.1. Jobs by User - Details

This page displays the workload, work duration and progress for a user in the currently selected project. The user's jobs can either be displayed listed in a table or visualized in a calendar view. Clicking the C icon opens the detail page for the corresponding job. For more information, see <u>Section 7.1.2, "Job - Details"</u>.

≡ 📌 KLAR	OS TEST	MANAGEM	MENT			Finance Tracker		२ @ ४ ≛ ४
📝 Define	Jobs by User:	Markus Meyer						8
😤 Plan	Jo Estimated Durat Ruey fo	obs 40 ion 5 Days 13:52:00			New Reopened			
🔹 Execute	Busy to 4/30/2021 Progress 33/40				Resolved Closed			
🕒 Evaluate	Schedule J	obs			Rejected			
🖌 Configure	Previous N	lext Current Date			April 2021		Month	Week Day Agenda
	Su	n	Mon	Tue	Wed	Thu	Fri	Sat
	W13	28	29	30	31	1 12a JOB00359 - Execution TC00	2 0021 - Update dashboard by canc	3 elling a standing order.
	W14	4	5	6	7	8	9	10
	JOB00359 - Exec	ution TC00021 - Update da 12a JOB(12a JOB)	shboard by cancelling a 10356 - Execution TCO0 10360 - Execution TCO0	a standing order. 016 - Delete standing orders. 022 - Update dashboard by addin	g a standing order.			
	W15	11	12	13	14	15	16	17
	JOB00359 - Exec JOB00356 - Exec JOB00360 - Exec	ution TC00021 - Update da ution TC00016 - Delete sta ution TC00022 - Update da	shboard by cancelling a nding orders. shboard by adding a sta	a standing order.				
	W16 1: JOB00359 - Ex	18 ecution TC0002	19	20	21	22	23	24
	JOB00356 - Exec JOB00360 - Exec	ution TC00016 - Delete sta ution TC00022 - Update da	nding orders. shboard by adding a sta	anding order.				
	W17 12a JOB00356 -	25 Execution TC00016 - Delet	26 e standing orders.	27	28	29	30	1
	12a JOB00360 -	Execution TC00022 - Upda	te dashboard by adding	a standing order.				
	W18	2	3	4	5	6	7	8

Figure 7.16. The "Jobs by User" Page

Chapter 8. Execute

This chapter describes the processes of executing test cases and test suites and jobs, continuing interrupted test executions and creating issues for failed tests.

8.1. My Jobs

enterprise edition Only available in Klaros-Testmanagement Enterprise Edition

On the *My Jobs* page, the jobs of the active user as well as all unassigned jobs can be executed. (Figure 8.1).



Figure 8.1. The "My Jobs" Page

The My Jobs and Jobs without Assignee tables show the following values:

Project	The project to which the job belongs to.
ID	The automatically assigned ID.
Summary	A summary of the job.
System under Test	The system under test assigned to this job.
Priority	The priority of the job. Possible values are <i>Trivial</i> , <i>Low</i> , <i>High-Critical</i> und <i>Blocker</i> .
State	The state of the job. Possible values are New, Reopened, In Progress, Resolved, Closed and Rejected.
Progress	The percentage of executed test cases of this job and its sub- jobs.

Success	The success rate of this job.
Due Date	The date on which this job is due to be finished.
Action	The actions that can be performed.

8.1.1. Action

The following actions can be performed in the action column:

Edit
Execute
Manually execute
Reopen or update
Execute review
Import results

8.1.2. Table Operations

The following operations can be performed in the line above the tables on the right:

- √ Filter / Sort
- All Projects / Current Project Only
- Q Search
- & Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

8.1.3. Executing a Job

The available action for each job in the tables depends on the job type, whether it contains manual or automated test cases and whether it has dependencies on other jobs.



Non-executable Jobs

If a job cannot be executed, the corresponding icon in the action column is grayed out. Hovering over this icon will display the reason in a tooltip.

🏚 Execute	The job is a task and it contains no test case and no test suite.
Manually execute	The job contains at least one manually executable test case.

Le Reopen or update	The job has already been completed.
▲ Execute review	The job is a review job.
-S Import results	The job only contains automated executable test cases.

8.1.3.1. Executing a Review

Review jobs are used to *Review* test cases or test suites. Clicking the Finish Review button opens a dialog in which the test case or test suite can be approved or rejected.

8.2. Run Test Case

Executing a test case means either executing the specified test steps one by one on a specific version of the system under test in a selected test environment (only for test cases with at least one step) or importing the associated test result files.

≡ 🞐 K L A R	OS TES	t man,	AGEMENT		Finance Tracker 🛛 📑	Q × 0 × ≗×
📝 Define	Run Test C	ase				
				b b b	_	
📇 Plan			24 Entries - Page 1 of 1 🛛 🔍 🔍	P N 30	•	₩Λ σ× ශ Ξ
	ID 🗘 🛇	Revision	Name 🗢	Execution 🖨	Traceability 🗢	Steps 🗢 Issues 🗢 Results 🗢 Action
📅 Execute	TC00024	🛕 1.0	Check the connection to the server	Automated	Requirement 1.2.4 in testplan_16.txt	0 2/1 3 📲
	TC00023	1.0	Change a standing order.	Manual	Requirement 1.2.4 in testplan_16.txt	5 0/2 35 🚑
	TC00022	A 1.0	Update dashboard by adding a standing order.	Manual	Requirement 1.2.4 in testplan_16.txt	6 5/0 24 🌲
🕒 Evaluate	TC00021	1.0	Update dashboard by cancelling a standing order.	Manual	Requirement 1.2.4 in testplan_16.txt	6 0/2 24 🏖
	TC00020	😢 1.0	Update dashboard by cancelling a debit mandate.	Manual	Requirement 1.2.4 in testplan_16.txt	6 0/0 19 🏖
6 Configuro	TC00019	1.0	Connection to the database.	Automated	de.verit.financetracker.web.Connection	0 0/0 0 📲
Je Conligure	TC00018	1.0	Connection to the server of the bank.	Automated	de.verit.financetracker.web.Connection	0 0/0 0 📲
	TC00017	🛕 1.0	Delete debit mandate.	Manual	Requirement 1.2.4 in testplan_16.txt	5 0/0 31 💂
	TC00016	1.0	Delete standing orders.	Manual	Requirement 1.2.4 in testplan_16.txt	5 0/0 43 🏖
	TC00015	🕑 1.0	Add a standing order.	Manual	Requirement 1.2.4 in testplan_16.txt	5 0/0 51 💂
	TC00014	🛕 1.0	Updating the dashboard after a bank transfer.	Manual	Requirement 1.2.4 in testplan_16.txt	5 0/0 14 🏖
	TC00013	1.0	Transfer money without specifying a reference.	Manual	Requirement 1.2.4 in testplan_16.txt	5 0/0 46 🏖
	TC00012	1.0	Transfer money without specifying an amount.	Manual	Requirement 1.2.4 in testplan_16.txt	4 0/0 44 🙇
	TC00011	1.0	Transfer money without specifying the IBAN.	Manual	Requirement 1.2.4 in testplan_16.txt	4 0/0 31 🏖
	TC00010	🥑 1.0	Transfer money without specifying a recipient.	Manual	Requirement 1.2.4 in testplan_16.txt	4 0/0 38 🚑
	TC00009	A 1.0	Display activities of the last three months.	Manual	Requirement 1.2.4 in testplan_16.txt	3 0/0 24 🌲
	TC00008	🛕 1.0	Display activities within user-specified time period.	Manual	Requirement 1.2.4 in testplan_16.txt	4 0/0 26 🎝
	TC00007	1.0	Display the activities of last month.	Manual	Requirement 1.2.4 in testplan_16.txt	3 0/0 23 🌲
	TC00006	>⊄ 1.0	Display the activities of last week.	Manual	Requirement 1.2.4 in testplan_16.txt	3 0/0 25 🍰
	TC00005	1.0	Check account balance.	Manual	Requirement 1.2.4 in testplan_16.txt	2 0/0 20 🌲
	TC00004	A 1.0	Transfer money to a foreign bank.	Manual	Requirement 1.2.4 in testplan_16.txt	4 0/0 24 🌲
	TC00003	1.0	Transfer money to a different bank account.	Manual	Requirement 1.2.4 in testplan_16.txt	4 0/0 17 🏖
	террора	. 10	Monay transfor with insufficient funds	Manual	Domisroment 1.0.4 in testalen 16 tut	4 0/0 14 .

Figure 8.2. The "Run Test Case" Page

The Run Test Case page shows all test cases of the selected project in a table.

The table shows the following values:

ID	The automatically assigned ID.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Revision	The revision of the test case.
Name	The name of the test case.
Execution	Manual oder automated.
Traceability	The requirement assigned to this test case.

Steps	The number of associated test steps.
Results	The number of test case results.
Action	The executable actions.

8.2.1. Action

The following actions can be performed in the action column:

- Lo Manually execute
- Import results

8.2.2. Table Operations

The following operations can be performed in the line above the table on the right:

- 品 Categorize
- √ Filter / Sort
- Q Search
- 곬, Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

8.2.3. Executing a Test Case



Manual or Automated Test Execution

If a test case has at least one test step, it can be executed manually. Alternatively, test result files can be imported for all test cases (see <u>Section 8.5, "Import Test Results"</u>. for more information.



Note

Only test cases with the status *Draft* or *Approved* can be executed.

Clicking the 🝰 icon displays a dialog with information about the test case (Figure 8.3).

≡	y KLAR	os test m	ANAGEMENT Pr	ance Tracker 🗃 🛢		ર× છ~	* ~
ľ	Define	Run Test Case					
:2:	Plan	Execute Test Case				QX	& ≡
		1.0 - TC00021,	Update dashboard by cancelling a standing order.		 Issues 	Results	Action
Å	Execute				2/1	3	-5
		— Test Case De	tails		0/2	35	20
			TC00021		5/0	24	
	Evaluate	Povisio	n 10		0/2	24	•.
		Nom	e Undete deebbeerd by concelling a standing order		0/0	19	
ىر	Configure	Nain			0/0	0	- 14
		Descriptio	n lests the automatic update of the dashboard by deleting a standing order.		0/0	31	2.
					0/0	43	
					0/0	51	20
		Dresenditie	Make sure the app is installed befor testing		0/0	14	20
		Freconditio	I make sure the app is instance befor testing:		0/0	46	20
					0/0	44	20
			 The application is installed on the device. If not, follow directions: https://finanztracker/install 		0/0	31	20
			Connection to the server exists		0/0	38	20
					0/0	24	20
					0/0	26	20
		Posteonditio	The dashboard has been undeted		0/0	23	
		Fostconditio	n nie daanboard naa been updated.		0/0	25	
					0/0	20	
					0/0	24	
		Expected Resu	t Dashboard has an entry less for standing orders than by starting the application.		0/0	1/	<u>.</u> 0
			· · · · · · · · · · · · · · · · · · ·		0/0	14	
					leeuo	e Roculte	Action
					Tabue.	5 Hesuits	Action
		Tear	n Team Alpha				
		Statu	s Draft				
_		Test Typ	e Functional				
		Design Techniqu	e Black-Box				
		Variet	y Positive				
_		Lev	- el Component Test				
					•		
				Execute Cancel			

Figure 8.3. The "Execute Test Case" Dialog

Before a test case can be executed, the test environment the test case is run in and the system under test itself have to be selected. In addition, the user-defined properties of the test run can be set here (for more information, see <u>Section 5.2.3.2.4</u>, <u>"User Defined Properties"</u>).

Clicking the Execute button opens a second browser window where the test execution takes place. Make sure that pop-ups are allowed in your browser or add an exception for Klaros-Test-management. By default, a view with step-by-step instructions will be shown, but this can be changed to a tabular view during the execution.

8.2.3.1. The "Step-by-step Instructions" View



Figure 8.4. The "Step-by-step Instructions" View

The step-by-step instructions screen shows the Action, Expected Result, Precondition and Postcondition values of the current test step.

Depending on the results of the step, the user can click the following buttons:

G	Back	Go back to the last test step to repeat it or to edit it.
()	Link Issue	Link an existing issue from an issue management system to this test case.
Ð	Create Issue	Create an issue related to this test case in an issue manage- ment system.
۲	Review	Request a review for this test case.
Ø	Edit	Edit this test step result.
₩	Skip All	Skip all remaining test steps without changing the result of the test case.
M	Skip	Skip the current test step without changing the result of the test case.
*	Inconclusive	Mark the test step result as Inconclusive.
⚠	Failure	Mark the test step result as Failure.
⊗	Error	Mark the test step result as Error.
\oslash	Passed	Mark the test step as successfully completed.

The same process is applied for each test step of the test case.

As soon as all steps have a result or have been skipped, the execution can be finished via a dialog.

Clicking the OK button, causes Klaros-Testmanagement to show the test run result of the test case (Figure 8.7).

8.2.3.2. The "Tabular Step Instructions" View



Figure 8.5. The "Tabular Step Instructions" View

The tabular step instructions screen shows the *Description*, *Expected Result*, *Precondition*, *Postcondition* and result of all steps, the default result being *Unknown*.

The following actions can be performed at each step:

M	Skip	Skip the current test step without changing the result of the test case.
G	Back	Go back to the last test step to repeat it again or to edit it.
۲	Review	Request a review for this test case.
0	Create Issue	Create an issue related to this test case in an issue management system.
()	Link Issue	Link an existing issue from an issue management system to this test case.
0	Pause execution	Pauses the test execution and stops the time measurement.

Skip all remaining steps Finish the test and mark all remaining steps as skipped.

6	-	1	
()
	1	IJ	/
- {		3	
	D	7	

What is the difference between a failure and an error?

A failure is a discrepancy between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition.

An error is the inability of the system to perform the test correctly.

For each error, failure or inconclusive result the following dialog will be shown: (Figure 8.6).

Tabular Vi	iew?	shboard by cancelling a standing or	ler.	S	Contraction to the second seco
F	Step 1 of 6 Result 😢	Reation is installed on the during if	aat fallau diraationaa	https://figenetics.la	- /install
Frecon	Gill the app	incation is installed on the device. If i	iot, tonow directions.	nups.//inanztrackei	/instail
4	Result	Error			
	Summary	App was not installed on the device			
Function	Description	Add a note to the test case to ensu device before testing.	re that the app is inst	alled on the	
Expected r	1. Upload /	Attachments	ОК	Cancel	
Postcon	dition The app	lication has started successfully.			

Figure 8.6. The "Edit step result" Dialog

- Upload Attachment opens a dialog for uploading attachments to the step.
- Finish test opens a dialog asking if the execution should be finished.
- Ok closes the dialog and displays the next step.
- Cancel closes the dialog.



Тір

The Description and Summary fields appear in all comment dialogs.

8.2.3.2.1. Finish Execution

After the last step result of a test case has been entered, a dialog is displayed asking whether the execution should be finished.

Clicking the OK button displays the result of the test run for this test case (Figure 8.7).

In the *Test Run - Overview* view, a summary and description can be entered for the test case result as well as for each individual test step result. (<u>Figure 8.7</u>).

Test Run Ove	rview			
Test Case Result Test Run Start Execution time	Test Case Result TCR0012651 Test Run TRU0009140 Start 4/20/21, 5:25 PM Execution time 00:00:22			
Summary				
Descriptior				
Steps	Upload Attachment			
Results 1	2 3 4 5 6			
Step Ste	ep 1: Start the application by pressing the blue icon b 💙			
Summary				
Description				

Figure 8.7. The "Test Run Overview" Page

0000000	Note
	If an issue management system is configured for the active project, the Link Issue and Create Issue buttons are activated.

8.2.3.3. Creating Issues

During the execution of a test case, issues can be created for this test case in an external issue management system by clicking the ① button. Issues already created in an issue management system can be linked to the test case by clicking the ① button.

For detailed information on creating and linking issues, see <u>Section 9.6.6, "Issue Details (Creating</u> <u>a new Issue)"</u> and <u>Section 9.6.7, "Link Issue"</u>.

8.2.3.4. Creating Review Jobs



If an error is found in a test case description during test execution, a review job can be created by clicking on the text by button. A dialog window opens in which the details of the review job can be entered:

Tabular Vi	Tabular View? Image: Comparison of the standing order. + TC00021 - Update dashboard by cancelling a standing order.				
Step 1 of 6 Precondition The application is installed on the device. If not, follow directions: https://finanztracker/install					
	Create a review jol				
Ļ	Description	Test Case is outdated.			
	Assignee	Felix Mustermann 🗸			
	Priority	Major 🗸			
Expected F	Start	#			
	Due				
	Lock Test Case				
Postcon		OK Cancel			

Figure 8.8. Creating a Review Job

8.3. Run Test Suite

The execution of a *Test Suite* consists of the execution of a defined set of test cases on a specific version of the system under test in a selected test environment.

≡ 👱 K L A R (d s test management	Finance Tracker 🖬 🗮 📃 🔍 🔍 🔍 🔍
🕑 Define	Run Test Suite	
🏩 Plan	7 Entries - Page 1 of 1 K 1 N 10	중
🄹 Execute	TS00008 A ≥ 4 1.0 Check the connection to the server TS00007 0 1.0 Edit standing orders	1 3 - 3 23 2
🕒 Evaluate	TS00006 Image: 1.0 Overview TS00004 1.0 Connection TS00003 1.0 Display activities of different time periods	7 19 2 2 0 - 4 3 •
🖋 Configure	TS00002 0 Display advantes of uniferent time periods TS00002 0 1.0 Bank Transfer TS00001 0 1.0 Update Dashboard	

Figure 8.9. The "Run Test Suite" Page

The Run Test Suite page shows all test suites of the selected project in a table.

The table shows the following values:

ID

The automatically assigned ID.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Revision	The revision of the test suite.
Description	The description of the test suite.
Test Cases	The number of test cases in the test suite.
Results	The number of test case results.
Action	The executable actions.

8.3.1. Action

The action column is located on the far right of the table. The following actions can be performed in the action column:



Import results

8.3.2. Table Operations

The following operations can be performed in the line above the table on the right:

- 윪 Categorize
- √ Filter / Sort
- Q Search
- 옰 Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

8.3.3. Executing a Test Suite



Important

A test suite with at least one test step in one of its test cases can be executed manually. For all other test suites, test result files can be imported (see <u>Section 8.5, "Import</u> <u>Test Results"</u>. for more information.



Warning

Test cases with the states *Locked* and *Skip* or without any steps will be skipped when executing the test suite.

≡ 🞐 K L A R	o s test management	Finance Tracker 🖬 📑	Q x 0 γ ≛γ
📝 Define	Run Test Suite		
北 Plan	7 Entries-Page 1 of 1 N 4 1 N 10 V	よく	
🔅 Execute	ID V Celevision Description TS00008 A 1.0 Check the connection to the server TS00007 >4 1.0 Edit standing orders	, i i i i i i i i i i i i i i i i i i i	1 3 -
🔥 Evaluate	Execute Test Suite 1.0 - TS00007, Edit standing orders		7 19 2 2 0 - 4 3 2
🗲 Configure	- Test Suite Details		8 7 20 4 11 20
	Revision 1.0 Description Edit standing orders		
	ID Revision	Execution 🗢 Status 🗢 Steps 🗢 Action	
	TC00015 1.0 Image: Add a standing order. TC00023 1.0 A Change a standing order.	Manual Draft 5 🛋	
	TC00016 1.0 🔮 Delete standing orders.	Manual Draft 5 🍰	
	Test Environment Android 8 Smartphone V System under Test Finance Tracker 2.1.0 V		
		Execute Cancel	

Clicking the 💄 button displays a dialog with information about the test suite (Figure 8.10).

Figure 8.10. The "Execute Test Suite" Dialog

Before a test suite can be executed, the test environment the test suite is run in and the system under test itself have to be selected. In addition, the user-defined properties of the test run can be set here (for more information, see <u>Section 5.2.3.2.4</u>, <u>"User Defined Properties"</u>).

Clicking the Execute button opens a second browser window where the test execution takes place. Make sure that pop-ups are allowed in your browser or add an exception for Klaros-Test-management.

- TS00007 - Edit standing or	ders		Tes	t Case 1 of 3
Test Suite TS00007				
Name Edit stand	ing orders			
Revision 1.0				
Test Cases 3				
System under Test Finance Tr	acker 2.1.0			
Test Environment Android 8	Smartphone			
Attachments				
- Test Case Overview				
Test Case	TC00015			
Revision	1.0			
Name	Add a standing order.			
Description	Tests the automatic update	ate of the dashboard	by adding a new st	anding order.
Precondition	Make sure the app is ins	talled befor testing	!	
	The application is i <u>nztracker/install</u> Connection to the s	nstalled on the devic server exists.	e. If not, follow direc	tions: <u>https://fina</u>
Postcondition	The dashboard has been	updated.		
Steps	5			
System under Test	Finance Tracker 2.1.0			
Test Environment	Android 8 Smartphone			
Testing internally at company				
Attachments				
		Start	Skip Test Case	Cancel

Figure 8.11. The "Test Case Overview" View

This view shows the overview of the next test case to be executed. Clicking the <u>Start</u> button starts the execution of the test case (the execution of a test case is described in detail in <u>Section 8.2.3</u>, "Executing a Test Case").

8.3.3.1. Skip Test Cases

During the execution of a test suite, individual test cases can be postponed for later execution or permanently skipped.

8.3.3.1.1. Skip Test Cases Permanently

Tabular View?	Jpdate dashboard by cancelling a standir	ng order.	Step 1 of 6
Step Result Precondition Action	1 of 6 ? The application is installed on the devia Start the application by <i>pressing</i> the blue	ce. If not, follow directions: ie icon ^{Con} button.	https://finanztracker/install
Expected Result Postcon	The start screen annears ou sure? This action will finish execution of the	current test case. Do you v	vant to continue? Cancel
	2 🖉 👁 🚺 🕕	• C	o 🕂 8 🛪

Figure 8.12. The "Permanently Skipping a Test Case" Dialog

After permanently skipping a test case, the tester is prompted to enter a reason why the test case was skipped. If explanatory templates are already defined (see <u>Section 10.4.3, "Test Execution"</u>), one of these templates can be selected.

By clicking on the + icon, the selected template is copied to the test case result summary.

Tabular View?	Ŷ
+ TC00021 - Update dashboard by cancelling a standing order.	Step 1 of 6
Step 1 of 6 Result 🕝	
Precondition The application is installed on the device. If not, follow directions: https://finanztrad	ker/install
Permanently Skip	
Please explain why this test case is skipped completely.	
Templates +	
Summary Test server unreachable.	
Expected F	
OK Cancel	
Postcondition The application has started successfully.	

Figure 8.13. The "Inserting a Reason Template" Dialog

0000000	Note		
	Skipping a	ll steps of a test case will mark the whole test case as permanentl	y skipped.
	Test Run Over	view	
	Test Case Result Test Run Start Execution time Summary Description	TCR0012692 TRU0009163 4/21/21, 12:37 PM 00:00:20	
	Steps		
	Results 1	2 3 4 5 6	
	Step Step	p 1: Start the application by pressing the blue icon b 💙	
	Summary		
	Description		

Figure 8.14. The "Test Run Overview" View

After finishing a test run, the tester can also edit the reasons for skipping test cases in the test run overview (see Figure 8.14).

8.3.3.2. Finish Execution

After all test cases have been executed, the *Test Suite Execution Overview* view is displayed. Here, a summary and description can be entered for each individual test case and test step of the test run (<u>Figure 8.15</u>).

+ TS00007 - E	lit standing orders	Test Case 1 of 3
Test Suite Exe	cution Overview	
Test Suite Result Test Run	TSR0001717 TRU0009153	
Execution time	00:00:23	
Results	1 2 3	
Test Case Result	TC00015-TCR0012670 🗸	
Test Case	TC00015 - Add a standing order.	
Summary		
Description		
Results 1	2 3 4 5	
Step Ster	1: Start the application by pressing the 🗙	
Summary	recomplete approach by precoming the s	
Samury		
Description		

Figure 8.15. The "Test Suite Execution Overview" View

8.4. Continue Test Run

If a test run is not completed and must be suspended, execution can be continued later. Execution continues with the first test step that does not yet have a result.

The Continue Test Run page displays all unfinished test runs of the selected project in a table.



Note

The *Testers may only resume their own test runs* setting controls whether users with the *Tester* role can see and resume suspended test runs from other users (<u>Section 10.4.1, "Miscellaneous"</u>).



Figure 8.16. The "Continue Test Run" Page

The table shows the following values:

ID	The automatically assigned ID.
Start	The start time of the test run.
Executor	The name of the user who last executed the test run.
Iteration	The iteration in which the test run is being executed.
Test Case/Suite	The ID of the test case or test suite of the test run.
Test Environment	The test environment in which the test is being executed.
System Under Test	The system under test being tested.
Progress	The number of finished test cases.
Action	The executable actions.

8.4.1. Action

The following actions can be performed in the action column:

Q Display the test case results captured so far of this test run.

2. Display the test case results captured so far of this test run. Continue the execution of the test run

- Open print view.
- Delete this test run.

If a test run has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted test runs, the following actions are available instead of *Delete*:

- Restore this test run.
- ☆ Irrecoverably remove the test run from the database (only Administrator).



Deleting a Test Run

When a test run is deleted, the associated test case results and test suite results are also deleted

Once all test runs for a specific system under test or test environment have been deleted, the respective system under test or test environment may also be deleted. Otherwise, they remain locked for deletion.

8.4.2. Bulk Actions

Certain actions can also be applied to several test runs at once. To do this, select the test runs to which the action is to be applied in the leftmost column.

The following bulk actions are supported for test runs:

- 🖉 Edit the attributes of this test run.
- Open print view.
- Delete
- Restore (only Administrator)
- ⊘ Irrecoverably remove the selected test runs from the database (only Administrator).

For additional information, see Section 5.2.3.1.5, "Bulk Actions".

8.4.3. Table Operations

The following operations can be performed in the line above the table on the right:

√ Filter / Sort

Show all / Only active (only Administrator)

- Q Search
- & Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

8.4.4. Continue test run

By clicking on the 2_{0} in the action column, you get to an overview page to continue the test run. All test case results and, if available, the test suite result of the test run are displayed in tables here.

= 学 K L A R ()S TEST MANAGEMENT	Finance Tracker 👕 🗮	५ × 0 × ≛×
📝 Define	Continue Test Run - TRU0018308 - P00002-TRU0018308		
📇 Plan	1.0 - TS00002, Bank Transfer		
🔅 Execute	ID TS00002 Description Bank Transfer		
🕒 Evaluate	Executor Felix Mustermann Test Environment Android 8 Smartphone System under Test Finance Tracker 2.1.0		
🖌 Configure	ID Revision Na	ne	Executor Time Result
	TC00013 1.0 Transfer money without specifying a reference.		Felix Mustermann 00:00:11 🥑
	TC00012 1.0 Transfer money without specifying an amount.		Felix Mustermann 00:00:07 🔥
	TC00011 1.0 Transfer money without specifying the IBAN.		Felix Mustermann 00:00:04 😣
	TC00010 1.0 Transfer money without specifying a recipient.		Felix Mustermann 00:00:05 🔀
	TC00004 1.0 Transfer money to a foreign bank.		Felix Mustermann 0 ms ?
	TC00003 1.0 Transfer money to a different bank account.		Felix Mustermann 0 ms ?
	TC00002 1.0 Money transfer with insufficient funds.		Felix Mustermann 0 ms 📍
	TC00001 1.0 Transfer money to a local bank account.		Felix Mustermann 0 ms ?
			Continue Back

Figure 8.17. The "Continue Test Run" Page

Clicking the Continue button resumes execution of the test run at the last saved test step.

Clicking the Back button navigates back to the Continue Test Run page (Figure 8.16).

8.5. Import Test Results

In Klaros-Testmanagement test results from various test frameworks can be imported (see <u>Section 8.5.1, "Supported Frameworks"</u>) for a complete list of supported formats). If the associated test cases or test suites are not already defined in Klaros-Testmanagement, they can be created automatically during the import.

To start the import, an XML file with results from a supported test framework must be uploaded. When the file upload is completed it is mandatory to select the system under test and the test environment these results should be assigned to. The format of the result file is typically detected automatically.

≡ 🙅 K L A R	o s test management		Finance Tracker 👕 🗮	५ × 0 × ≗ ×
📝 Define	Import Test Results			
🙁 Plan	Upload a test result file to continue			×
🔅 Execute	Upload Results			
Evaluata	Format	Test Environment	System under Test	
		Android 8 Smartphone		
& Configure	Boost Test	Android 9 Smartphone	Finance Tracker 2.0.1	
	Check	Android Smartwatch	Finance Tracker 2.0.0	
	CppUnit	Android Tablet	Finance Tracker 1.1.1	
	ctest	Android 10 Smartphone	Finance Tracker 1.1.0	
	CUnit		Finance Tracker 1.0.0	
	EmbUnit			
	Fitnesse			
	Free Pascal Unit			
	GoogleTest			
	GLib/gtester			
	JBehave			
	JMeter			
	JsUnit			
	Jubula/GUIDancer			
	JUnit			
	MbUnit			
	MSTest			

Figure 8.18. The "Import Test Results" Page

Clicking the Next button opens the Assign Test Results page. The test case of each test result can be selected here using the corresponding drop-down box. Each result must be assigned to a different test case. In addition, the test suite can be assigned as well.

Initial test suite / test case name matching is determined by the internal name of the test case or test suite in Klaros-Testmanagement vs. the external name of the test case or test suite in the result file. This heuristic can be overridden here. Once associated, the matching of a test case or test suite to an external name is automatically applied for further imports until defined otherwise. This also applies to result imports via REST API or other external sources like the Jenkins plugin.

You may decide to skip the import of individual results by deselecting the entry for this result using the checkbox on the left-hand side. If the import process is triggered by a test execution job, the initial selection is matched against the test cases or test suites contained in the job. Otherwise, all results are initially selected for import automatically.

If the option Automatically create new test cases for unknown tests when importing test case results is selected, a new test case is automatically created upon import when it is not already present.

When the option *Create additional test suite results and a corresponding test suite if necessary when importing test case results* is selected, test suite results are automatically created for the corresponding test suite information contained in the result file. This information may vary with the import format. Additionally, a corresponding test suite is created for the test suite result if it did not yet exist.

≡ 👱 K L A R	o s test management	Finance Tracker 🖬 📕 🔍 🥥 🖌 💄	~
📝 Define	Import Test Results		
🚢 Plan	Automatically create new test cases for unknown te Create additional test suite results and a corresponding test suite if necess	sts when importing test case results 🗹 ary when importing test case results 🗹	
📅 Execute	External Test Suite Name 🗢	Associated Test Suite 🗢	
	de.verit.klaros.arquillian.test.LoginTest	TS00008 - Check the connection to the server	~
🕒 Evaluate	External Test Case Name 🖨	Associated Test Case 🗢	
	 de.verit.klaros.arquillian.test.LoginTest.loginAdmin 		~
🖋 Configure	de.verit.klaros.arquillian.test.LoginTest.loginManager	TC00005 - Check account balance.	~
		Import Cancel Back	

Figure 8.19. The "Import Test Results" Page

Pressing the Import button will import all results into Klaros-Testmanagement.

8.5.1. Supported Frameworks

The following frameworks are supported:

AUnit	AUnit is an adaptation of the Java JUnit and C++ CppUnit unit test frameworks for Ada code.
Boost Test	The Boost Test library provides a matched set of components for writing test programs, organizing tests into simple test cases and test suites, and controlling their runtime execution.
Check	Check is a unit testing framework for C.
CppTest	CppTest is a portable and powerful, yet simple, unit testing framework for handling automated tests in C++. The focus lies on usability and extendability.

CppUnit	CppUnit is a C++ unit testing framework.
ctest	ctest is the testing driver provided by CMake.
CUnit	CUnit is a lightweight system for writing, administering, and running unit tests in C.
embUnit	Embedded Unit is unit testing framework for Embedded C Systems.
Fitnesse	Fitnesse is a tool for specifying and verifying application acceptance criteria (requirements).
Free Pascal Unit	Free Pascal Unit is a port to Free Pascal of the JUnit core framework.
GLib/gtester	GLib provides the core application building blocks for libraries and applications written in C. It provides the core object sys- tem used in GNOME, the main loop implementation, and a large set of utility functions for strings and common data structures. gtester is a utility to run unit tests that have been written using the GLib test framework.
Gauge	Gauge is a light-weight cross-platform test automation tool with the ability to author test cases in a business language.
GoogleTest	GoogleTest is a unit testing library for the C++ programming language created by Google.
JBehave	JBehave is a framework for Behaviour-Driven Development (BDD). BDD is an evolution of test-driven development (TDD) and acceptance-test driven design, and is intended to make these practices more accessible and intuitive to newcomers and experts alike. It shifts the vocabulary from being test-based to behaviour-based, and positions itself as a design philosophy.
JMeter	JMeter is a 100% pure Java application designed to load test functional behavior and measure performance. It was originally designed for testing Web Applications but has since expanded to other test functions.
JsUnit	JsUnit is a Unit Testing framework for client-side (in-browser) JavaScript. It is essentially a port of JUnit to JavaScript. Also included is a platform for automating the execution of tests on multiple browsers and multiple machines running different OSs.
Jubula	Jubula provides automated functional GUI testing for various types of applications.
JUnit	JUnit is a simple framework to write repeatable tests. It is an instance of the xUnit architecture for unit testing frameworks.

MbUnit	MbUnit is an extensible unit testing framework for the .NET Framework that takes in and goes beyond xUnit pattern test- ing. MbUnit is part of the Gallio bundle.
MSTest	MSTest is the command-line command that is used to run tests. This command has several options you can use to customize your test run.
NUnit	NUnit is a unit-testing framework for all .Net languages.It is written entirely in C# and has been completely redesigned to take advantage of many .NET language features, for example custom attributes and other reflection related capabilities.
PHPUnit	PHPUnit is a programmer-oriented testing framework for PHP. It is an instance of the xUnit architecture for unit testing frame- works.
QF-Test	QF-Test is a professional tool for automated testing of Java and Web applications with a graphical user interface.
QTestLib	The QTestLib framework is a tool for unit testing Qt based applications and libraries.
Ranorex	Ranorex is a GUI test automation framework for testing of desktop, web-based and mobile applications.
Selenium	Selenium is a web browser automation tool primarily used for automated testing of web apps. Selenium is able to pro- duce JUnit -compatible test results, which can be imported in- to Klaros-Testmanagement.
Squish	Squish is a GUI Test Automation Tool for all kinds of cross- platform desktop, mobile, embedded and web applications. Squish is able to produce JUnit -compatible test results, which can be imported into Klaros-Testmanagement.
TestComplete	TestComplete is a cross-platform automated GUI testing tool from SmartBear Software.
TESSY	TESSY is a software tool to automate module-unit testing of embedded software written in various embedded dialects of the programming languages C and C++.
TestNG	TestNG is a testing framework inspired from JUnit and NUnit but introducing some new functionalities that make it more powerful and easier to use. TestNG is able to produce JUnit - compatible test results, which can be imported into Klaros- Testmanagement.
TUSAR	TUSAR stands for Thales Unified Software Analysis Report. It is a generic metric format composed of 4 categories: Coverage, Measure, Test and Violations.

Unified Functional Testing (UFT) / QuickTest Professional (QTP)	Unified Functional Testing is a functional automated testing software by Micro Focus formerly known as QuickTest Professional (QTP) by HPE Software.
UnitTest++	UnitTest++ is a lightweight unit testing framework for C++. It was for test-driven development on a wide variety of plat- forms. Simplicity, portability, speed, and a small footprint are all very important aspects.
Valgrind	Valgrind is an instrumentation framework for building dynamic analysis tools.
xUnit.net	xUnit.net is a free, open source, community-focused unit testing tool for the .NET Framework. xUnit.net works with ReSharper, CodeRush, TestDriven.NET and Xamarin.



Support for Frameworks not Listed Here

Even if you did not find your favorite testing framework here this does not mean that it is not already supported.

Many tools offer to export their result files in the JUnit XML format (e.g. Selenium, Robot and others) which is a supported format.

If you like to see an additional format supported, please contact support@verit.de.

Chapter 9. Evaluate

This chapter shows how data and results can be evaluated, that were gathered during testing. It explains how the dashboard works and how it can be configured, how reports are configured and rendered and how issues can be created in a connected issue management system.

9.1. The Dashboard

A dashboard is an overview page for configurable reports displayed as a chart. Dashboards can be individually configured and shared with or blocked for other users. Each user can configure any number of dashboards. An example of a dashboard is shown in <u>Figure 9.1</u>.

New adds a new report to the dashboard.

Save saves all changes to the dashboard and its reports.

Cancel discards all changes to the dashboard and its reports.



Figure 9.1. The "Dashboard" Page

Users can switch between dashboard at any time. The drop-down list on the upper right page shows all available dashboards (private and shared by other users). Next to this drop down list are several icons:

- 密 When selected, all users can use this dashboard.
- △ When selected, this dashboard can only be edited by the owner.
- $rac{1}{3}$ If selected, marks this as the default dashboard.

Creates a private copy of this dashboard.

Deletes this dashboard and all of its reports.

The number of reports that should be shown in a row can be set with the numbered buttons (1 to 4) on the upper right. The currently selected number of reports per row is marked in orange. Reports can be moved to another position by dragging and dropping.

All dashboard reports have the same structure: they consist of a title bar and the area where the chart is displayed. The title bar contains the name of the report and several icons:

- ☑ Opens up a dialog to edit the report configuration.
- 凸 Downloads the report as a PNG file.
- Expands/collapses the report.
- **x** Deletes the report.

9.1.1. Default Dashboard

Upon the first start of Klaros-Testmanagement, a default dashboard with three predefined reports is created. This dashboard is accessible by all users an is automatically displayed to each new user. Administrators can change the default dashboard by setting any other dashboard as the default dashboard.



Note

There is always only one default dashboard. If a new default dashboard is set, the previous default dashboard loses this status.

9.1.2. Editing a Dashboard Report

Clicking the 😰 icon opens up a dialog that shows all configurable parameters of the report as well as the report type (which cannot be changed). Changes can be applied to the report by clicking the OK button or discarded by clicking the Cancel button.



Figure 9.2. The "Configure Report" Dialog

9.1.3. Report Types

All reports initially show data from the currently selected project.

In addition to the data of the active project, data of specific projects can also be displayed in the Klaros-Testmanagement Enterprise Edition.

Klaros-Testmanagement Community Edition contains three report types:

- Project Overview
- Latest Success Rate
- Test Activity

enterprise edition Only available in Klaros-Testmanagement Enterprise Edition

The Klaros-Testmanagement Enterprise Edition also contains four additional report types:

- System under Test Overview
- Test Environment Overview
- Test Progress History
- Test Progress

These reports are described in detail in the following sections.

9.1.3.1. The "Project Overview" Report



The Project Overview report shows several metrics of a single project in a chart.



This report displays the following metrics:

- The number of executed/not executed test cases.
- The number of executed/not executed test suites.
- The number of planned/executed jobs.

The following properties of the "Project Overview" report diagram are configurable:

- The name of the report diagram.
- The selection of whether to use the active project or another one in this report. (Klaros-Testmanagement Enterprise Edition only).
- The selection of whether to use the active iteration or another one in this report.
- If the active iteration is not used, a dropdown list with the available iterations is displayed here.

9.1.3.2. The "Latest Success Rate" Report

The *Last Success Rate* report shows the latest results of executed test cases for the selected combination of system under test (one or more) and test environment (one or more).

Evaluate



Figure 9.4. The "Latest Success Rate" Report

The following properties of the "Latest Success Rate" report diagram are configurable:

- One or more systems under test.
- One or more test environments.
- The name of the report diagram.
- The selection of whether to use the active project or another one in this report. (Klaros-Testmanagement Enterprise Edition only).
- If the active project is not used, the following objects to be viewed can be selected:
- The selection of whether to use the active iteration or another one in this report.
- If the active iteration is not used, a dropdown list with the available iterations is displayed here.

9.1.3.3. The "Test Activity" Report

The *Test Activity Report* shows the test case results for a selected combination of system under test and test environment in a selected period of time as a histogram.





The following properties of the "Test Activity" report diagram are configurable:

- Project
- Iteration (if available)
- One or more systems under test.
- One or more test environments.
- The name of the report diagram.
- The selection of whether to use the active project or another one in this report. (Klaros-Testmanagement Enterprise Edition only).
- · If the active project is not used, the following objects to be viewed can be selected:
- The selection of whether to use the active iteration or another one in this report.
- If the active iteration is not used, a dropdown list with the available iterations is displayed here.
- The time period to be covered in days.
- 9.1.3.4. The "System under Test Overview" Report

Only available in Klaros-Testmanagement Enterprise Edition

The *System under Test Overview Report* shows the success and progress rate of test environments under a single system under test in a radar chart. If fewer than three test environments are configured, a bar chart will be displayed instead.



Figure 9.6. The "System under Test Overview" Report

The following properties of the "System under Test Overview" report diagram are configurable:

• The Project.

- Iteration (if available).
- A system under test.
- One or more test environments.
- The name of the report diagram.
- The selection of whether to use the active project or another one in this report. (Klaros-Testmanagement Enterprise Edition only).
- · If the active project is not used, the following objects to be viewed can be selected:
- The selection of whether to use the active iteration or another one in this report.
- If the active iteration is not used, a dropdown list with the available iterations is displayed here.

9.1.3.5. The "Test Environment Overview" Report



The *Test Environment Overview Report* shows the success and progress rate of systems under test under a single test environment in a radar chart. If fewer than three systems under test are configured, a bar chart will be displayed instead.



Figure 9.7. The "Test Environment Overview" Report

The following properties of the "Test Environment Overview" report diagram are configurable:

- Project
- Iteration (if available)
- One or more systems under test.
- A test environment.
- The name of the report diagram.

- The selection of whether to use the active project or another one in this report. (Klaros-Testmanagement Enterprise Edition only).
- If the active project is not used, the following objects to be viewed can be selected:
- The selection of whether to use the active iteration or another one in this report.
- If the active iteration is not used, a dropdown list with the available iterations is displayed here.

9.1.3.6. The "Test Progress" Report



The *Test Progress* report shows the rate of executed versus defined test cases of a project for a given set of test environments and test systems.



Test progress (SUT FinanceTracker 2.1.0 and test environment Android Smartphone 5.0)

Figure 9.8. The "Test Progress" Report

The following properties of the "Test Progress" report diagram are configurable:

- Project
- · Iteration (if available).
- One or more systems under test.
- One or more test environments.
- The name of the report diagram.
- The selection of whether to use the active project or another one in this report. (Klaros-Testmanagement Enterprise Edition only).
- If the active project is not used, the following objects to be viewed can be selected:
- The selection of whether to use the active iteration or another one in this report.

- If the active iteration is not used, a dropdown list with the available iterations is displayed here.
- A boolean value indicating that only test cases should be considered that are part of a job.

9.1.3.7. The "Test Progress History" Report

Only available in Klaros-Testmanagement Enterprise Edition

The *Test History* report shows the rate of defined test cases versus executed test cases for a project for one or more test environments and one or more systems under test in a given time period.



Figure 9.9. The "Test Progress History" Report

The following properties of the "Test Progress History" report diagram are configurable:

- Project
- · Iteration (if available).
- One or more systems under test.
- One or more test environments.
- The name of the report diagram.
- The selection of whether to use the active project or another one in this report. (Klaros-Testmanagement Enterprise Edition only).
- If the active project is not used, the following objects to be viewed can be selected:
- · The selection of whether to use the active iteration or another one in this report.
- If the active iteration is not used, a dropdown list with the available iterations is displayed here.
- The time period in days

9.2. Reports

On the *Reports* page, predefined reports can be generated as PDF or HTML files. The Klaros-Testmanagement Enterprise Edition also allows configurable reports to be created, customized and retrieved as PDF or Excel files.

9.2.1. Predefined Reports

= 坐 K L A R (OS TEST MANAG	GEMENT	Finance Tracker 🧧 📑	२ १ ~ ≛ ~
📝 Define	Reports			
🏝 Plan	Name Test Environment Report An ov System under Test Report An ov	Description erview report of the test environments erview report of the systems under test		Action
🏟 Execute	Test Suite Report An ov Test Run History Report Test r	erview report of the test suites un history for selected system under test and period		2 S 2 S
쓵 Evaluate	Configurable Reports	7 Entries - Page 1 of 1 🔣 🚽 1 🕨 🕅 10 🗸		7 & ≡
∲ Configure	Name ♥ Issue Overview Report Iteration Status Report Job Status Report Job Status Report Job Status Report System under Test Status Report Test Run Overview Report	Lescription \$ Lists the issues in a project sorted by selectable parameters Summary of an iteration status in a project Lists the jobs in a project grouped by selectable parameters Lists the jobs in a project sorted by selectable parameters Lists the test case results in a SUT sorted by selectable parameters Lists the test runs in a project sorted by selectable parameters	Revision Changed By ◆ Changed Changed By ◆ Changed	Ingel Formal \$ Action ,11:47 AM PDF D ,11:47 AM PDF D

Figure 9.10. The "Reports" Page

The predefined reports are:

- The "Test Environment Report"
- The "System under Test Report"
- The "Test Suite Report"
- The "Test Run History Report"

The predefined reports can be generated in two different file formats:

- 🖾 PDF
- 🗟 HTML

Clicking on one of the file type icon will generate the report in the selected format.

The upper part of the *Reports* page shows a table with the available predefined reports:

9.2.1.1. Test Environment Report

The *Test Environment Report* shows all test runs of a single test environment, including the execution date, executor and the quantity of *Passed*, *Failure*, *Error*, *Inconclusive* and *Skipped* results. An example of the report is shown in Figure 9.11.



Test Environment Overview



Project: Test Environments:	Finance Tracke	er (P00002)			Author: Date:	Felix Mu Dec 8, 20	stermann)21, 4:39 Pl	м
							-	
ld:	ENV00004							
Description:	Android 9 Sma	rtphone						
Test Run		Execution Date	Executed by	Passed	Inconclu	sive Failed	Error	Skipped
P00002 TRU0000039: TC0000)1 - Transfer	7/2/20, 2:34:23 PM	Till Tegen	1	0	0	0	0
money to a local bank accoun P00002 TRU0000067: TC0000	t.)5 - Check	7/7/20, 9:33:32 AM	Tim Thiel	1	0	0	0	0
account balance. P00002 TRU0000066: TC0000	03 - Transfer	7/7/20, 9:33:32 AM	Tim Thiel	1	0	0	0	0
money to a different bank acc P00002 TRU0000052: TC0001	ount. 10 - Transfer	7/7/20, 9:33:32 AM	Thomas Tafel	1	0	0	0	0
money without specifying a re P00002 TRU0000063: TC0001	cipient. 13 - Transfer	7/7/20, 9:33:32 AM	Tim Thiel	1	0	0	0	0
money without specifying a re P00002 TRU0000053: TC0000	ference.)4 - Transfer	7/7/20, 9:33:32 AM	Thomas Tafel	1	0	0	0	0
money to a foreign bank. P00002 TRU0000097: TC0001	12 - Transfer	7/9/20, 6:53:55 AM	Tanja Toppler	1	0	0	0	0
money without specifying an a P00002 TRU0000108: TC0000	amount.)3 - Transfer	7/9/20, 6:53:55 AM	Tanja Toppler	0	0	0	1	0
money to a different bank acc P00002 TRU0000098: TS0000	ount.)6 - Overview	7/9/20, 6:53:55 AM	Tanja Toppler	6	0	0	0	1
P00002 TRU0000094: TC0001	13 - Transfer	7/9/20, 6:53:55 AM	Tanja Toppler	1	0	0	0	0
money without specifying a re P00002 TRU0000095: TC0001	ference. 10 - Transfer	7/9/20, 6:53:55 AM	Tanja Toppler	1	0	0	0	0
money without specifying a re P00002 TRU0000135: TC0001	cipient. 10 - Transfer	7/10/20, 3:45:38 PM	Timo Tunklik	1	0	0	0	0
money without specifying a re P00002-TRU0001074: TC0002	cipient. 23 - Change a	7/16/20, 1:43:03 PM	Felix Mustermann	0	0	0	1	0
standing order. P00002-TRU0001673: TC0002	22 - Update	7/29/20, 7:28:37 AM	Felix Mustermann	0	0	1	0	0
dashboard by adding a standi P00002-TRU0001676: TC0002	ng order. 21 - Update	7/29/20, 7:33:07 AM	Felix Mustermann	1	0	0	0	0
dashboard by cancelling a sta P00002-TRU0001679: TC0002	nding order. 20 - Update	7/29/20, 7:34:20 AM	Felix Mustermann	0	0	0	1	0
dashboard by cancelling a del P00002-TRU0001685: TS0000	oit mandate.)7 - Edit standing	7/29/20, 7:41:14 AM	Felix Mustermann	3	0	0	0	0
orders P00002-TRU0001688: TS0000	06 - Overview	7/29/20, 7:50:24 AM	Felix Mustermann	7	0	0	0	0

Figure 9.11. The "Test Environment" Report

9.2.1.2. System under Test Report

The *System under Test Report* shows all test runs of a single system under test, including the execution date, executor and the quantity of *Passed*, *Failure*, *Error*, *Inconclusive* and *Skipped* results. An example of the report is shown in Figure 9.12.



System Under Test Report



Project:		Finance Tra	icker (P00002)			Author:	Felix Mus	termann	
Systems Under T	est:	6				Date:	Dec 8, 202	21, 4:42 Pl	М
ld:	SUT000	001							
Version:	Finance	e Tracker 1.0.	0						
Test Run			Execution Date	Executed by	Passed	inconclu	isive Failed	Error	Skipped
P00002 TRU0000016:	TC00005 -	Check account	7/1/20, 7:03:00 AM	Till Tegen	0	0	1	0	0
balance. P00002 TRU0000011:	TC00013 -	Transfer money	7/1/20, 7:03:00 AM	Timo Tunklik	0	1	0	0	0
P00002 TRU0000007:	TC00010 -	Transfer money	7/1/20, 7:03:00 AM	Timo Tunklik	0	1	0	0	0
without specifying a re P00002 TRU0000001:	cipient. TC00013 -	Transfer money	7/1/20, 7:03:00 AM	Till Tegen	0	1	0	0	0
without specifying a re P00002 TRU0000008:	ference. TC00013 -	Transfer money	7/1/20, 7:03:00 AM	Timo Tunklik	0	0	1	0	0
without specifying a re P00002 TRU0000004:	ference. TC00013 -	Transfer money	7/1/20, 7:03:00 AM	Till Tegen	0	1	0	0	0
without specifying a re P00002 TRU0000012:	ference. TS00008 -	Check the	7/1/20, 7:03:00 AM	Timo Tunklik	0	0	1	0	0
P00002 TRU0000014:	er TS00008 -	Check the	7/1/20, 7:03:00 AM	Sabrina Gidley	0	1	0	0	0
P00002 TRU0000006:	er TC00010 - '	Transfer money	7/1/20, 7:03:00 AM	Timo Tunklik	0	1	0	0	0
without specifying a re P00002 TRU0000013:	cipient. TS00008 - (Check the	7/1/20, 7:03:00 AM	Sabrina Gidley	1	0	0	0	0
connection to the serv P00002 TRU0000002:	er TC00012 -	Transfer money	7/1/20, 7:03:00 AM	Till Tegen	0	0	1	0	0
without specifying an a P00002 TRU0000010:	amount. TC00012 - `	Transfer money	7/1/20, 7:03:00 AM	Tim Thiel	0	1	0	0	0
without specifying an a P00002 TRU0000003:	amount. TC00010 - T	Transfer money	7/1/20, 7:03:00 AM	Timo Tunklik	0	0	1	0	0
without specifying a re P00002 TRU0000015:	cipient. TS00002 - I	Bank Transfer	7/1/20, 7:03:00 AM	Tim Thiel	0	7	1	0	0

Figure 9.12. The "System under Test" Report

9.2.1.3. Test Suite Report

The *Test Suite Report* shows all test cases including team, creator, priority and name, for each test suite in the active project. An example report is shown in <u>Figure 9.13</u>.

	ligement	Test	Suite Ove	erview
Project:	Finance Tracker (P00002)		Author: Felix Mustermann
Test Suites:	7			Date: Jun 15, 2021, 11:26 PM
ld:	TS00006	Test Cases:	7	Test Runs: 19
Name:	Overview			
Test Case	Team	Created by	<u>Priority</u>	Name
TC00015	Team Alpha	Talal Arif	Medium	Add a standing order.
TC00016	Team Alpha	Talal Arif	Medium	Delete standing orders.
TC00017	Team Alpha	Talal Arif	Medium	Delete debit mandate.
TC00006	Team Alpha	Talal Arif	Medium	Display the activities of last week.
TC00007	Team Alpha	Talal Arif	Medium	Display the activities of last month.
TC00008	Team Alpha	Talal Arif	High	Display activities within user-specified time period
TC00009	Team Alpha	Talal Arif	Medium	Display activities of the last three months.
ld:	TS00002	Test Cases:	8	Test Runs: 7
Name:	Bank Transfer			



9.2.1.4. Test Run History Report

The *Test Run History Report* shows all test runs of a single system under test, including the execution date, executor and the quantity of *Passed*, *Failure*, *Error*, *Inconclusive* and *Skipped* results in a given time period.



Figure 9.14. The "Test Run History" Report

9.2.1.5. Configurable Reports

edition Only available in Klaros-Testmanagement Enterprise Edition

In addition to predefined reports, the Klaros-Testmanagement Enterprise Edition can create PDFs and Excel files from user-defined reports. Creating user-defined reports is explained in <u>Section 10.2, "Report Templates"</u>. Clicking on the \square and \square icons will generate the reports.



Note

Clicking on one of the icons [A] or [B] opens a dialog where report parameters can be set if any are needed for generating.

≡ 坐 К L A R (o s test manac	SEMENT	Finance Track	er 🖬 📑	٩	× @~ ≛~
📝 Define	Reports					
📇 Plan	Name Test Environment Report Repo	t on the test environments and their test results	Description			Action
🔅 Execute	System under Test Report Repo Test Suite Report Repo Test Run History Report Test	t on the systems under test and their test results t on the test suites and their test results un history for selected system under test and period				6 2 6 2 6 2
伕 Evaluate	Configurable Reports					
🔑 Configure	Name 🗢	7 Entries - Page 1 of 1	1 N 15 V	Revision	Changed By 🗢 Changed 🗢	Format
	Issue Overview Report	Lists the issues in a project sorted by selectable parameters		1.1	System Account 4/6/22, 5:17 AM	PDF 🕅
	Iteration Status Report	Summary of an iteration status in a project		1.1	System Account 4/1/22, 5:21 AM	PDF
	Job Excel Overview Report	Lists the jobs in a project		1.1	System Account 4/1/22, 5:21 AM	Excel 🕅
	Job Progress Report	Lists the jobs in a project sorted by selectable parameters		1.1	System Account 6/24/22, 2:20 AM	PDF 🖪
	System under Test Status Report	Lists the test case results in a SuT sorted by selectable paran	neters	1.1	System Account 6/24/22, 2:20 AM	PDF
	Test Run Overview Report	Lists the test runs in a project sorted by selectable parameter	/s	1.1	System Account 6/24/22, 2:20 AM	PDF C
	Job Status Report	Lists the jobs in a project grouped by selectable parameters		1.1	System Account 6/24/22, 2:20 AM	PDF 🚨

Figure 9.15. Generate a Parameterized Report



Note

The OK button becomes active after entering all required parameters.

9.3. Test Runs

This page lists all test runs that have been executed in the selected project.



Note

Only fully executed test cases are displayed on this page. Suspended test runs can be continued at <u>Section 8.4, "Continue Test Run"</u>.

										5	ave	Disce
12 A B			220 Entries - Page 1 of 12 N	4 1 2	3 4 5 🕨	M 30 V			V I	lis work	0	Q X Z
00	Start 🕈	Assignee @	Iteration @	Job 0	Test Case/Suit	e Test Environment	System under Test 🕈	Results 0	0 A	0 24	0.	
TRU0012351	2 years ago	Felts Mustermann			TC02023	Android 8 Smartphone	Finance Tracker 2.1.0	1			P	RA.
TRU0012350	2 years ago	Felx Mustermann			TC00028	Android 8 Smartphone	Finance Tracker 2.1.0	1			PA	RA
TRU0011693	2 years ago	Felx Mustermann			TC0006	Android 8 Smartphone	Finance Tracker 2.1.0	1			PA	RA
TRU0011692	2 years ago	Felx Mustermann		30800355	TC00015	Android 8 Smartphone	Finance Tracker 2.0.0	1			PA	RA.
TRU0011691	2 years ago	Felix Mustermann			TC00006	Android 8 Smartphone	Finance Tracker 2.1.0	1			0	BA.
TRU0011650	2 years ago	Felx Musternano			TC00020	Android 8 Smartphone	Finance Tracker 2.1.0	1			0	BA.
TRU0001735	4 years ago	Felts Mustermann			T\$00027	Android 8 Smartphone	Finance Tracker 2.1.0	2	21	100	. A	RA.
TRU0001774	A years and	Daily Mustamany	7.0 v. Seciet 01. Internation dashboard		7500017	Andreid & Smortebone	Einersee Tracker 2.1.0				B	DA.
T81/0001733	4 years ann	Felx Musternano			T\$00027	Andread & Smortphone	Finance Tranker 2.1.0	8			- 0	ñĂ
T81/0001732	4 years ann	Fely Musternann	2.1 x-Societ 01-Octimizing for use on smartwatches		TC00014	Andread 9 Smortphone	Finance Tracker 2.1.0	1			10	ñA.
TRU0001731	4 years ago	Felx Musternano	2.1 x-Sprint 01-Optimizing for use on smartwatches		TC00065	Android 10 Smartphone	Finance Tracker 2.1.0	1			0	BA
TRU0001730	A second ago	Eally Mustermann	2.1 x Sprint 01. Optimizing for use on emericant/has		TC000522	Android Tables	Einanne Tracker 2.1.0		_		10	BA.
TRU0001759	A second ago	Fely Murtemann	2.1 x Societ 01. Optimizing for use on constantibles		7500022	Android Smathaatch	Einence Tracker 2.1.0				0	6A
TRU0001728	A years ago	Felt Musternary	2.1 x Sector 01-Optimizing for use on emethanishes		T500016	Android & Smartchone	Einerge Tracker 2.1.0	7			6	6A
100001728	A years ago	Fels Mustermann	2.1. Control Of Control and Apply an		100000	Andreid Tables	Finance Tracker 2.1.0				0	200
TRUCOUT727	A years ago	Fels Mustermann	2.6 x Postal IVI Extended dashboard		1000000	Andread Tablet	Finance Tracker 2.0.0				0	88
100001720	A years ago	Fels Mustermann	2.5 x Postal 72 Extended dashboard		7500000	Andread D Presetchone	Finance Tracker 2.0.0				0	BA
1900001725	+ years ago	Felt Musternann	2.0.0 oprint 02.00 and op dashboard		1500001	Android V Smartprone	Finance Tracker 2.0.0				13	80
1900001724	+ years ago	Felc Musternann	2.00 spini 02 Extending dashedard		7000000	Android To Smarphone	Finance Tracker 200		0			89
1900001725	+ years ago	Felt Musternann	2.00 spini 02 Extensing dashedard		1000002	Andreid to Omentalise	Finance Indoler 2.0.0		0			89
100001721	4 years ago	Pett Mostermann	2.0.0 opinit of integrating easiloand		1500007	Andread To Smartprone	Phance hadver 2.0.0					20
1800001720	a years ago	Peak Neuerennann	2. Constanti on maganing associatio		1300077	Printing Taken	Pliance harder 2010				0	20
1800001719	4 years ago	Feex Multernam	2.0.x-spinit 01-integrating dashboard		1800007	Andread 9 smartprione	Finance Itacker 2.0.0				8	200
1100001718	+ years ago	Feet Musternann	1.1.x-sprint 04-optimizing for tablets		1000022	Andread TO Similar process	Pinance Insper 1.1.1				13	
TRU0001717	4 years ago	Felx Mustermann	1.1.x-Sprint D4-Optimizing for tablets		TC00021	Android Tablet	Finance Tracker 1.1.1	1	_		La la	80
1900001716	4 years ago	Felx Mustermann	1.1.x-spnilt 04 optimizing for tablets		1000004	Andreed Tablet	Finance Tracker 1.1.1	1			E	80
1900001715	4 years ago	Felx Mustermann	1.1.x-sprint 04-optimizing for tablets		1000002	Andreed Tablet	Pinence Tracker 1.1.1	1	_		6	80
TRJ0001714	4 years ago	Felx Mustermann	1.1.x-Sprint 04-Optimizing for tablets		TC00012	Android Tablet	Finance Tracker 1.1.1	1	1		6	80
TRU0001713	4 years ago	Felx Mustermann	1.1.x-Sprint 04-Optimizing for tablets		TC00068	Android 10 Smartphone	Finance Tracker 1.1.1	1	1		6	80
TRU0001712	4 years ago	Felx Mustermann	1.1.x-Sprint 04-Optimizing for tablets		TC0006	Android Tablet	Finance Tracker 1.1.1	1	1		6	90
	4 years ago	Felts Mustermann	1.1.x-Sprint D4-Optimizing for tablets		T500003	Android Tablet	Finance Tracker 1.1.1	4	2		6	88
	Cf C C TRACOTION TRACOTION TRACOTION TRACOTION TRACOTION	Image: Constraint of the second se	Image: Constraint of the second se	Image: Solution of the second seco	Image: Control (Control (Contro) (Contro) (Control (Contro) (Control (Contro) (Control (Contro)	Image: Control of the second secon	Image: Solution: Part (Strict) Image:	Control Distance Distance	Image: Second	Construction Distance instruction Construction Cons	Image: Second	Totol Totol <th< td=""></th<>

Figure 9.16. The "Test Runs" Page

The table shows the following values:

ID	The automatically assigned ID.				
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.				
Start	The start time of the test run.				
Assignee	The name of the user to whom the test run is assigned.				
Iteration	The iteration in which the test run was executed.				
Job	The task with which the test run was executed.				
Test Case/Suite	The ID of the test case or test suite of the test run.				
Test Environment	The test environment in which the test was executed.				
System Under Test	The system under test that was tested.				
Results	The number of test case results.				
Passed	The number of test case results with the status/verdict "Passed".				
Failed	The number of test case results with the status/verdict "Failed".				
Error	The number of test case results with the status/verdict "Error".				
Inconclusive	The number of test case results with the status/verdict "Inconclusive".				
Skipped	The number of test case results with the status/verdict "Skipped".				
Action	The executable actions.				

9.3.1. Action

The following actions can be performed in the action column:

- Generate a test run report in PDF format.
- Generate a test run report in HTML format.
- Open print view.
- Q Show the test case results of this test run.
- Delete this test run.

If a test run has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted test runs, the following actions are available instead of *Delete*:

Restore this test run.

⊘ Irrecoverably remove the test run from the database (only Administrator).

Deleting a Test Run

When a test run is deleted, the associated test case results and test suite results are also deleted

Once all test runs for a specific system under test or test environment have been deleted, the respective system under test or test environment may also be deleted. Otherwise, they remain locked for deletion.

9.3.2. Bulk Actions

Certain actions can also be applied to several test runs at once. To do this, select the test runs to which the action is to be applied in the leftmost column.

The following bulk actions are supported for test runs:

🖉 Edit

- Open print view.
- 🗊 Delete

For additional information, see Section 5.2.3.1.5, "Bulk Actions".

9.3.3. Table Operations

The following operations can be performed in the line above the table on the right:

Filter / Sort

Show all / Only active (only Administrator)

Q Search

- 凸, Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

9.3.4. The Test Run Report

The "Test Run Report" shows the details of a test run including its execution date, executor, system under test, test environment and the number of *Passed*, *Failure*, *Error*, *Inconclusive* and *Skipped* results. It can be generated in PDF or HTML format. An example PDF is shown in Figure 9.17.

		Test F P00002 - F				
	Creation Date	4			Author Felix Mustermann	
		Reporte	ed Test R	uns		
F00002-TRU0001729	Run ID : TS00002 - Bank Transfer	Date 7/29/20. 9:45:12 AM	Finance Track	SUT er 2.1.0	Android Sma	st Environment
		Test Res	sult Sumr	mary		
Success Rate	Passed	Failed	Error	Inconclusive	Skipped	Total
50%	4	3	1	0	0	8
		Passed Failed Figure 9.17. Th Test Re	Error Incon	n" Report tails	ed	
		Test Case				Result
T000010 Transfer	Test Require	ement(s)			Duration [sec]	Test Run
At l	east 100 users should be a	able to use the applicat	ion simultaneou	Isly without any	8.02	P00002-TRU000172
		Test Cas	e Sten			Result
Start th	e application by pressing t	the blue icon button.				Passed
Select t	the button that is found in	the main menu.				Failed
Enter th	ne following in the fields: B	eneficiary of payment:	Max Mustermai	nn IBAN: DE19 123	4 1234 1234 1234	Skipped
Select '	'Yes".					Skipped
Enter T	AN and press the button					Skipped



9.3.5. Details Page

Each test run has its own detail page. By clicking on the ID of the respective test run or on the Q in the action column, you can access the detail page. All test case results and, if available, the test suite result of the test run are displayed in tables here.

The following tabs are available: *Properties, User Defined* and *Results*. At first invocation this is the *Results* tab with the sub tabs *Test Case Results* and *Test Suite Results*.

9.3.5.1. Actions

The action toolbar is located in the top right corner of the title bar The following actions can be performed on the detail pages:

🔓 Open execute view	Re-execute this test run.
👌 Open print view	A print-ready view of the iteration can be created here. With a click on the icon 🔒 this opens in a new browser tab.
	Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .
☐ Create Bookmarks	Each individual detail page can also be reached directly via a hyperlink. By clicking on the icon 🔲 this link is copied to the clipboard.
	The creation of bookmarks is described in detail in <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".
G Browse	Use the green arrows at the very top right to switch between the properties present on the previous page.

9.3.5.2. Properties

The Properties tab shows the following information:

ID	The automatically assigned ID.
Executor	The user who executed this test run.
System under Test	A drop-down menu for selecting the assigned system under test.
Test Environment	A drop-down menu for selecting the assigned test environ- ment.
Iteration	The iteration in which the test run was executed.
Assignee	The user to whom this test run is assigned.

* A test manager or administrator can retroactively correct the entries here. This may be necessary, for example, in the case of an incorrect entry when starting the test run.

9.3.5.3. User Defined

You can create your own fields to meet individual requirements. For further information please refer to <u>Section 5.2.3.2.4</u>, "User Defined Properties".

9.3.5.4. Results

The *Results* tab splits into the *Test Case Results* subviews. and *Test Suite Results*. The table displays the following values:

ID	The automatically assigned ID.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.

Evaluate

Start	The start time of the test run.
Iteration	The iteration in which the test run was executed.
Test Case	The test case.
Test Environment	The test environment in which the test was executed.
System Under Test	The system under test that was tested.
Executor	The name of the user who last executed the test run.
Duration	The execution time.
Result	The test case result (Passed, Failed, Error, Inconclusive, Skipped).

Action

The executable actions.



Figure 9.19. The "Test Run Details" Page

9.3.6. Changes

The tab *Changes* shows the change history of this test run.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

9.4. Test Case Results

The Test Case Results page shows an overview of all test cases, including the quantity of Passed, Failure, Error, Inconclusive and Skipped results.

≡ 📌 K L A R	o s test management	Finance Tracker 👕 📑	२ × 0 ४ ≗ ४
Define	Test Case Results		
🙁 Plan	+ 24 Entries - Page 1 of 3 🖌 ┥ 🚺 2	3 ▶ ▶ 10 ▼ 品√	
_	□ ID ♦ Revision ♦ Name ♦	Traceability 🗢 🛛 Issues 🗢 Resu	lts ≑ 📀 ≑ 🛕 ≑ 😣 ≑ 🔀 ≑ 🤶 ≑ Action
📅 Execute	TC00024 1.0 A Check the connection to the server	Requirement 1.2.4 in testplan_16.txt 2 / 1	3 💼 💼 💼 -5 Q
	TC00023 1.0 Schange a standing order.	Requirement 1.2.4 in testplan_16.txt 0 / 2 3	5 17 10 3 3 2 🚑 📿
	TC00022 1.0 A Update dashboard by adding a standing order.	Requirement 1.2.4 in testplan_16.txt 5 / 0 2	4 10 10 2 2 2
e Evaluate	TC00021 1.0 Ø Update dashboard by cancelling a standing order.	Requirement 1.2.4 in testplan_16.txt 0 / 2 2	4 17 4 3 🎥 🔍
	TC00020 1.0 Opdate dashboard by cancelling a debit mandate.	Requirement 1.2.4 in testplan_16.txt 0 / 0 1	9 🛛 8 🛛 4 📕 5 1 1 🖓 🕰 🔍
🔑 Configure	TC00019 1.0 Connection to the database.	de.verit.financetracker.web.Connection 0 / 0) - 1 Q
	TC00018 1.0 Connection to the server of the bank.	de.verit.financetracker.web.Connection 0 / 0) - 1 Q
	TC00017 1.0 A Delete debit mandate.	Requirement 1.2.4 in testplan_16.txt 0 / 0 3	1 12 7 4 8 🚑 🔍
	TC00016 1.0 Selete standing orders.	Requirement 1.2.4 in testplan_16.txt 0 / 0 4	3 31 4 3 5 🚑 📿
	TC00015 1.0 X Add a standing order.	Requirement 1.2.4 in testplan_16.txt 0 / 0 5	1 34 9 4 3 1 🔩 🔍
	ID Revision Name		ults Action

Figure 9.20. The "Test Case Results" Page

The table shows the following values:

ID	The automatically assigned ID.
Revision	The revision of the test case.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Name	The name of the test case.
Traceability	The requirement assigned to this test case.
Issues	The number of open and resolved issues.
Results	The number of test case results.
Passed	The number of test case results with the status "Passed".
Failed	The number of test case results with the status/verdict "Failed".
Error	The number of test case results with the status/verdict "Error".
Inconclusive	The number of test case results with the status/verdict "Incon- clusive".
Skipped	The number of test case results with the status/verdict "Skipped".
Action	The executable actions.
00000000 Note	



Note

The Result column shows the latest result of the test cases.

9.4.1. Action

The following actions can be performed in the action column:



Import results.

Q Show the test case results of this test case (see Section 9.4.4, "Single Test Case Results").

9.4.2. Bulk Actions

Certain actions can also be applied to several test cases at once. To do this, select the test cases to which the action is to be applied in the leftmost column.

The following bulk actions are supported for test cases:

+ Add test cases to a test suite (see Figure 9.21).

For additional information, see Section 5.2.3.1.5, "Bulk Actions".

= 📌 K L A R (d s test managei	MENT	Financ	e Tracker 🧧 🗮		Q × 0 × ≛×
📝 Define	Test Case Results					
📇 Plan	+ 24 of 24 selected	24 Entries - Page 1 of 3 🛛 🖊 🔍	1 2 3 🕨 🕅 10 🗸	#V	۵ 🕸 🔁	Q × & ≡
	✓ ID	Name 🗢	Traceability 🗢	Issues 🖨 Result	ts 🗘 🥥 🗘 🔔 🎗 🏵	🗢 🔀 🗢 😨 🗢 Action
🏟 Execute	✓ TC00024 1.0	ne connection to the server	Requirement 1.2.4 in testplan_16.txt	2/1 3	1 1	- 5 Q
	✓ TC00023 1.0 ⊘ Change	a standing order.	Requirement 1.2.4 in testplan_16.txt	0/2 35	17 10	3 3 2 🍰 Q
4 5 1 1	✓ TC00022 1.0 ▲ Update	dashboard by adding a standing order.	Requirement 1.2.4 in testplan_16.txt	5/0 24	10 10	2 2 🎝 🕹 🕹 🖓
e Evaluate	✓ TC00021 1.0	dashboard by cancelling a standing order.	Requirement 1.2.4 in testplan_16.txt	0/2 24	17 4	3 2
	✓ TC00020 1.0 S Update	dashboard by cancelling a debit mandate.	Requirement 1.2.4 in testplan_16.txt	0/0 19	8 4	5 1 1 🎝
🌽 Configure	✓ TC00019 1.0 Connect	ion to the database.	de.verit.financetracker.web.Connection	0/0 0		- 5 Q
	✓ TC00018 1.0 Connect	ion to the server of the bank.	de.verit.financetracker.web.Connection	0/0 0		- 5 Q
	✓ TC00017 1.0 🔥 Delete d	ebit mandate.	Requirement 1.2.4 in testplan_16.txt	0/0 31	12 7	4 B 🍰 Q
	✓ TC00016 1.0 Ø Delete s	Assign the test cases to a (new) test-suite		0/0 43	31 4	3 5 🔩 Q
	✓ TC00015 1.0 × Add a st	The following test cases will be ac	Ided to a test suite	0/0 51	34 9	4 3 1 🍰 Q
	ID Revision			Issues Resu		Action
		Create new test suite Add to an existing test suite Name				
			OK Cancel			

Figure 9.21. The "Add test cases to test suite" Dialog

9.4.3. Table Operations

The following operations can be performed in the line above the table on the right:

√ Filter / Sort

Show Executed/All Test Cases

Show Newest Only/Show All

- Q Search
- 옰 Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

9.4.4. Single Test Case Results

The "Test Case Results" page shows all executions of the selected test case, including the result, the used system under test and the test environment (Figure 9.22).

= 📌 K L A R (o s test management	Finance Tracker 🧧 🗮	५ × 0⁄ × ≰×
🕑 Define	Test Case Results: TC00022 - Update dashboard by adding a standing order	ar.	
🏩 Plan	+ Test Case Details		
🏟 Execute	24 Entries - Page 1 of 3 ▲ 1 2 3 ID ◆ Ø Start ◆ Test Run ◆ Iteration ◆	► N 10 - Test Environment ♦ System under Test ♦ Executor ♦	Q × ⅔ ≡ Duration \$ Result \$ Action
쓵 Evaluate	TCR0002460 V 4 years ago TRU0001730 2.1.x-Sprint 01-Optimizing for use on smartwatches	Android Tablet Finance Tracker 2.1.0 Sandra Selen	00:00:08
🖌 Configure	CR0002439 4 years ago RU0001725 2.0.x-Sprint 02-Extending dashboard TCR0002407 4 years ago TRU0001718 1.1.x-Sprint 04-Optimizing for tablets	Android 9 Smartphone Finance Tracker 2.0.0 Sandra Selen Android 10 Finance Tracker 1.1.1 Felix Mustermann Smartphone	
	TCR0002380 4 years ago TRU0001705 1.1.x-Sprint 03-Cancelling debit mandates TCR0002342 4 years ago TRU0001697 1.1.x-Sprint 02-Edit standing orders	Android 9 Smartphone Finance Tracker 1.1.1 Felix Mustermann Android 10 Finance Tracker 2.0.0 Felix Mustermann Smartphone	00:00:08 😢 🍰 📿 00:00:12 🧭 🍰 📿
	□ TCR0002338 ♀ 4 years ago TRU0001696 1.1.x-Sprint 02-Edit standing orders	Android 10 Finance Tracker 2.0.0 Sandra Selen Smartphone	00:00:09 🔀 🎥 🖨 🔍
	TCR0002334 4 years ago TRU0001695 1.1.x-Sprint 02-Edit standing orders TCR0002330 4 years ago TRU0001694 1.1.x-Sprint 02-Edit standing orders	Android 9 Smartphone Finance Tracker 2.0.0 Felix Mustermann Android 8 Smartphone Finance Tracker 2.0.0 Felix Mustermann	00:00:09 ♥ ♣ ⊕ Q 00:00:09 ♥ ♣ ⊕ Q
	TCR0002297 4 years ago TRU0001686 1.1.x-Sprint 01-Overviews TCR0002282 4 years ago TRU0001683 1.0.x-Sprint 02-Extending basic functions	Android 8 Smartphone Finance Tracker 2.0.0 Felix Mustermann Android 8 Smartphone Finance Tracker 2.0.0 Felix Mustermann	00:00:09 🛕 よ 🛱 Q 00:00:10 🛕 🏖 🛱 Q
	ID Start Test Run Iteration	Test Environment System under Test Executor	Duration Result Action
			Back

Figure 9.22. The "Test Case Results" Page

The collapsible panel on top of the page shows the details of the selected test case. Below it is a table with all results.

The table shows the following values:

ID	The automatically assigned ID.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Start	The start time of the test run.
Test Run	The test run, to which the result belongs.
Iteration	The iteration in which the test run was executed.

Test Environment	The test environment in which the test was executed.
System Under Test	The system under test that was tested.
Executor	The name of the user who last executed the test run.
Duration	The execution time.
Result	The test case result (Passed, Failed, Error, Inconclusive or Skipped).
Action	The executable actions.

The **?** icon in the Additional Information column shows whether the corresponding test case result has been manually changed by a user.

Clicking the Back button navigates back to the test case results page (Figure 9.20).

9.4.4.1. Action

The following actions can be performed in the action column:

Re-execute this test case.

Import results.

Q Show the detail page of the test case result (see <u>Section 9.4.4.4, "The "Test Case Result"</u> <u>Details Page "</u>).

9.4.4.2. Bulk Actions

Certain actions can also be applied to several test case results at once. To do this, select the test case results to which the action is to be applied in the leftmost column.

The following bulk actions are supported for test case results:

Open print view

For additional information, see Section 5.2.3.1.5, "Bulk Actions".

9.4.4.3. Table Operations

The following operations can be performed in the line above the table on the right:

√ Filter / Sort

Show Newest Only/Show All

Q Search

& Export

 \equiv Column selection
All operations are described in detail in Section 5.2.3.1, "Overview Page".

9.4.4.4. The "Test Case Result" Details Page

This page shows a single test case result and its test case step results.

= 👱 K L A R	OS TEST MANAGEMENT	Finance Tracker 🗧 🚆	Q × Ø~ ≛~
📝 Define	🛕 🖗 Test Case Result: TCR0002460 (TC00022 - Update dashb	pard by adding a standing order.)	□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □
📇 Plan	Properties User Defined Attachments Issues (5) Changes	Save	Discard Back
🌣 Execute	Summary		
쓵 Evaluate			
🖌 Configure	Description		
	Result	25	
	# Result S	Step Summary	Description Duration Action
	1 Passed Action Expected Result Precondition Postcondi Start the application by pressing the blue Con button.		00:00:04 🕖
	2 2 Inconclusive Action Expected Result Precondition Postcondit Select the (Standing Order) button that is found in the main menu.	lion	00:00:02 🥔

Figure 9.23. The "Test Case Result" Page

9.4.4.4.1. Actions

Auf den Detailseiten lassen sich jeweils folgende Aktionen vornehmen:

Execute this single test case/suite	Re-execute the test case or the test suite.			
- Execute this single test case/suite	Re-execute the test case or the test suite.			
읍 Open print view	A print-ready view of the test case result can be created here. With a click on the icon 🖨 this opens in a new browser tab.			
	Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .			
☐ Create Bookmarks	Requirements may be linked to from outside Klaros-Test- management using the link on the \square icon. By clicking on the \square icon this link will be copied to the clipboard.			
	Weitere Informationen hierzu finden Sie unter <u>Sec-</u> tion 5.2.3.2.2. "Bookmarks".			

G D Browse

Use the green arrows at the very top right to switch between the test case results present on the previous page.

The Summary, Description and Result of the test case result and each individual test step result can be edited here. All changes are automatically logged and any change of a result will cause a recalculation of the corresponding test case result, test suite result and test run. If a test case result has been changed, the ? icon is displayed on this page and in all tables where the result is displayed. The *Result* field of a test case result or test step result may only be changed in Klaros-Testmanagement Enterprise Edition.

≡ 📌 K L A R	OS TEST MANAGEMENT Finance Tracker		۹	• ≛•
📝 Define	🛕 🖗 Test Case Result: TCR0002460 (TC00022 - Update dashboard by adding a standing order.)		₽ •⊖	
		Save	Discard	Back
📇 Plan	Properties User Defined Attachments Issues (5) Changes			
🌣 Execute	Summary			
🔶 Evaluate	Description			
🔑 Configure				
	Result Y Failure Start 7/29/20, 8:46:29 AM Executor Sandra Selen Execution time 00:00:08 Iteration ITR00009 - 2.1.x-Sprint 01-Optimizing for use on smartwatches Test Run TRU0001730 - 7/29/20, 8:46:29 AM Test Case 1.0 - TC00022, Update dashboard by adding a standing order. Test Environment ExW00002 - Android Tablet System under Test SUT00006 - Finance Tracker 2:1.0 Steps			
	# Result Step	Summary	Description Dura	tion Action
	Constant of the second se		00:00	J:04 🖉
	2 24 C Error Expected Result Precondition Postcondition User Defined		00:00	0:02 🕜

Figure 9.24. Editing a Test Step Result

Print Options					
Details	Complete Brief 😨	Print			
Test Case Res	sult TC00005: Check account balance.				
Properties					
Result	▲ Failure: Button "Check Account Balance" is not clickable.				
Summary	Button "Check Account Balance" is not clickable.				
Start	8/4/21, 3:51:27 PM				
Executor	Felix Mustermann				
Execution time	00:03:15				
Test Run	TRU0010087 - 8/4/21, 3:51:27 PM				
Test Case	TC00005 - Check account balance.				
Test Environme	nt ENVUUUUS - Android 8 Smartphone				
System under 1	est Sofoooo-Philance Hacker 2.1.0				
Steps					
Step	1/2 O Passed				
Action	Start the application by <i>pressing</i> the blue icon ^{Solution} button.				
Expected Result	The start screen appears .				
Precondition	The application is installed on the device. If not, follow directions: https://finanztracker/install				
Postcondition	The application has started successfully.				
Step	2/2 🛕 Failure				
Description	The button is inactiv.				
Summary	Can not click on button "Check Account Balance"				
Action	Select the Check Account Balant button that is found in the main menu.				
Expected Result	The Account Balance is shown .				

Figure 9.25. The "Test Case Result" Print Page

9.4.4.5. Reasons for Skipped Test Cases

≡ 🖞 K L A R	OS TEST MANAGEMENT	Finance Tracker 🧧 📑	۹ 0∘ ∎۰
📝 Define	Test Run: TRU0009201 (SUT00006 - Finance Tracker 2.1.0 / ENV00005 - And	droid 8 Smartphone) - 4/23/21, 11:59:52 AM	🎝 🖓 🖓 🖓
📇 Plan	4 Entries-Page 1 of 1 K ◀ 1 ► K	l 10 ✓ V nt ≑ System under Test ≑ Executor ≑ Durat	Q × ⅔ ☰
🏟 Execute	TCR0012749 4/23/21, 11:59:52 AM TC00006 Android 8 Smartphone TCR0012748 () 4/23/21, 11:59:52 AM TC00007 Android 8 Smartphone	e Finance Tracker 2.1.0 Felix Mustermann 00 e Finance Tracker 2.1.0 Felix Mustermann 5	:00:06 🔮 🍰 🖨 Q
븑 Evaluate	TCR0012747 Skipped for the following reasons: P8 Android 8 Smartphone TCR0012746 Skipped for the following reasons: p9 Android 8 Smartphone	e Finance Tracker 2.1.0 Felix Mustermann 00 e Finance Tracker 2.1.0 Felix Mustermann 00	:00:06 🔮 🌲 🖨 Q :00:05 🥝 🏭 🖨 Q
🖌 Configure	- Preconation is not rumilea.		Back



If a test case has been skipped, the tester can enter one or more reasons from predefined templates (see Section 10.4.3, "Test Execution" and Section 8.3.3.1.1, "Skip Test Cases Permanently"). The icon will be displayed on appropriate pages of the *Evaluate* section. Hovering over this icon will display the predefined reasons in a tooltip.

9.5. Test Suite Results

The Test Suite Results page shows an overview of all test suites, including the number of results.

= 👱 K L A R	os test management	Finance Tracker 🖬 📰 🔍 🔍 🗙 🖓 V 💄 V
📝 Define	Test Suite Results	
🙁 Plan	7 Entries - Page 1 of 1 H 4 1 H 10 V	문장 ♥ 문화 ♣ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥ ♥
🏟 Execute	TS00007 1.0 24 Check the connection to the server TS00007 1.0 24 Check the connection to the server	1 3 1 1 5 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
쓵 Evaluate	TS00006 1.0 Overview TS00004 1.0 Connection TS00003 1.0 ▲ Display activities of different time periods	7 19 11 4 4 a a $Q2 0 a Q4 3 2 1 a Q$
🔑 Configure	TS00002 1.0 O Bank Transfer TS00001 1.0 O Update Dashboard	
	io revision Description	

Figure 9.27. The "Test Suite Results" Page

The table shows the following values:

ID	The automatically assigned ID.
Revision	The revision of the test suite.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
Description	The description of the test suite.
Test Cases	The number of test cases.
Results	The number of test suite results.
Action	The executable actions.

9.5.1. Action

The following actions can be performed in the action column:

- **Re-execute the test suite.**
- Import results.
- Q Show the test suite results of this test suite (see <u>Section 9.5.3, "Single Test Suite Results"</u>).
- Delete

9.5.2. Table Operations

The following operations can be performed in the line above the table on the right:

♀ Filter / Sort
 Show Executed Only/Show All Test Suites
 Q Search
 ▲ Export
 ➡ Column selection
 All operations are described in detail in Section 5.2.3.1, "Overview Page".

9.5.3. Single Test Suite Results

The "Test Suite Results" page shows all executions of the selected test suite, including the result, the used system under test and the test environment (<u>Figure 9.28</u>).



Figure 9.28. The "Test Suite Results" Page for a Single Test Suite

The collapsible panel on top of the page shows the details of the selected test suite. Below it is a table with all results.

The table shows the following values:

ID	The automatically assigned ID.
Start	The start time of the test run.
Test Environment	The test environment in which the test was executed
System Under Test	The system under test that was tested.

Executor	The name of the user who last executed the test run.
Duration	The execution time.
Result	The test case result (Passed, Failed, Error, Inconclusive or Skipped).
Action	The executable actions.

9.5.3.1. Action

The following actions can be performed in the action column:

Le Re-execute this test suite.

-S Import results.

Q Show the detail page of the test suite result (see <u>Section 9.5.3.4, "The "Test Suite Result"</u> <u>Details Page "</u>).

前 Delete

9.5.3.2. Bulk Actions

Certain actions can also be applied to several test suite results at once. To do this, select the test suite results to which the action is to be applied in the leftmost column.

The following bulk actions are supported for test suite results:

Open print view

🗊 Delete

For additional information, see Section 5.2.3.1.5, "Bulk Actions".

9.5.3.3. Table Operations

The following operations can be performed in the line above the table on the right:

- √ Filter / Sort
- Q Search
- 凸 Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

9.5.3.4. The "Test Suite Result" Details Page

This page shows a single test suite result and its test case results.

≡ 👱 K L A R	OS TEST	MANA	GEMENT				Finance Tracker 🔤	100		q x	0~ 1 ~
📝 Define	🔀 Test Suite	e Result: TS	SR0000313 (TS0000	7 - Edit stan	ding orders)				•=	
😂 Plan											
			3 E	ntries - Page 1 of	i M 4	1 ▶ ▶ 10 ♥		Y		Q	× & ≡
🏚 Execute	ID 🗢 🛛 🖓	Start 🖨	Test Case 🗢	Test Run 🗘	Iteration 🖨	Test Environment 🖨	System under Test 🖨	Executor 🗢	Duration 🖨	Result 🖨	Action
_	TCR0002471	4 years ago	Delete standing orders.	TRU0001735		Android 8 Smartphone	Finance Tracker 2.1.0	Felix Mustermann	00:00:04	0	₽ ⊕ Q
, Evaluate	TCR0002470	4 years ago	Change a standing order.	TRU0001735		Android 8 Smartphone	Finance Tracker 2.1.0	Sandra Selen	00:00:05	>\$	₽ ⊕ Q
	TCR0002469	4 years ago	Add a standing order.	TRU0001735		Android 8 Smartphone	Finance Tracker 2.1.0	Felix Mustermann	00:00:08	Ø	₽ ₽ Q
✗ Configure	ID	Start	Test Case	Test Run	Iteration	Test Environment	System under Test	Executor	Duration	Result	Action
J configure											Back
	Created 4 years a	igo by Felix Mu	stermann					Las	t changed 3 ye	ears ago by	Sandra Selen

Figure 9.29. The "Test Suite Result" Details Page

9.5.3.4.1. Actions

Auf den Detailseiten lassen sich jeweils folgende Aktionen vornehmen:

.	Execute this single test suite	Re-execute the test suite.
-5	Execute this single test suite	Re-execute the test suite.
₿	Open print view	A print-ready view of the test suite result can be created here. With a click on the icon \bigoplus this opens in a new browser tab.
		Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .
☐ Create Bookmarks		Requirements may be linked to from outside Klaros-Test- management using the link on the \square icon. By clicking on the \square icon this link will be copied to the clipboard.
		Weitere Informationen hierzu finden Sie unter <u>Sec-</u> tion 5.2.3.2.2, "Bookmarks".
G	Browse	Use the green arrows at the very top right to switch between the test suite results present on the previous page.

Print Optio Suppress er Details	ons mpty fields ✓ Complete Brief ♡ Print						
Test Suite Result TSR0001874							
Description: Result: Serror (Passed: 0, Failure: 0, Error: 2, Skipped: 0)							
Result:	S Error: Could not export deployment to temp						
ID:	CR0013920						
Start:	8/4/21, 4:18:51 PM						
Test Case:	1.0 - TC00018, Connection to the server of the bank.						
Test Run:	TRU0010089						
Test Environment:	ENV00005 - Android 8 Smartphone						
System under Test:	SUT00006 - Finance Tracker 2.1.0						
Executor:	Felix Mustermann						
Result:	S Error: Could not export deployment to temp						
ID:	TCR0013919						
Start:	8/4/21, 4:18:51 PM						
Test Case:	1.0 - TC00019, Connection to the database.						
Test Run:	TRU0010089						
Test Environment:	ENV00005 - Android 8 Smartphone						
System under Test:	SUT00006 - Finance Tracker 2.1.0						
Executor:	Felix Mustermann						

Figure 9.30. The "Test Suite Result" Print View

9.6. Issues

Issues from external issue management systems (like Jira or GitLab) can be linked with test cases and systems under test in Klaros-Testmanagement. In addition to this, issues can be created from Klaros-Testmanagement in connected issue management systems (even during test execution).

For this, at least one issue management system must be configured and assigned to the desired project. The configuration of an issue management system is described in <u>Section 10.5.1, "Issue Management"</u>, <u>Section 6.1.2.6.1, "Issue Management"</u> shows how to assign the issue management system to a project.

9.6.1. Overview Page

The overview page displays all issues present in the project in a table. This chapter shows how to create, edit and edit issues.

Pressing the New button opens up the <u>Section 9.6.6, "Issue Details (Creating a new Issue)"</u> page.

Pressing the Link button opens up the Link Issues panel, in which existing issues from issue management systems can be linked to test cases and systems under test (see <u>Section 9.6.7, "Link Issue"</u>).

= 👱 K L A R (o s test manageme	E N T		Finance Tracker 🗧		Q	× 0~ ±~
📝 Define	Issues						S
: ⊈t Plan	New Link					Save	Discard
	₽ û	12 Entries - Page 1 of 2	M ┥ 1 2 🕨 M 10 🗸		Show	/ all	Q × & ≡
💏 Evecute	ID♦ 🛛 System ♦	Summary 🗢	Test Cases	Systems under Test	Assigned To 🖨	Priority 🗘 Status	Action
	PLAYGROUND-25702 🙀 IM00008	test Issuevtvttvtv	Update dashboard by adding a standing order.	Finance Tracker 2.1.0	admin	Blocker Open	┏┎╘р
쓵 Evaluate	PLAYGROUND-25702 🙀 IM00008	test Issuevtvttvtv	Update dashboard by adding a standing order.	Finance Tracker 2.0.1	admin	Blocker Open	┏┎╘ॖѱ
🔑 Configure	PLAYGROUND-25702 🙀 IM00008	test Issuevtvttvtv	Update dashboard by adding a standing order.	Finance Tracker 1.1.1	admin	Blocker Open	┏┎╘ॖ╓
	PLAYGROUND-25702 🙀 IM00008	test Issuevtvttvtv	Update dashboard by adding a standing order.		admin	Blocker Open	┏┎╘ॖѱ
	PLAYGROUND-25701 🙀 IM00008	test Issue	Update dashboard by adding a standing order.		gidley	Blocker Open	┏ ┎ 🖨 🗇
	PLAYGROUND-25700 🙀 IM00008	test Issue	A Check the connection to the server		admin	Critical Open	┏┎╘р
	PLAYGROUND-25699 🙀 IM00008	test Issue	Check the connection to the server		admin	Blocker Open	௴ └ ⊖ @
	PLAYGROUND-22965 🛱 IM00008	Closed Issue for Test Data	Update dashboard by cancelling a standing order.	Finance Tracker 2.1.0	hudson	Major Closed	┏┎╘╒╖
	PLAYGROUND-22965 🛱 IM00008	Closed Issue for Test Data	A Check the connection to the server		hudson	Major Closed	┏ ┖ 🖨 🛍
	PLAYGROUND-22963 🙀 IM00008	Closed Issue for Test Data	Change a standing order.		hudson	Major Closed	┏┎╘┇
	ID System	Summary	Test Cases	Systems under Test	Assigned To	Priority Status	Action
	New Link					Save	Discard

Figure 9.31. The "Issues" Page

The table shows the following values:

ID	The external issue Id.
Additional Information	On the one hand, the symbol of the issue management system is displayed here. Clicking this icon opens the issue in the issue management system. On the other hand, a \triangle can appear. In this case, the administrator should be informed immediately. Reasons for displaying the \triangle can be among others:
	Deactivation of the issue management system.
	 Incorrect configuration of the issue management system in Klaros-Testmanagement or in the issue management sys- tem. The configuration of an issue management system is described in <u>Section 10.5.1, "Issue Management"</u>. A tooltip appears when the cursor is placed over the icon shown here.
System	The ID of the issue management system.
Summary	The summary of the issue.
Test Cases	A list of test cases assigned to this issue with the result of their latest execution.
Systems under Test	A list of systems under test assigned to this issue.
Assigned To	The username of the responsible user.
Priority	The priority of the issue.
State	The state of the issue.

Action

The executable actions.

9.6.2. Action

The following actions can be performed in the action column:

🖉 Edit

- Open issue in external issue management system
- Open print view
- 🗊 Delete

If a issue has been deleted, it is internally marked with a deletion marker and is only visible to administrators. For deleted issues, the following actions are available instead of *Delete*:

- F Restore
- \diamond Remove the issue from the database



Important

Pressing the Δ icon does not delete the issue in the issue management system, only the reference in the database!

9.6.3. Bulk Actions

On the *Issue* page, one or more issues can be selected for bulk actions. Bulk actions are described in <u>Section 5.2.3.1.5, "Bulk Actions"</u>.

The following bulk actions are supported for issues:

- Open print view
- 🗊 Delete
- Restore the issue (only Administrator).
- ☆ Remove the issue from the database (only Administrator).

For additional information, see Section 5.2.3.1.5, "Bulk Actions".

9.6.4. Table Operations

The following operations can be performed in the line above the table on the right:

√ Filter / Sort

Show all / Only active (only Administrator)

- Q Search
- 옰 Export
- ⊟ Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

9.6.5. Details Page

Each issue has its own detail page with several additional tabs. Clicking on the ID of the respective requirement or on the 😰 icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Properties* tab.

The following tabs are available: Properties, Test Cases, Systems under Test and Changes.

9.6.5.1. Actions

On the detail pages there are more icons in the upper right corner. The following actions can be performed here:

🖸 Open external link	Shows the issue in the external issue management system.
Synchronize	Synchronizes the local issue with the external source.
🔒 Open print view	A print-ready view of the issue can be created here. With a click on the icon 🔒 this opens in a new browser tab.
	Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .
☐ Create Bookmarks	Issues may be linked to from outside of Klaros-Test- management using the link on the \square icon. By clicking on the \square icon this link will be copied to the clipboard.
	Bookmarks are described in detail in <u>Section 5.2.3.2.2, "Book-marks"</u> .
G S Browse	Use the green arrows at the very top right to switch between the issues present on the previous page.

9.6.5.2. Properties

In this tab (<u>Figure 9.32</u>) several attributes of the issues are displayed and can be edited if the user has the appropriate permission.

= 👱 K L A R	o s test manac		Finance Tracker 🔤 📑	० × 0 ~ ≛~
📝 Define	PLAYGROUND-22965 - Cl	osed Issue for Test Data		C ⇔ □ <
🍰 Plan	Properties Test Cases (1)	Systems under Test Changes		Save Discard Back
🔹 Execute	Issue Management System	IM00008 - Jira (legacy), PLAYGROUND, Klaros 🔀	🗉 (admin)	
쓵 Evaluate	ID Status	PLAYGROUND-22965 Closed		X
🖋 Configure	Summary *	Closed Issue for Test Data		
	Priority	Maior 🗸		
	Components	~		
	Affected Versions	~		
	Fixed Versions	~		
	Assignee	v		
	Due	Ξ		
	Description	Open and Closed Issue Test Cases		
		(3a)		
		TC00078; TC00081; TC00084; TC00089; TC00094	; TC00097;	
	Include Test Case Links			
	Environment			



The issue *ID* is a hyperlink to the corresponding issue in the issue management system.

9.6.5.3. Test Cases

This view lists all test cases that are associated with this issue.



Figure 9.33. The "Test Cases" Tab

Test cases can be assigned to the issues by clicking the Assign button. This opens up a dialog, where multiple test cases can be selected at once.



Figure 9.34. The "Assign Test Cases" Dialog

The following bulk actions are supported for test cases:

– Remove

The detail page of a test case shows a list of all issues that are associated with it, as shown in section <u>Section 6.7.2.8, "Issues"</u>.

9.6.5.4. Systems under Test

The page lists all systems under test assigned to the issue.

If you want to associate systems under test with the issue, click the Assign button and select one or more systems under test.

The following bulk actions are supported for systems under test:

- Remove

The detail page of a system under test shows a list of all issues that are associated with it, as shown in section <u>6.5.2.6</u>, "Issues".

≡ 📌 K L A R	o s test management	Finance Tracker 👕 🗮	Q x 00 × ≜×
📝 Define	PLAYGROUND-22965 - Closed Issue for Test Data		໕ຬᇢຐຩໟ
🏝 Plan	Properties Test Cases (1) Systems under Test Changes	Save	Discard Back
🏟 Execute	- 0 Entries - Page 1 of 1 🖌 📥 🕨 10 🗸	کر ا	Q × ♣ Ξ
, Evaluate	Version Ver		Action
🖌 Configure			Action
ar configure	Created 6 years ago by Felix Mustermann	Last	changed 22 days ago by System Account
	Assign	Save	Discard Back

Figure 9.35. The "Systems under test" Tab

Evaluate

≡ 👱 K L A R () S test management	Finance Tracker 🖬 🔜	Q × Q × ≜×
📝 Define	PLAYGROUND-22965 - Closed Issue for Test Data		◩▱◓▯◶◓
📇 Plan	Properties Test Cases (1) Systems under Test Changes	Save	Discard Back
🏟 Execute	- 0 Entries - Page 1 of 1 N 4 N 10 V	Y	Q × & Ξ
쓵 Evaluate	ID ÷ Version ÷ No entries available Moreign		Action
🔑 Configure	C Are you sure?		d 5 months ago by System Account
			Discard Back
	6 Entries - Page 1 of 1 🙀 🚽 1 🕨 🕅 10 🗸	∇ Q × & Ξ	
	■ ID\$ Version \$		
	SUT00006 Finance Tracker 2.1.0		
	SUT00005 Finance Tracker 2.0.1		
	SUT00004 Finance Tracker 2.0.0		
	SUT00002 Finance Tracker 1.1.0		
	SUT00001 Finance Tracker 1.0.0		
		Assign Cancel	

Figure 9.36. The "Assign Systems under test" Dialog

9.6.5.5. Changes

The tab Changes shows the change history of this issue.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

9.6.6. Issue Details (Creating a new Issue)

= 🖞 KLARC) s test manag	EMENT	Finance Tracker 🖻 🗮	५ × ₽ ~ ≛~
🕑 Define				윤 🖨 🛛 😔 ᢒ
🏩 Plan	Properties		Save	Discard Back
🏟 Execute	Issue Management System	IM00008 - PLAYGROUND 🗸 🚺 (admin)		*
쓵 Evaluate	Status	Open		~
4	Summary *	Confirm button is not clickable.		
Configure	Issue Type	Bug 💙		
	Priority	Blocker 💙		
	Components	~		
	Affected Versions	~		
	Fixed Versions	~		
	Assignee	V		
	Due			
	Description	After editing a standing order, the confirm button is not clickable.		
	Environment	Android Tablet, ID:A76		



The first defined issue management system of the active project is automatically selected from the *Issue-Management-System* dropdown list.



Note

If the list of issue management systems is empty, then at least one issue management system has to be configured and linked to the active project. The configuration of issue management systems is described in <u>Section 10.5.1</u>, "Issue Management". <u>Section 6.1.2.6.1</u>, "Issue Management" contains instructions on how to add an issue management system to a project.

By default Klaros-Testmanagement uses the credentials entered at the login screen to authenticate the user to the issue management system. When using Bugzilla, the email/password combination of the user is tried as well, as this is the common case for a login name there. If the credentials entered at login do not work and the issue management system has not yet been used during the session, Klaros-Testmanagement will request new authentication information in a dialog window.



Token based authentication (Jira Cloud instances, GitLab)

Klaros-Testmanagement is supporting token based authentication scheme as well. Just use an empty username and the token as password.

Some Vendors like Jira and GitLab limit access to token based authentication Authenticating via username and password is no longer supported for these systems.

Clicking the \square icon opens up the external issue management system in a new tab.

Clicking the 🗊 icon opens up a dialog to change the authentication information.

If an issue management system has been selected, a list of issue fields is shown. The exact number and names of those fields vary, depending on the type and configuration of the issue management system. Mandatory fields are shaded in a different color.



Note

All mandatory fields need to be filled, otherwise the issue cannot be saved.

Issue Fields for every Issue-Management-System

Test Cases	The test cases this issue is linked to.
Systems under Test	The systems under test this issue is linked to.
Update Test Case State	Selecting this checkbox opens up a new dropdown list in which the new state of the linked test cases can be selected.

9.6.6.1. Creating a Bugzilla Issue

Issue Management System	IM00012 - TestProduct	t 💙 🗹	📧 (root@v	erit.de)		<u>}</u>
ID						
Status	UNCONFIRMED 🖌					
Summary *						
Platform	All 🗸					
Operating System	All 💙					
Priority	Highest 💙					
Importance	blocker 🗸					
Components	Left Component 💌					
Version	unspecified 💙					
Milestone	💙					
Description						
Description						
Description (generated)	TC00001 - Check acco	ount balance				
	 http://localbost:8080/l	klaros-web/b	rowse isp?tv	na-tastCasal	uuid=37101c2c.cd9	0.4958-901c-
	2ccd9079584f	Naros Neb/D	101136.J3p.ty	pe-restoused	4414-57101020-045	04930-9010
Large Text						
Multiple Select	~					
Drop down	💙					
Bug ID						
Free Text						
Integer						
Date						
Date/Time	苗					
Test Case	TC00001, Check rese	unt balance	v u			
System under Test	Finance Tracket VIC 0		^ ·			
Undate Test Case State?	Finance tracker-v16.0.4	4 🗙	~			
opuate rest dase state?	•					
* Mandatory fields must be p	rovided					
				Save	Discard	Back
				Jure	Dioodra	Duck

Figure 9.38. The "Bugzilla Issue " Page

9.6.6.2. Creating a GitHub Issue

Issue Management System	IM00009 V 🗹 🖭 (·······)
ID	
Status	open 🗸
Summary *	
Assignee	
Milestone	V
Description	
Description (generated)	TC00001 Check account balance
	http://localhost:8080/klaros-web/browse.jsp?type=testCase&uuid=37101c2c-cd90-4958-901c- 2ccd9079584f
Test Case	TC00001 - Check account balance X
System under Test	FinanceTracker-V16.0.4 X
Update Test Case State?	v
* Mandatory fields must be p	rovided
	Save Discard Back

Figure 9.39. The "GiHub Issue" Page

9.6.6.3. Creating a GitLab Issue

issue management system	IM00010 💙 🗹 🗉 🗛 🗛
ID	v
0 states	
Status	opened 💙
Summary *	
Labels:	
Assignee	V
Milestone	v
Description	
Description (generated)	
	TC00001 - Check account balance
	http://localhost:8080/klaros-web/browse.jsp?type=testCase&uuid=37101c2c-cd90-4958-901c-
	200390795841
Test Case	TC00001 - Check account balance
Test Case System under Test	TC00001 - Check account balance X V
Test Case System under Test Undate Test Case State?	TC00001 - Check account balance X V FinanceTracker-V16.0.4 X V
Test Case System under Test Update Test Case State?	TC00001 - Check account balance X Y FinanceTracker-V16.0.4 X Y
Test Case System under Test Update Test Case State? * Mandatory fields must be pr	TC00001 - Check account balance X Y FinanceTracker-V16.0.4 X Y
Test Case System under Test Update Test Case State? * Mandatory fields must be pr	TC00001 - Check account balance X Y FinanceTracker-V16.0.4 X V V
Test Case System under Test Update Test Case State? * Mandatory fields must be pr	TC00001 - Check account balance X Y FinanceTracker-V16.0.4 X V rovided
Test Case System under Test Update Test Case State? * Mandatory fields must be pr	TC00001 - Check account balance X FinanceTracker-V16.0.4 X V rovided

Figure 9.40. The "GitLab Issue" Page

9.6.6.4. Creating a Jira Issue

Issue Management System	IM00006 - PLAYGROUND 💙 🗹 🖭 (builder)
ID	· · · · · · · · · · · · · · · · · · ·
Summary *	
Issue Type	Task Y
Priority	
Components	V
Affected Versions	
Fixed Versions	
Assignee	AUTOMATIC
Due	
Description	
Description (generated)	T000001 Obsels securit belance
	http://localhost:8080/klaros-web/browse.jsp?type=testCase&uuid=37101c2c-cd90-4958-901c- 2ccd9079584f
Environment	
Sprint	V
Epic Link	
JiraClientCustom	
Test Case	TC00001 - Check account balance
System under Test	FinanceTracker-V16.0.4 X
Update Test Case State?	×
Test Environment	✓
* Mandatory fields must be n	rovided
and a construction of the p	
	Save Discard Back

Figure 9.41. The "Jira Issue" Page

9.6.6.5. Creating a Mantis Issue

Issue Management System ID	
Status	new 🗸
Summary *	
Reproducibility	always
Priority	none V
Severity	feature Y
Product Version	2.0 🗸
Target Version	2.0 🗸
Fix Version	2.0 🗸
Assignee	
Category	General 🗸
Description	
Description (generated)	
	TC00001 - Check account balance
	http://localhost:8080/klaros-web/browse.jsp?type=testCase&uuid=37101c2c-cd90-4958-901c- 2ccd9079584f
Steps To Reproduce	
Additional Information	
DATE	曲
ENUM	1 🗸
FLOAT	
NUMERIC	
MULTILIST	×
TEXTBOX	
RADIO	0 1 0 2
STRING	
Checkboxes are fun	V
EMAIL	
LIST	1 🗸
Test Case	
Custom under Taat	
System under Test	Finance tracker-V16.0.4 X
update rest case State?	▼
* Mandatory fields must be p	rovided

Figure 9.42. The "Mantis Issue" Page

9.6.6.6. Creating a Redmine Issue

Issue Management System	IM00007 - PLAYGROUND 🗡 🗹 🖬 (builder)	****		
ID				
Cummony *				
Summary *				
issue type	Bug 🗸			
Priority	Low			
Assignee	~			
Version	~			
Category	Left Category 💙			
Status	New 🗸			
Progress	0% 🗸			
Start	曲			
Due	曲			
Description				
Description (generated)				
	TC00001 - Check account balance			
	 http://localhost:8080/klaros-web/browse.jsp?type=testCase&uuid=37101c2c-cd90-4958-901c-			
	2ccd9079584f			
Entimated time				
Esumated ume				
Sung				
Boolean	-			
Float				
Tout				
List				
LISU				
ιητ				
Test Case	TC00001 - Check account balance			
System under Test				
System under rest	Finance Tracker-V 16.0.4			
opuate rest case state?	~			
* Mandatory fields must be p	rovided			
	Save	Discard Back		

Figure 9.43. The "Redmine Issue" Page

9.6.6.7. Creating a Trac Issue

Issue Management System	IM00013 🔽 🖾 (root@verit.de)
ID	· · · · · · · · · · · · · · · · · · ·
Summary *	
Issue Type	defect V
Priority	blocker ¥
Components	
Owner	
Version	v
Milestone	v
Description	
Description (generated)	
	TC00001 - Check account balance
	http://localhost:8080/klaros-web/browse.jsp?type=testCase&uuid=37101c2c-cd90-4958-901c- 2ccd0070584f
	2003073004
Keywords	
test_three	
test_four	two 🗸
test_one	
test_two	
test_five	◯ uno ● dos ◯ tres ◯ cuatro ◯ cinco
test_six	Default text
Test Case	TC00001, Check account halance
Svetom under Teet	
Undate Test Case State?	
opuale resi case state?	Y
* Mandatory fields must be p	rovided
	Dava Diagond De-tr
	Save Discard Back

Figure 9.44. The "Trac Issue"Page

9.6.6.8. Creating a YouTrack Issue

Issue Management System ID	
Summary *	
Description	
Description (generated)	TC00001 - Check account balance
Priority	Show-stopper V
Туре	Bug
State	Submitted ¥
Subsystem	No Subsystem 💙
Fix versions	×
Affected versions	~
Fixed in build	Next Build 💙
Assignee	Unassigned 💙
Estimation	
Spent time	
Test Case	TC00001 - Check account balance X
System under Test	FinanceTracker-V16.0.4 X
Update Test Case State?	v
* Mandatory fields must be p	vrovided
	Save Discard Back

Figure 9.45. The "YouTrack Issue" Page

9.6.7. Link Issue

In addition to creating issues from within Klaros-Testmanagement, it is possible to link issues from an issue management system to test cases and systems under test in Klaros-Testmanagement.

≡ 👱 K L A R	o s test management		Finance Tracker 🖀 📑	Q x Q × ≗ ×
📝 Define	Issues			S
北 Plan	New Link Link Link			Save Discard
🅸 Execute	System IM00008 - PLAYGROUND V 🔀 🖬 (admin)	2 ×		
쓵 Evaluate				
🖌 Configure	Link			Close
	□ 12 Entries - Page 1 of □ ID	2 Test Cases	Systems under Test Assigned To 🕈	rall Q X 23 Ξ
	PLAYGROUND-25702 🙀 IM00008 test Issue	Update dashboard by adding a standing order.	Finance Tracker 2.1.0 admin	Blocker Open 🗹 🖒 🛱 🛍
	PLAYGROUND-25702 🙀 IM00008 test Issue	Update dashboard by adding a standing order.	Finance Tracker 2.0.1 admin	Blocker Open 🕜 🗹 🖨 🔟



To link an issue to a test case or sytem under test, an issue management system must be selected first. Pressing the Link button links the issue with the entered *ID* to the selected test cases and systems under test.

System	The issue management system from which issues are to be
	linked.

ID The id of the issue to be linked.

Pressing the Q icon searches the issue management system for an issue that matches the entered id. Upon a successful search, pressing the Link button links the issue to the selected test cases and systems under test.

Chapter 10. Configure

In the *Configure* section, all administrative tasks are performed. These include the **system configuration**, managing **custom reports**, creating and managing **user accounts**, changing **application settings**, configuring **remote system** parameters and **backing up and restoring** projects.

To access the majority of the functions described here, the global Administrator role is required.

10.1. Overview

On the *Overview* page you will find the following information:

- Server
- Database
- Operating System
- Locale
- Java
- Memory
- License
- Version





10.2. Report Templates



Only available in Klaros-Testmanagement Enterprise Edition

On the *Report Templates* page, user-defined report templates can be created, edited, cloned and reports can be generated in PDF or Excel format.



Note

The predefined report templates are not editable and should be cloned for editing.



Figure 10.2. The "Report Templates" Page

The table shows the following values:

Additional Information	A tooltip appears when the cursor is placed over the icor shown here.	
Name	The name of the report template.	
Description	The description of the report template.	
Revision	The revision of the report template.	
Status	The status of the report template (draft, approved).	
Format	The output format of the report (PDF or Excel).	
Action	The executable actions.	



Note

The Additional Information column indicates whether a report template is locked for editing.

10.2.1. Action

The following actions can be performed in the action column:

- Create the report as a PDF file.
- Create the report as an Excel file.
- 🕜 Edit
- Duplicate
- 🗊 Delete

If a report is to be generated that requires the input of parameters, a dialog opens. After confirmation with OK the report is generated. The button remains locked if mandatory parameters are missing user input. The Cancel button cancels the action.







Note

Clicking the New button opens the page Report Templates Details. (Figure 10.4).

10.2.2. Report Templates Details

On the *Report Template Details* page, the name and description of the report template can be specified. A valid script and layout template must also be specified. The script and layout template can be edited directly in the text area or uploaded as an external file. In the Revisions tab, older versions of the report can be restored and differences to a previous version can be listed.

= 🞐 K L A R G	DS TEST MA	NAGEMENT		Finance Tracker 📱 📑		द × 0∕ ≛×
📝 Define	RT00007 - Issue Ov	verview Report				6 6
🏖 Plan	Properties Parame	eters (3) Script Layout Tem	nplate Revisions Changes	Preview	Save Discard	Back
🔹 Execute	Name	Label	Description	Туре	Default	Mandatory Action
	sortBy	Sort By	The sort parameters for the report	List	Creation Date	1
🕒 Evaluate	onlyOpen	Only Unresolved Issues	List only the issues that are not yet resolved	Boolean	\checkmark	1 🗊
	testCaseDetail	Show Test Case Details	Show additional information for test cases	Boolean	\checkmark	1 🗊
🔑 Configure	New Parame	eter				
				Preview	Save Discard	Back





Integrated Syntax-Check

To be able to save the report template, the script and the layout template must not contain any syntactical errors. This is checked before saving.

Detailed information about creating user-defined reports is available in <u>Section 11.2, "Creating a</u> <u>Report Template"</u>.

10.3. Users

The user accounts are created and edited in the Users section.

10.3.1. Overview Page

The overview page gives an overview of the user accounts in Klaros-Testmanagement. The user accounts are created, edited, merged, activated and deactivated here.

E 📌 KLARO	o s test management	Finance Tracker 🖬 🗮	५ ४ 0 ४ ⊥ ४
📝 Define	Users		
💐 Plan	New		Save Discard
_	[2] 🖨 ⊠ 📚 û	21 Entries - Page 1 of 3 🛛 ┥ 🧵 2 3 🕨 🕅 10 🗸	√ Show all & ≡
🚓 Evocuto	□ Q Username	Full Name ♦ E-Mail ♦	System Account 🗢 🛛 Action
	🗌 🛕 admin 🛛 Administrator Felix Mustermann		ピ 🛱 🛍
	gross Guest Günther Gross	gross@mail.de	び合血
C Evaluate	koennecke Administrator Claudia Könnneck	e	ぱ 🖨 🛍
	🗌 🛕 manager 🛛 Test Manager Max Mustermann		ぱ 🖨 🛍
🔑 Configure	Meyer Test Manager Markus Meyer	meyer@mail.de	ぱ 合 前
	olli Tester Oliver Krams	te@mail.de	ぱ 合 前
	reilly Administrator Patrick Reilly		☞ 🖨 🛍
	Sabrina Administrator Sabrina Gidley	gidley@verit.de	
	sahra Tester Sahra Berger		ぱ 合 前
	selen Administrator Sandra Selen	arif@verit.de	
	Username Role	Full Name E-Mail	System Account Action
	New		Save Discard

Figure 10.5. The "Users" Page

The table shows the following values:

Additional Information	A tooltip appears when the cursor is placed over the icon shown here.	
Username	The unique username.	
Role	The user roles (Administrator, Test Manager, Tester, Guest).	
Full Name	The full name of the user.	
E-Mail	The email address of the user.	
Actions	The executable actions.	

10.3.1.1. Actions

The action column is located on the far right of the table. The following actions can be performed here:

🖉 Edit

- Open print view
- Delete

If a user has been deleted, it is initially marked with a deletion marker and is only visible to administrators. For deleted users, the following action is available instead of *Delete*:

Restore (only Administrator)

10.3.1.2. Bulk Actions

Some actions can be applied to multiple user accounts at the same time. To do this, select the user accounts to which the action is to be applied in the leftmost column.

The following bulk actions are supported for user accounts:

🕜 Edit

Open print view

🖂 Send Email

Se Merge User Accounts

It is not possible to permanently delete users directly, but users can be merged. The user to be kept will be selected in the dialog (<u>Figure 10.6</u>). The selected user takes over the job assignments, test results and test runs of the other users. The other users will then be permanently deleted and are therefore no longer listed in the user list.

Restore (only Administrator)

Bulk actions are described in detail in Section 5.2.3.1.5, "Bulk Actions"

≡ 🙅 K L A R	d s test management	Finance Tracker 🖬 🚍	५ × 0 × ≛ ×
📝 Define	Users		
📲 Plan	New	N N 10.44	Save Discard
🕸 Execute	Image: Construction Image: Construction Image: Construction Role ◆ Image: Construction Full Name ◆ Image: Construction Administrator Image: Construction Administrator	E-Mail 🕈	System Account
🕒 Evaluate	A manager Test Manager Max Mustermann A tester Tester Erika Mustermann		
🔑 Configure	Are you sure? Ob you want to merge these users?		System Account Action Save Discard
	ID \$ Description \$ Felix Mustermann Username: admin - ID: Felix Mustermann Max Mustermann Username: manager - ID: Max Mustermann Erika Mustermann Username: tester - ID: Erika Mustermann Merged User • Marged User • Max All other Users will be purged after they have been merged. This process cannot be reverted.	Created \$ Changed \$ 6/14/2020 3/9/2022 6/14/2020 6/14/2020 6/14/2020 6/14/2020 6/14/2020 6/14/2020 Created Changed	

Figure 10.6. The "Merge User" Dialog

10.3.1.3. Table Operations

The following operations can be performed in the line above the table on the right:

Filter / Sort

Show all	/ Only active	(only Administrator)
----------	---------------	----------------------

- & Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".



Changing User Roles

Users - including administrators - cannot change their own *System Account* flag or *role*. This can only be done from another user account that has the *Administrator* role.

10.3.1.4. Creating a new User

Clicking the New button creates a new user account (<u>Figure 10.5</u>). The Username, Role, Full Name, and E-Mail attributes can be set here.

When the <u>Save</u> button is clicked, a dialog will appear for each new user account to set the *Password* and *System Account* flag (Figure 10.7).

≡ 📌 K L A R	OS TEST	MANAGEME	NT		Finance Tracker 🖀 🗮	९ छ ॰ ≛ ॰
📝 Define	Users					
	New					Save Discard
📇 Plan	₫ 🖨 🖂		21 En	tries-Page 1 of 3 🛛 🛋 1 2 3 🕨 🕅	10 🗸	♡ Only active 🕹 🗮
🌣 Execute	🔲 💡 Usernam	e \$ Role \$		Full Name 🗢	E-I	Mail ♦ Action
	🗌 🛕 admin	Admini: Additional in	nformation is r	required		
🕒 Evaluate	gross koenneck	e Adminis Plea	ase complete t	he information	@mail.de	ビ辛茴
🔑 Configure	🗌 🛕 manager	Test Ma	licornamo *	Henri		
	Meyer	Test Ma	Full Name *	Henri Herrmann	r@mail.de	
	reilly	Adminis New	Password *		lan.de	
	Sabrina	Admini: Retype new	v Password *		@verit.de	C 🖨 🛍
	sahra	Tester	Role	Test Manager 🗸		r 🖨 🗓
	Usernan	ne Ro	E-Mail	henri@herrmann.com	E	-Mail Action
	New	Syst	tem Account			Save Discard
				OK Cancel		

Figure 10.7. The "Save New User" Dialog

Per default users of the role Administrator or Manager are able to create new user accounts. Users of the role Manager can only create accounts of the role Tester or Guest.



Restricting Account Creation to Administrators

If a more restrictive policy is desired, the *Only Administrators can create Users* setting can be used. See <u>Section 10.4.1, "Miscellaneous"</u> for more information.

10.3.1.5. Deleting a User

Clicking the in icon page deactivates the user account. Only users of the role Administrator can see and reactivate deactivated user accounts.



User accounts can only be deactivated, not deleted!

By toggling the Only Active and Show All button above the table, administrators can hide or show deactivated user accounts.

Instead of deleting user accounts, they can be merged with other existing user accounts. This operation will delete all user accounts to be merged except the remaining account.

10.3.2. Details Page

Each user account has its own detail page with several additional tabs. Clicking on the ID of the respective user account or on the icon on the right in the action column takes you to the tab that was selected last. When called for the first time, this is the *Overview* tab.

= 👱 K L A R	ROS TEST MANAGEMENT	Finance Tracker 🔤 🗎	≡Q× ₽ ×
📝 Define	Sabrina Gidley		🖨 😋 😜
📇 Plan	Properties Jobs Results Changes		Save Discard Back
🔅 Execute	Username * Sabrina	New Password *	
Evaluate	Full Name * Sabrina Gidley	Retype new Password *	
	E-Mall gidley@verit.de	Role Administr	ator 🗸
🄑 Configure	Enabled		
	* Mandatory fields must be provided		
	Created 4 years ago by Felix Mustermann		Last changed 4 years ago by Felix Mustermann
			Save Discard Back

Figure 10.8. The "Properties" Tab

The following tabs are available: Properties, Project Roles, Jobs, Results and Changes.

10.3.2.1. Actions

On the detail pages, there are additional icons in the top right corner of the header. The following actions can be performed here:

🔒 Open print view		A print-ready view of the user account can be created here. With a click on the icon 🖨 this opens in a new browser tab.	
		Print views are described in detail in <u>Section 5.2.3.2.1, "Print</u> <u>Pages"</u> .	
g ð	Browse	Use the green arrows at the very top right to switch between the user accounts present on the previous page.	

10.3.2.2. Properties

This tab (<u>Figure 10.8</u>) allows the user to view or change the following attributes of the user account:

Username	The login name of the user
Full Name	The full name of the user
E-Mail	The user's email address which is used for email notifica- tions. If a corresponding event has been configured (see <u>Sec-</u> <u>tion 10.4.2, "Notifications"</u>) and an email address is specified

	when creating a new user (see <u>Section 10.3.1.4</u> , <u>"Creating a</u> <u>new User"</u>), this user will receive a registration email with user- name and password.
System Account	If this box is checked, this user will not be able to log in to the login page. System accounts are intended for automated tasks such as importing data.
Active	If this box is checked, this user will not be able to log in to the login page.
Password	The password to log in to Klaros-Testmanagement.
Retype Password	Confirmation of the password.
User Role	The user role. Available roles are <i>Administrator</i> , <i>Test Manager</i> , <i>Tester</i> or <i>Guest</i> .



Limitations when Changing User Roles

Users can not change their own *System Account* flag and *Role*. This must be done by another user with administrator role.

10.3.2.3. Project Roles



The *Project Roles* tab shows the project roles the user has in projects with project specific roles (see <u>Section 6.1.2.5, "Access"</u> for more information on how to assign project-specific roles to a project). Administrators can change the project roles for these projects here.

This tab is available once the first project has been assigned project-specific roles.

≡ 📌 K L A R 🤅	OS TEST MANAGEMENT Finance	e Tracker 🖬 🗮 🔍 🔍 🕹 🗸
🕑 Define	Oliver Krams	🖨 G S
📇 Plan	Properties Project Roles (1) Jobs Results Changes	Save Discard Back
🕸 Execute	2 1 Entries-Page 1 of 1 M 4 1 D 10 V] \\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
🕒 Evaluate	P00003 Printer Tester	Tester 🗸
🔑 Configure	ID Description Created 4 years ago by Felix Mustermann	
	Assign	Last changed 4 years ago by Felix Mustermann Save Discard Back

Figure 10.9. The "Project Roles" Tab

Pressing the Assign button opens up a dialog with a list of all projects with project specific roles that the user isn't assigned to yet. Pressing the Ok button assigns the user to the selected projects.

≡ 📌 K L A R	o s test management	Finance Tracker 🧧 🗮	२ ४ 0 ४ ≗ ४
📝 Define	Markus Meyer		🖨 😋 ᢒ
😤 Plan	Properties Project Roles Jobs Results Changes	Save	Discard Back
🏟 Execute	은 🛍 0 Entries - Page 1 of 1 🔰 🔺 🕨	10 🗸	▽ & =
🕒 Evaluate	No entries available Description Description Description		
🄑 Configure	Created 4 years ago by Felix Mustermann		
	Assign Project Role The user will be assigned to the following projects with the role selected below.		Discard Back
	1 Entries - Page 1 of 1 🕅 🚽 1 🕨 🕅 10 🗸	7 & ≡	
	Description		
	P00003 Printer Tester		
	Add users as Tester V		
		OK Cancel	

Figure 10.10. The "Assign Project Roles" Dialog

10.3.2.4. Jobs

The *Jobs* tab lists all the jobs assigned to the user. See <u>Section 7.1, "Jobs"</u> for more information on how to manage jobs.

10.3.2.5. Results

The *Results* tab shows all the test results of the tests that the user has executed, see <u>Section 5.2.3.2.7, "Test Runs and Results"</u>.

10.3.2.6. Changes

In the *Changes* tab the change history for the user can be found, see <u>Section 5.2.3.2.8, "Change History"</u>.

10.4. System

This page manages the settings for the behavior and appearance of the application. It contains five tabs, *Miscellaneous*, *Notifications*, *Test Execution*, *Interface* and *Languages*.

10.4.1. Miscellaneous

≡ 📌 K L A R	o s test management	Finance Tracker 🗃 📑	۹ × 0 × ≛×
📝 Define	System		
📇 Plan	Miscellaneous Notifications Test Execution Interface Language	S	
🏟 Execute	Application URL	http://infosys.verit.de/klaros-prime	0
🗘 Evaluate	Maximum Upload File Size [Byte]	10485760	
f Orafauna	Use the default container session timeout		
Configure	Session Timeout (Minutes)	30	
	Only administrators may create users		
	Only managers and administrators can see other users in the user management		
	Only managers are allowed to change a job status		
	Only show approved report templates in the evaluate section		
	Hide predefined reports		
	Search Index	Rebuild	
			Save Discard

Figure 10.11. The "Miscellaneous" Tab

The following settings can be made here to define the behavior of the application:

0000000	Session Timeo	out			
	A changed session ti	meout does not affect the current session.			
Application URL		If the application should be accessed via a proxy server, the configured URL must be specified here.			
		For detailed information on configuring a proxy, see https://httpd.apache.org/docs/2. 4/mod/mod_proxy.html (Apache), https://docs.nginx. com/nginx/admin-guide/web-server/reverse-proxy/ (NG- inx) https://docs.microsoft.com/en-us/iis/extensions/config- uring-application-request-routing-arr/creating-a- forward-proxy-using-application-request-routing (IIS). Clicking the 😰 will verify the specified connection.			
Maximum [Byte]	Upload File Size	The maximum size for attachments to be uploaded.			
Use the default container ses- sion timeout Session Timeout (Minutes)		If checked, the default tomcat session timeout will be used (30 minutes).			
		If the checkbox above is unchecked, this value is used as the session timeout.			
Only Admi Users	nistrators can create	If checked, only administrators can create users. If unchecked, managers can also create users.			

Only Managers and Administra- tors can see other users in the user management	If enabled, users in the <i>Tester</i> and <i>Guest</i> roles will only see their own user entry. Entries from other users will not be displayed.
Only managers are allowed to change a job status:	If checked, only users in the role of <i>Manager</i> or higher may change the status of jobs.
Only show approved report templates in the evaluate sec- tion	If enabled, only reports with the status <i>Approved</i> will be dis- played in the <i>Configurable Reports</i> table on the <i>Evaluate / Re-</i> <i>ports</i> page.
Hide predefined reports	If enabled, the display of the <i>Reports</i> table is suppressed on the <i>Evaluate / Reports</i> page.
Search index	If problems should occur with the general search, a re-index- ing of the database can be started here using the Rebuild button.

10.4.2. Notifications



For various events, such as the creation of a user account or the assignment of a task, notifications are sent by the system via e-mail. In the *Notifications* tab, the corresponding conditions are set.

Multiple notification settings can be defined in a notification scheme and reused for multiple projects. When a notification scheme is declared as the default notification scheme, it is automatically assigned to each existing project unless another scheme has already been defined for that project.



Email-Configuration Required!

For notifications to work properly, you must configure email server settings as described in <u>Section 10.5.3, "E-Mail"</u> and ensure that all user accounts have a valid email address.

10.4.2.1. Overview Page

The overview page displays all existing notification schemes in a table. New notification schemes are created here.

😑 👱 KLARC) S TEST MANAGE	MENT			Finance Tracker 🗧 🚆			Q X	0 ~ ≗ ~
📝 Define	System								
📇 Plan	Miscellaneous Notifications	Test Execution	Interface Lan	guages					
-	Notification schemes								
😰 Execute			23 Entries - Page	1 of 3 🕅 ┥	1 2 3 🕨 🗎 10 🗸				7 & ≡
	ID≑	Name 🗢			Description 🗢	Default	Events 🖨	Projects 🖨	Action
🕒 Evaluate	NS01765 test						0	0	CDÛ
	NS01763 test						0	0	₡₽₫
🄑 Configure	NS01760 test						0	0	₡₽₫
•	NS01756 test						0	0	₡₽₫
	NS01754 test						0	0	₡₽₫
	NS01751 test						0	0	☞ @ @
	NS01264 test			dis			5	0	₡₽₫
	NS00779 SupportAutoReply						0	0	◪▯
	NS00778 SupportAutoReply						5	0	C D ti
	NS00777 SupportAutoReply						0	0	┏╻๓
	ID	Name			Description	Default	Events	Projects	Action
	New						Save	e	Discard

Figure 10.12. The "Notification Schemes" Page

The table shows the following values:

ID	Assigned automatically.
Name	The name of the notification scheme.
Description	The description of the notification scheme.
Default	When enabled, the notification scheme is the default scheme. If a notification scheme is declared as the default scheme, it is automatically assigned to any existing project, unless another scheme has already been defined for that project.
Events	The number of notification events.
Projects	The number of projects to which this notification scheme is assigned.
Actions	The executable actions.

10.4.2.1.1. Actions

The action column is located on the far right of the table. The following actions can be performed here:

🕜 Edit

Duplicate

🗊 Delete

10.4.2.1.2. Table Operations

The following operations can be performed in the line above the table on the right:

√ Filter / Sort
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

10.4.2.1.3. Details Page

Figure 10.13 shows the edit options for a notification scheme.

. ≡ 👱 K L A R	o s test management	Finance Tracker	ι≡ Qγ ≛γ
📝 Define	System		
📇 Plan	Miscellaneous Notifications Test Execution Interfac	e Languages	
🔹 Execute	Scheme: NS01892 - SupportAutoReply	Target	Action
욙 Evaluate	Account Created Account Password Changed		+
🔑 Configure	Job Assigned		+
	Job ready to Add notification Test executic Event	Target	+
	Created less Account Created Account Created	O Assignee	changed less than a minute ago by Sandra Selen
	Assig Job Assigned Job ready for execution	Current User Specific User Claudia Könnnecke	re Discard Back
	rest execution failed	Add	Cancel

Figure 10.13. The "Add notification" Dialog

This page lists all notifications for the notification scheme. To create a notification, events from a predefined set of available notification events can be individually assigned to one or more recipients (user accounts).

The available notification events are:

Job Assigned	This event is triggered once a Job is saved with a new as- signee.
Job ready for execution	This event is triggered whenever a non-executable job be- comes executable, or all dependencies of the job are fulfilled (See <u>Section 7.1.2.5, "Dependencies"</u>).
Account Created	This event is triggered once a user account has been created.
Account Password Changed	This event is triggered once an account password has been changed in the local user database.
Test execution failed	This event is triggered after completing a test case execution if a test case step was marked as failed or error during execu- tion.

The available notification targets are:

Assignee	The assignee of the job/test case or the created/edited user account.
Creator	The user responsible for triggering the notification event, e.g. by assigning a job.
Current User	The user currently active.
Specific User	A specific user.

By clicking the Assign projects button, the notification scheme can be assigned to one or more projects (see Figure 10.14).

≡ 📌 K L A R	o s test managem			Finance Track	ker 🖻 📑	Q X	0~ * ~
📝 Define	System						
砦 Plan	Miscellaneous Notifications	Assign Projects to Notification Scheme		Assisted			
 Execute Evaluate Configure 	Scheme: NS01765 - test Event Account Created Account Password Changed Job Assigned Job ready for execution	Available P00011 P000010 - Test PRV P000009 - PRV 2 P00008 - PRV P00007 - Issue Management Int P00000 Filoman Tacklor	>	Assigned			Action + + + + + +
	Test execution failed Created 13 days ago by selen2 selen2	P00003 - DE Finanz - Iracker P00003 - Printer Tester P00002 - Finance Tracker P00001 - FinanceTrackerLocalT	* *	-	L	ast changed 13 days ago.	+ by selen2 selen2
	Assign Projects			Close	Save	Discard	Back

Figure 10.14. Assigning Projects to a Notification Scheme

10.4.3. Test Execution

≡ 👱 К L A R	0 s test management	Finance Tracker 📓 🗮	Q × 0 × ≗×
🗹 Define	System		
📇 Plan	Miscellaneous Notifications Test Execution Interface Languages		
🕸 Execute	Automatically create new test cases for unknown tests when importing test case results Create additional test suite results and a corresponding test suite if necessary when importing test case results Testers may only execute tests via a related job	✓ ダ	
🕒 Evaluate	Testers may only resume their own test runs Testers are required to add a test sten result summary		
🌽 Configure			
	Explanatory templates for skipped test results		
	0 Entries - Page 1 of 1 🔣 🚽 🕨 🗎	10 🗸	
	No entries available		relet
	Reason		Action
	New		
			Save Discard

Figure 10.15. The "Test Execution" Tab

In this view, the following settings regarding test execution can be made:

Automatically create new test cases for unknown tests when importing test case results	If selected, a test case is automatically created for the respec- tive test case information in the result file. This information may vary with the import format.
Create additional test suite re- sults and a corresponding test suite if necessary when import- ing test case results	If selected, a test suite result is automatically created for the respective test suite information in the result file. This information may vary with the import format. In addition, a corresponding test-suite is created for the test-suite result, if it doesn't already exist.
Testers may only execute tests via related jobs	If selected, users in the role of <i>Tester</i> may only start tests by executing a job.
Testers may only resume their own test runs	If enabled, users in the <i>Tester</i> role can only resume test runs, that they have started themselves. If a test run was started via a job, they can only resume the test run only if the job is assigned to them.

10.4.3.1. Explanatory templates for skipped test results

If testers want to skip a test case during manual test execution, the system asks for a justification for this. This table contains the templates for these reasons (see <u>Section 8.3.3.1.1, "Skip Test</u> <u>Cases Permanently</u>"). These templates can be applied when skipping.

10.4.3.1.1. Actions

The following actions can be performed in the action column:

🕜 Edit

前 Delete

≡ 👱 K L A R	0 S TEST MANAGEMENT Finance Tracker	≡ Q× £ ×
🕑 Define	System	
📇 Plan	Miscellaneous Notifications Test Execution Interface Languages	
Execute	Automatically create new test cases for unknown tests when importing test case results Create additional test suite results and a corresponding test suite if necessary when importing test case results Testers may only execute tests via a related job Testers may only resume their own test runs	
	Testers are required to add a test step result summary	
Configure	Explanatory templates for skipped test results Too Entries-Page 1 of 70 N 1 2 3 4 5 6 7 8 9 10 N	
	Reason Server is unreachable	
	Back	C
		යි බ රැ බ
		ピー



10.4.4. Interface

The Interface tab defines the behavior of the user interface.

≡ 坐 K L A R	d s test management	Finance Tracker 🗧 🗮	५ × 0 × ≜ ×
📝 Define	System		
📇 Plan	Miscellaneous Notifications Test Execution Interface L	anguages	
🗱 Execute	Test Runner starts in tabular view Show test case information during test execution Test step editing starts in the tabbed view mode		
🕒 Evaluate	Rows per table page 10 V		
🌽 Configure	Maximum preview image height 72 Maximum preview image width 320		
	Quote of the day		
	Use random quote of the day		
		Name 🗢	Size Action
	quotes.txt Upload Quotes		24 КВ Ш
			Save Discard

Figure 10.17. The "Interface" Tab

The following settings can be configured here:

Test Runner starts in tabular view	If enabled, the test runner will per default show the tabular view.
If enabled, show test case in- formation during test execution	If enabled, the test case detail information panel is always ex- panded per default when launching or executing tests.
Test step editing starts in the tabbed view mode	If enabled, the default view in the <i>Steps</i> tab in the <i>Test Case Details</i> page will be the tabular view.
Rows per table page	This option presets the number of rows per table page.
Maximum preview image height	This option sets the maximum height of the preview images for embedded attachments during test case execution (see <u>Figure 6.45, "An Attachment Reference Replaced by a Preview</u> <u>of the Attachment"</u>).
Maximum preview image width	This option sets the maximum width of the preview images for embedded attachments during test case execution (see <u>Fig-</u> <u>ure 6.45, "An Attachment Reference Replaced by a Preview of</u> <u>the Attachment"</u>).
Show version number in footer	If enabled, the version number of the application is displayed in the footer.
Quote of the day	A message to be displayed on the login screen.

Use random quote of the day	When selected, a random line from the file uploaded below will be displayed as the quote of the day.
Upload Quotes	By clicking the Upload Quotes button, a new text file with quotes can be selected and uploaded. The file must contain one quote per line. The uploaded file can be removed and replaced by the supplied file of the application using the min icon.
	Please refer to <u>Section 4.2, "Quote of the Day"</u> for detailed in- formation on the file format.

10.4.5. Languages

≡ 🎐 KLAROS TEST MANAGEMEN Finance Tracker 👕 📑 द × 0/ ▲ × System 📝 Define 📇 Plan Miscellaneous Notifications Test Execution Interface Languages abled Lang Execute 🚾 English (en_US) 🚎 German (de_DE) 🕒 Evaluate 🔑 Configure > **>>** < « Refresh messages files

Additional languages can be added to the user interface using language files.

Figure 10.18. The "Languages" Tab

10.4.5.1. Enabling and Disabling Languages

Up to five different languages can be activated at the same time. Clicking the > icon adds the selected language, clicking the < icon removes it. To add all available languages, click on the » or on « to remove all languages or use the drag and drop function.



File Format

Please refer to <u>Section 4.1, "Languages"</u> for detailed information on the file format.

Klaros-Testmanagement comes with language files in German and English. Support for additional languages can be added by using additional language files if needed. The corresponding language file must be stored in the .klaros/resources/messages directory for this.

Clicking the Refresh messages files button reloads the language files in the .klaros/resources/ messages directory while the application is running and displays them in this view.

10.5. Integration

On the *Integration* page, external systems are connected and their settings are managed. It features six tabs, *Issue Management*, *Requirements Management*, *E-Mail*, *Network*, *LDAP* and *CAS*.

10.5.1. Issue Management

An issue management system (also called issue tracker or bug tracker) is an application for managing issues that occur during the testing process. Issues can contain information and reports about errors in a software system.

Issues can be created retrieved and assigned to failed test case results. It is possible to configure and use multiple issue management systems simultaneously.

Klaros-Testmanagement supports the following issue management systems:

Bugzilla, a free open source issue management system (see http://www.bugzilla.org/).

GitHub, a web-based Git repository hosting service offering both plans for private repositories or free accounts, which are usually used to host open-source software projects (see http://github.com/)

GitLab, a web-based DevOps lifecycle tool that provides a Git-repository manager providing wiki, issue-tracking and CI/CD pipeline features, using an open-source license (see http://gitlab.com/).

Jira, is a commercial issue management system produced by Atlassian Pty Ltd. (see http://www.atlassian.com/).

Mantis, a free open source issue management system (see http://www.mantisbt.org/).

Redmine, a free open source issue management system (see http://www.redmine.org/).

Trac, a free open source issue management system (see http://trac.edgewall.org/).

YouTrack, is a commercial issue management system produced by JetBrains (see https://www.jetbrains.com/youtrack/).



External Configuration Required

Some issue management systems need to be configured before they can be successfully connected to Klaros-Testmanagement. For more information, see <u>Section 3.14</u>, <u>"Configuring External Issue-Management-Systems"</u>.



Token based authentication

Some issue management systems like Jira, GitHub, GitLab, Redmine and YouTrack require access via token-based authentication. Authentication via username and password is not supported in this case.

Klaros-Testmanagement is supporting a token based authentication scheme as well. Just use the required username and the token as password.

= 📌 K L A R) s test managemi	ENT		Finance Tra	acker 🖬 🛢	Q ;	× @~ ≛~
🕑 Define	Integration						
🏩 Plan	Issue Management (8) Requirem	nents Management (1) E-Mail	Network LDAP C	AS			
🔹 Execute	C ² D □ ID ≑ Ω System ≑ Ex	8 Entries - Pa	ge1of1 KI ◀ 1	▶ N 10 ▼	Projects ≑	Ssues ≑	Show all 🕹 🗮
🕒 Evaluate	IM00017 A Redmine Pla	ayground http://dev-redm ICS http://infosys.ve	ine.verit.de/ erit.de/jira6			2	C © C © C © C ©
🖉 Configure	☐ IM00015 ▲ 🙀 Jira (legacy) ☐ IM00014 🐳 Mantis MA	http://infosys/ji ANTISTEST http://dev-mant	ra/ is2.verit.de		Issue Management Integration	0	C C C D C C C D
	IM00013 👩 GitHub	https://github.c	om/susp248/ktmtest		Issue Management Integration	2	ピピピ₪
	🗌 IM00012 🛛 🖊 GitLab	https://dev- gitlab.verit.de/n	oot/awesomeproject	Gitlab - test project	Issue Management Integration PRV PRV 2 Test PRV	4	2020
	🗌 IM00010 🋛 🦳 Redmine fina	ancetracker http://dev-redm	ine40.verit.de/			0	ľÌľ
	🗌 IM00008 🦉 Jira (legacy) PL	AYGROUND http://infosys/ji	ra/	Klaros	DE Finanz-Tracker Finance Tracker FinanceTrackerLocalTester	36	ſĊſ
	ID System E	ixternal Project ID	URL	Description	Projects	Issues	Action
	New					Save	Discard

Figure 10.19. The "Issue Management" Tab

The table shows the following values:

ID	Assigned automatically.
Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
	$\underline{\Lambda}$ no credentials for background synchronization has been provided.
System	The used issue management system (Bugzilla, Jira (Version 4.3 or higher), Jira Legacy (Version 4.2 or lower), GitHub, Git-Lab, Mantis, Redmine and Trac).
URL	The link to the issue management system.
External Project ID	Die Projekt-ID des Issue-Management-Systems. See <u>Sec-</u> tion 10.5.1.1, "External Project ID".
Description	Additional information on the issue management system.
Projects	The projects in which the issue management system is used.
Actions	The executable actions.

10.5.1.1. External Project ID

If the issue management system organizes issues in projects, the project ID is required to address the specific project. Jira, Redmine, Mantis, Bugzilla and YouTrack use such a project id. With GitHub, GitLab, and Trac, the URL is used to address different projects.

The valid project ID values can be found in the issue management systems at the following locations:

10.5.1.1.1. Bugzilla

The Bugzilla project id consists of the Product field value.



Figure 10.20. The Bugzilla Project ID

10.5.1.1.2. Jira

The Jira project id consists of the Key field value.



Figure 10.21. The Jira Project ID

10.5.1.1.3. Mantis

The Mantis project id consists of the Project Name field value.





10.5.1.1.4. Redmine

The Redmine project id consists of the Identifier field value.

Home My page Projects Adminis	stration Help
MvAwesomeProi e	ect
,	
+ Overview Activity	Issues Spent time Gantt Calendar News Documents Wiki Files Settings
Settings	
Pariat Marchana Tarana	
Project Members Issue	tracking versions issue categories Repositories Forums Time tracking
Name *	MyAwesomeProject
Description	Fdit Preview B / U + C HI H2 H3 := := := := := := @
Identifier *	awesome
Homepage	
Public	: 🖉
Tabasit washes	Public projects and their contents are available to all logged-in users.
Innerit members	

Figure 10.23. The Redmine Project ID

10.5.1.1.5. YouTrack

The YouTrack project id consists of the ID field value.

		Configure
/	YouTrack	Issues 🗸 Dashboards Agile Boards Reports Projects More 🗸 🖒 🕐 💽 🗬
	Projects > Den	no project > Edit Project Clone project Archive project Delete project
	Settings Access	Team Fields VCS Notifications Build Server Integration Time Tracking Workflow
	Name ID	Demo project DEMO Will be used as issue ID prefix
	Project owner	Administrator (root) v 3
	Logo	DEM Reset to default Upload a JPG, GIF, or PNG file. The image is resized to 48 × 48 pixels automatically
	Custom project template	
	Default visibility setting	Select an option
	Recommended visibility options	Project Default - Reduces the available options for restricting visibility in issues and articles to selected groups and teams
	Description	Normal text v B I S A v 77 S I E E I Visual Markdown Aa 🕜

Figure 10.24. The YouTrack Project ID

10.5.1.2. Adding a new Issue-Management-System

By clicking on the button New a new empty table row will appear. Attributes of the issue management system can be defined here.

With Save the issue management system saved. The ID of the issue management system (IM00001) is automatically assigned by Klaros-Testmanagement.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



10.5.1.3. Actions

The following actions can be performed in the action column:

Open external link to the issue Management System

- Validate the URL (see Section 10.5.1.5, "Editing an existing Issue-Management-System")
- C Edit (see Section 10.5.1.6, "Editing an existing Issue-Management-System")
- 🗊 Delete
- 10.5.1.4. Table Operationen

The following operations can be performed in the line above the table on the right:

√ Filter / Sort

Only active / Show all

- 凸, Export
- \equiv Column selection

All operations are described in detail in Section 5.2.3.1, "Overview Page".

10.5.1.5. Editing an existing Issue-Management-System

In the *Action* column, click the icon to check whether a connection to the issue management system can be established.



Trailing Slashes of URLs

In case of an authentication error, first check whether the addition or deletion of a trailing slash to the URL resolves the problem.

10.5.1.6. Editing an existing Issue-Management-System

Each issue management system created has its own detail page with further subviews. By clicking on the ID of the respective issue management system or on the icon \square on the right side of the action column you will get to the last selected detail view. When called for the first time, this is the Properties tab.

The following tabs are available: Properties, Projects, Attachments and Changes.

10.5.1.7. Actions

On the detail pages there are more icons in the upper right corner. The following actions can be performed here:

ľ	Open external link	Shows the request in the external source.
G	Browse	Use the green arrows at the very top right to switch between the issue managements systems present on the previous page.

10.5.1.8. Properties

≡	👱 KLAR	OS TES	T MANAGEMENT	Finance Tracker 🛅 🗮		Q :	× 0° *
ľ	Define	IM00017 -	Redmine, Playground				299
***	Plan	Properties	Attachments Projects Issues (2) Changes		Save	Discard	Back
\$	Execute	System	Redmine 🗸				
	Evoluato	URL	http://dev-redmine.verit.de/				
•	Evaluate	Description	Playground				
بر	Configure	Authentic	ation				
		To Th	periodically refresh the remote issues please enter credentials for background synchronization. e user entered here must have the administrator role				×
		Username	admin				
		Password					
		Te	st Authentication				
		State Valu	les				
		Th Th	e state values listed in the following table correspond to a resolved issue. e specified value must exactly match the status name in the connected system.				×
			Name				Action
		Closed					Û
		Rejected					
		Resolved					Ш
		New					Default
		Created 3 yea	rs ago by Felix Mustermann		Last ch	anged 3 years ago by	y Felix Mustermann
					Save	Discard	Back

Figure 10.25. The "Properties" Tab

The *Properties* view is divided into three sections: Issue Management System, Authentication, Status Names

Issue-Management-System

System	A drop-down menu can be used to select the desired system.		
URL	The URL where the system can be reached.		
Project	The name of the project.		
Description	An optional description of the system.		

Below this is the "Authentication" section. Here you enter the login information for the background synchronization. The user entered here must have full access to the entered project.

Username	The valid username.
Password	The valid password.

The Test login button can be used to check whether the entries are correct.



Important

The credentials entered here will solely be used for background synchronization. If users creates or updates an issue, their personal credentials will be used to authenticate against the issue management system. If credentials are missing or not valid, then users are prompted to enter them. The last section is "Status names". The status names listed in the following table correspond to a completed issue. The specified value must exactly match the status name in the connected system.

≡ 🞐 к L A R	o s test management	Finance Tracker 🗧 블	० 0 ४ ≗ ४
Define	IM00015 - Jira (legacy)		🖸 😋 🗲
_		Save Discard	Back
📇 Plan	Properties Attachments Projects Changes		
🔹 Execute	System Jira (legacy) 🗸		
	URL http://infosys/jira/		
🕒 Evaluate	Project		
	Description		
🔑 Configure	Authentication		
	To periodically refresh the remote issues please enter credentials for background synchronization. The user entered here must have full access to the remote project.		×
	Username		
	Password		
	Test Login		
	State Values		
	The state values listed in the following table correspond to a resolved issue. The specified value must exactly match the status name in the connected system.		×
	Name		Action
	Closed		Û
	Done		Û
	Resolved		Û
	New		Default
		Save Discard	Back

Figure 10.26. Edit the status mappings

The New button can be used to add a new status name.

The Default button resets the fields to the default values.

All entries are transferred to the database with the <u>Save</u> button. The <u>Discard</u> button deletes the entries. The button <u>Back</u>. takes you to the overview page with the created issue management systems.

10.5.1.9. Attachments

You can add any files as attachments to an issue management System. For more information, see <u>Section 5.2.3.2.6, "Attachments"</u>.

10.5.1.10. Projects

The *Projects* view displays all projects to which the Issue-Management-System has been assigned.

10.5.1.11. Changes

The tab Changes shows the change history of this issue management system.

For a detailed description of the Changes view, see Section 5.2.3.2.8, "Change History".

10.5.1.12. Assignment of status values for completed issues

Each issue has a status field that documents its current state (e.g. "In progress" or "Fixed"). The individual statuses and their naming vary according to the issue management system used. Klaros-Testmanagement only distinguishes between open and resolved issues. An issue is considered closed when no further work needs to be done on it (e.g. "Closed", "Rejected"), otherwise it is still open.

Resolved issues are displayed in green, open ones in red. Section 9.6.5, "Details Page"

The resolved issue states must be named exactly as they are in the issue management system, otherwise they will not be properly identified and will be counted as open issues.

10.5.2. Requirements Management

Requirements can be managed either locally in Klaros-Testmanagement or in a remote system.

Local Requirements Manage-	By default, test managers can create, edit, and delete require-
ment	ments for a project from within the project.
Remote Requirements Manage- ment	A project can also receive its requirements from an external requirement management system (RMS). If this is configured, the creation, editing and deletion of requirements can only be done via the connected RMS.

Currently Klaros-Testmanagement supports the following external requirement management systems:

Jira 4.3 or newer

A commercial issue management system produced by Atlassian Pty Ltd., see http://www.atlassian.com/.

≡ 👱 K L A R	o s test management		Finance Tracker 🔤 📑	५ × 0∕ × ×
🗭 Define	Integration			
🏩 Plan	Issue Management (8) Requirements Management (1)	E-Mail Network LDAP CAS		
🏟 Execute	1 Er	ntries-Page 1 of 1 🔰 🖣 1 🕨 🛛	Description ♦ Projects ♦	▼ Show all Action
Evaluate	RM00001 🛆 🔷 Jira http://infosys.verit.de/jira6	SYNC	Printer Tester	4 ៥២ <i>೯៥</i> ២
	ID System URL	External Project ID	Description Projects	Requirements Action
🔑 Configure	New			Save Discard



The table shows the following values:

Assigned automatically.

ID

Additional Information	A tooltip appears when the cursor is placed over the icon shown here.
	合 the requirement management system be deleted because it is connected with at least one project.
	$\underline{\Lambda}$ no credentials for background synchronization has been provided.
System	The used requirements management system (Jira Version 4.3 or higher).
URL	The link to the requirement management system.
External Project ID	See Section 10.5.1.1, "External Project ID".
Description	A short description of the requirement management system.
Projects	The projects in which the issue management system is used.
Actions	The executable actions.

10.5.2.1. Adding a new Requirements Management System

By clicking on the button New a new empty table row will appear. All attributes of the requirement management system can be defined here.

With <u>Save</u> the requirement management system is saved. The ID of the requirement management system (RM00001) is automatically assigned by Klaros-Testmanagement.

With New several table rows can be created and edited at the same time. Only when clicking on Save the data is stored in the database.

With Discard all changes are undone.



Red IDs

All rows with red IDs have been changed and are not yet saved!

10.5.2.2. Actions

The following actions can be performed in the action column:

- Solution of the URL (see Section 10.5.1.5, "Editing an existing Issue-Management-System")
- C Refresh requirements
- C Edit (see Section 10.5.2.3, "Editing an existing Requirements Management System")
- 🗊 Deactivate

Synchronizing Attachments

When connected to a Jira server, attachments from its requirements are automatically transferred to Klaros-Testmanagement during synchronization.

10.5.2.3. Editing an existing Requirements Management System

Clicking on the \square icon in the action column on the right opens a dialog. The credentials for the background synchronization can be provided here.

10.5.3. E-Mail

Username

In the *Email* tab the e-mail server settings of the application are configured. These settings are required for sending notification e-mails (see <u>Section 10.4.2, "Notifications</u>"). The attributes that can be edited are shown in <u>Figure 10.28</u>.

≡	Y KLAR	os test M <i>i</i>	A N A G E M E N T			Finance Tracker 👕 🗮	Q x Ø~ ≛~
ľ	Define	Integration					
**	Plan	Issue Management	(8) Requirements Management (1)	E-Mail Network	LDAP CAS		
\$		SMTP Server SMTP-Server-Port	localhost 25				
¢	Evaluate	Sender Address Authentication	No Authentication 🗸				
٦	Configure	Security	None sou				
			SSL TLS				
			E Send rest Man				Save Discard

Figure 10.28. The "E-Mail" Tab



The username needed to authenticate against the SMTP server. This field is only present if SMTP or POP authentication is selected in the *Authentication* field below.

PasswordThe password needed to authenticate against the SMTP serv-
er. This field is only present if SMTP or POP authentication is
selected in the Authentication field below.

AuthenticationTo prevent or identify spam, some mail servers require users to
authenticate themselves before they allow them to send mails.

	This option selects whether no authentication is required, or a username / password combination via SMTP or POP is used. If one of the latter two is selected, the <i>User Name</i> and <i>Password</i> fields are shown in the user interface.
Security	This setting defines the transport layer security used to access the SMTP mail server. Each transport security setting may re- quire a different port value. The options are <i>None</i> (Port 25), <i>Se- cure Socket Layer / SSL</i> (Port 465) or <i>Transport Layer Securi-</i> <i>ty / TLS</i> aka STARTTLS (Port 25). Your local port settings may vary from the given defaults.
Send Test Mail	This link sends a test email to verify that your e-mail server settings are valid. It is therefore necessary that a valid e-mail address is specified in your user account.

Click the Send Test Mail button to test if the fields are filled in properly.

10.5.4. Network

Klaros-Testmanagement supports the use of HTTP and SOCKS proxies for external network connections. Proxy settings are configured in the *Network* section.

≡ 🞐 K L A R	os test man.				Finance Tracker 🧧 🔜	Q X	0 ∼ ≛~
📝 Define	Integration						
🏩 Plan	Issue Management (8)	Requirements Management (1)	-Mail Network	LDAP CAS			
🕸 Execute	Proxy Host Port	proxy.verit.de 3128					
🕒 Evaluate	No Proxy For Type of Proxy	HTTP O SOCKS					
🄑 Configure	Requires Authentication	🕼 Test proxy settings					
						Save	Discard

Figure 10.29. The "Network" Tab



Setting the application URL

If Klaros-Testmanagement is run behind a proxy, the application URL must be set accordingly in order for Klaros-Testmanagement functioning properly. The Application URL can be set in the *Miscellaneous* tab of *General Settings* (see <u>Section 10.4.1, "Miscellaneous"</u>).

The properties of the proxy settings are:

Proxy Host

The network address or hostname of the proxy server.

Port	The port that the proxy server is active on.
No Proxy for	A -separated list of domains which should bypass the proxy settings.
Type of Proxy	Possible settings are HTTP proxy or SOCKS proxy.
Requires Authentication	Specifies whether the proxy requires authentication or not.
Username	The username to use for the username/password authentica- tion.
Password	The password to use for the username/password authentica- tion.

The proxy settings can be tested by clicking the Test proxy settings button.

10.5.5. LDAP



Only available in Klaros-Testmanagement Enterprise Edition



Тір

To access an LDAP or Active Directory server, it is necessary to set numerous configuration parameters first. When in doubt, please consult your system administrator about the values to enter here.

≡ 🞐 KLAROS TEST MANAGEMENT			Finance	Tracker 🖻 🗮	۹×07 ±۰
📝 Define	Integration				
🎥 Plan	Issue Management (8)	Requirements Management (1) E-Mail N	Vetwork LDAP CAS		
 Execute Evaluate Configure 	Server Address Server Port Secure (LDAPS) Bind DN Bind Credentials Follow Referrals	Idap1.verit.de 389 cn=Manager,dc=verit,dc=de	User Context DN User Object Classes Enable naive DN Matching Mode User DN Prefix User DN Suffix	ou=Users,dc=verit,dc=de person,posixAccount uid= ,ou=Users,dc=verit,dc=de	
	User Search Attribute User Name Attribute User Password Attribute	uid uid userPassword	Full Name Attribute Email Attribute Enabled Attribute	cn mail	
	Set as default	ⓒ Test LDAP access	Disable automatic user registration Disable Password Synchronization		Save Discard



Parameters required to contact the LDAP server:

Server Address	The URI under which the LDAP server resides (e.g. ldap.acme. com).
Server Port	The port on which the LDAP server is located (typically 389 for unencrypted connections or 686 for LDAPS).

Secure (LDAPS)	If this option is enabled, the encrypting LDAPS protocol is used.
Bind DN	The Distinguished Name of the user account that will log on to the server and execute the bind operation.
Bind Credentials	The password credential of the user account that will log on to the server and execute the bind operation.
Follow Referrals	If this option is enabled, searching a directory automatically follows any referrals the server might return. When disabled referrals will be ignored and other servers will not be contacted during the search.
Parameters needed to locate a use	er account:
User Context DN	The distinguished name of the path under which user ac- counts will be searched (e.g. ou=Users,dc=verit,dc=de).
User Object Classes	A comma separated list of the LDAP object classes a user account must match to be included in the search (e.g. person,posixAccount).
Enable naive DN Matching Mode	If this option is enabled, the returned DN of a user search is directly used to authenticate the user. If this option is disabled, the following two parameters will be used in conjunction with the <i>User Name Attribute</i> to build a DN to authenticate with.
User DN Prefix	The prefix of the distinguished name used to locate user ac- counts (e.g. uid=).
User DN Suffix	The suffix of the distinguished name used to locate the user account (e.g. ,ou=Users,dc=acme,dc=com). When locating user accounts, the prefix, the account id and the suffix are concatenated to form the distinguished name of the user account.
Parameters describing the attribut	es of a user account:
User Search Attribute	The LDAP username attribute which corresponds to the Klaros-Testmanagement account name (e.g. uid).
User Name Attribute	The LDAP attribute which will be used in the DN bind action which authenticates the user (if naive DN Matching mode is deactivated). In a simple scenario this will match the <i>User</i> <i>Search Attribute</i> .
	If your LDAP Server setup does not allow you to bind a user with the specified user search attribute you should specify the corresponding attribute here (e.g. cn) and use this in conjunc- tion with the corresponding user DN prefix/suffix.

User Password Attribute The LDAP password attribute which corresponds to the Klaros-Testmanagement account password (e.g. user-Password).

Full Name Attribute	The LDAP attribute containing the full name of the user (e.g cn). If specified, this will be automatically be transferred into the Klaros-Testmanagement database upon first successful login.
Email Attribute	The LDAP attribute containing the email address of the user account (e.g. mail). If specified, this will automatically be transferred into the Klaros-Testmanagement database upon the first successful login.
Enabled Attribute	If specified, this LDAP boolean attribute defines whether a user is allowed to log in. Not all directory servers provide such an attribute.

If the Set as default checkmark is activated, the login screen will default to LDAP authentication for all users. It is still possible for existing users to authenticate against the Klaros-Testmanagement user database if selected in the login screen.

Use the *Disable automatic user registration* option to prevent the automatic creation of a user in Klaros-Testmanagement upon the first successful login.

Upon the first successful login a matching password hash is created in the local user database so that users can also authenticate themselves against the local user database with their LDAP password. If the *Disable Password Synchronization* checkmark is activated, this synchronization will not be performed and users will only be able to log in locally when an administrator assigns them a local password interactively.

Clicking the *Test LDAP access* button tests whether the parameters entered on this page are correct. The test process in divided in two phases:

In the first phase, an attempt is made to log on to the server and a search for all available users is performed. If this is successful, a dialog is displayed listing the users found in the LDAP directory.

In the second phase, a username and password can be entered to test the LDAP login of a user. The result of the login attempt is logged in the log panel.

= 👱 K L A R (OS TEST MAN	AGEMENT	Finance	
🕑 Define	13:48:07 The LDAP set	earch has been successful, 34 users found	G The LDAP search has been su	+ X
📇 Plan	Integration			
🏟 Execute	Issue Management (8)	Requirements Management (1) E-Mail Netwo	ork LDAP CAS	au-Lleare do-varit do-da
🕒 Evaluate	Server Port	389	User Object Classes	person,posixAccount
🔑 Configure	Bind DN	cn=Manager,dc=verit,dc=de	User DN Prefix	uld=
	Follow Referrals	Authentication required for LDAP		,ou=Users,ac=vent,ac=ae
	User Search Attribute User Name Attribute User Password Attribute	uid 34 Entries found stolp uid Please enter a valid user name and password userPa server. server.	to validate the authentication at the LDAP	2n nail
	Set as default	Username selen Password ·····		
		Thurebon and a	OK Cancel	1

Figure 10.31. The "LDAP Authentication" Dialog

10.5.6. CAS

Only available in Klaros-Testmanagement Enterprise Edition

Klaros-Testmanagement supports Single sign-on Authentication (SSO) using a Central Authentication Service (CAS).

≡ 📌 KLAR	o s test managem	ENT	Finance Tracker 🍯 📑	२ × 0 × ≛×
🕑 Define	Integration			
📇 Plan	Issue Management (8) Requirer	nents Management (1) E-Mail Network LDAP CAS		
🏟 Execute	CAS Server URL Prefix CAS Server Login URL	https://192.168.56.2/cas https://192.168.56.2/cas/login		
🕒 Evaluate	CAS Server Logout URL Disable automatic user registration	https://192.168.56.2/logout		
🔑 Configure	Enable CAS authentication			
				Save Discard

Figure 10.32. The "CAS" Tab

Parameters needed to contact the CAS server:

CAS Server URL Prefix	The URL under which the CAS server resides (e.g. cas.acme. com).
CAS Server Login URL	The URL of the CAS server for single sign-on logins. A user that has yet not been authenticated by CAS will be redirected to this URL.
CAS Server Logout URL	The URL of the CAS server for single sign-on logout. When a user logs out from Klaros-Testmanagement he is redirected to this URL.
Disable automatic user regis- tration	When activated, a successful CAS authentication does not au- tomatically create a guest user account in the system.
Enable CAS authentication	When activated, all authentication is delegated to the CAS Server and the default login window is no longer reachable. To complete the activation, a reboot of the application server is required.



CAS Single Sign-on Disables Local Authentication

Once CAS support is activated and Klaros-Testmanagement has been rebooted it is no longer possible to authenticate using the Klaros-Testmanagement login screen.

Please double-check that the account verification works before activating. In case of a broken setup please set the property cas.enabled in the configuration file %KL-AROS_HOME%/.klaros/klaros.properties to false and reboot the application server. This restores the default behavior.

10.6. Backup

On the *Backup* page, complete projects can be backed up and restored in XML format. Only administrators and managers are authorized to import or export projects.

10.6.1. Export

In this tab one or more projects can be selected and exported by clicking the Export button (Figure 10.33).

≡ 👱 K L A I	ROS TEST MANAGEMENT			Finance Track	er 🖻 🛢		Q X	0 ~ ≛ ~
🕑 Define	Backup							
📇 Plan	Export Import							
📥 Evenue	9 of 9 selected	9 Entries - Page 1 of 1	H 🖣 🚺 🕨	▶ 10 ✔				7 & ≡
Execute	ID≑	Description 🗢			Requirements 🗢	Test Cases 🖨	Test Suites	🗘 Test Runs 🖨
	✓ P00011				0	1	0	0
🕒 Evaluate	P00010 Test PRV				0	5	1	3
	P00009 PRV 2				0	1	0	0
🄑 Configure	✓ P00008 PRV				0	3	1	1
	✓ P00007 Issue Management Integration				0	4	0	0
	P00005 DE Finanz-Tracker				6	24	7	8800
	Poulos Printer Tester				4	1	0	0
	P00002 Finance Tracker P00001 Finance Tracker				6	24	/	338
					0	20	9	204
								Export



Clicking the Export button exports the selected projects to an XML formatted output file.



Note

The backup files are marked with the respective database version that was exported. Backup files may only be imported if the database version of the Klaros installation matches the version of the backup file.

10.6.2. Import

In this tab, projects from backup files can be imported. By clicking the Upload Backup File button, a backup file can be uploaded. After uploading, the projects that are contained in the file are displayed. One or more of these projects can be selected (<u>Figure 10.34</u>).

≡ 👱		FinanceTracker Selenide Tests 🧧 📑	৭ @৵ ≛৵
📝 Define	Backup		
📇 Plan	Export Import		
Execute	Version: 5.0 Created by Felix Mustermann on 3/10/20, 12:17 PM		×
民 Evaluate	Description \$	Requirements 🖨 Test Cases	s ≑ Test Suites ≑ Test Runs ≑
	FinanceTrackerLocalTester	6 24	7 233
🔑 Configure	FinanceTracker Selenide Tests	7 24	7 235
	орюао Васкир File		Import Discard

Figure 10.34. The "Import" Tab

Clicking the Import button imports the selected projects and clicking the Discard button cancels the restore.



Note

An import will never overwrite existing projects or other data therein.



Important

When trying to import a backup file from a previous Klaros database version, the import is rejected with a corresponding error message. You must first import this backup file with a version of Klaros-Testmanagement that matches the backup file, and then update the installation to your current version.

An export from that version is now compatible with your current release.

Chapter 11. Custom Reports

enterprise edition

Only available in Klaros-Testmanagement Enterprise Edition

Although Klaros-Testmanagement already offers a large number of predefined reports, these are often not sufficient if there are special requirements regarding content or layout. For this reason, the Klaros-Testmanagement Enterprise Edition can also be used to create your own reports. Figure 11.1, "The Report Generation Process" gives an overview of the reporting process.



Figure 11.1. The Report Generation Process

Two user groups are involved in the generation of a report: Report Designer and User.



Programming Knowledge

To create custom reports, basic knowledge of the Java or Groovy programming languages and XML-based template creation with SeamPDF or SeamExcel is required.

• The **report designer** determines the parameters of a report and defines which data from the database should be used for this purpose. It also defines the layout and content of the report template.

- The **user** selects the parameters with which the report is to be created and retrieves the report as a PDF or Excel file.
- The **layout** template defines the layout and content of the report and is defined in an XML-based description language.
- The report script extracts the data from the database and prepares it for the use of the data in the report template. The report script retrieves and formats the data and is formulated in Groovy or Java. It is provided by a class that implements the <u>Section E.2</u>, "KlarosScript Interface". This interface defines a single method called execute and uses a <u>Section E.2</u>, "KlarosScript Interface" object as an input parameter.

This chapter covers the creation of user defined report templates so it primarily of interest to report designers. <u>Section 9.2, "Reports"</u> shows how users can generate reports from them.

11.1. The Context Object

The report script provides the data for the rendering of the layout template. It retrieves the data either directly from the object model <u>de.verit.klaros.core.model</u> or from the database via the HQL query language.

The resulting data for the layout template must then be stored in the <u>Section E.2, "KlarosScript</u> <u>Interface"</u> object which has been passed to the execute method.

If parameters are defined, they are available via the context object to both the report script and the layout template.



Predefined Objects in the Context

The <u>Section E.2, "KlarosScript Interface"</u> object already contains predefined objects. For a list please refer to <u>Section E.1, "The Klaros Report Context"</u>.

11.2. Creating a Report Template

Go to the *B* Configure section and select the Report Templates page.

= 👱 K L A R	d s test management		Finance Tracker 📔 📑	۹×0° ≛۲
📝 Define	Report Templates			
📇 Plan	New			Save Discard
		7 Entries - Page 1 of 1 🕅 ┥ 🧻 🕨 🗎 15	5 🗸	√ Show all &
📩 Execute	🔲 ID 🗘 💡 🛛 Name 🗢	Description 🗢	Status 🗢 Format 🗢 Revision	n Changed By ♦ Changed ♦ Action
	RT00007 🛆 Issue Overview Report	Lists the issues in a project sorted by selectable parameters	Approved PDF 1.1	System Account 2 years ago 🛛 🗋 📿 🛄 前
	RT00006 👌 Iteration Status Report	Summary of an iteration status in a project	Approved PDF 1.1	System Account 2 years ago 🛛 🛱 📿 🗍
C Evaluate	RT00005 👌 Job Excel Overview Report	Lists the jobs in a project	Approved Excel 1.1	System Account 2 years ago 🔹 📿 🛄
	RT00004 🛆 Job Progress Report	Lists the jobs in a project sorted by selectable parameters	Approved PDF 1.1	System Account 2 years ago 🛛 🛱 📿 🛄 🗊
🌽 Configure	RT00003 👌 System under Test Status Repo	rt Lists the test case results in a SuT sorted by selectable parameters	Approved PDF 1.1	System Account 2 years ago 🛛 🗘 💭 🛍
	RT00002 👌 Test Run Overview Report	Lists the test runs in a project sorted by selectable parameters	Approved PDF 1.1	System Account 2 years ago 🛛 🛱 📿 🛄 🛍
	RT00001 🛆 Job Status Report	Lists the jobs in a project grouped by selectable parameters	Approved PDF 1.1	System Account 2 years ago 🛛 🗋 📿 🛄 🗊
	ID Name			Changed By Changed Action
	New			Save Discard



A click on the button New creates a new report template. Here you can enter a *name* and a short *description* for the report.

Name	The name of the report.
Description	A description of the report.
Output Format	The output format: PDF or Excel.
Status	The status of the report: Draft or Approved.

Klaros-Testmanagement comes with several predefined report templates These are read-only, but can be duplicated with the 🔲 icon. The duplicate can then be modified as desired.

≡ 🞐 K L A R	os test management	Finance Tracker 📓 🚆	५ × 0 × ≛ ४
📝 Define	RT00005 - Job Excel Overview Report		6 9
🏩 Plan	Properties Parameters (2) Script Layout Template Revisions Changes	Preview Save	Discard Back
🌣 Execute	Name Label Description	Туре	Default Mandatory Action
	sortJobBy Sort Job List Select the order to show the jobs	Multi List	Job ID I III
🕒 Evaluate	showFinishedJobs Show Finished If selected, the report includes closed, resolved and rejected jobs	Boolean	✓ I Ш
🄑 Configure	New Parameter		
		Preview Save	Discard Back



• The *Parameters* view shows the parameters defined for this template. For newly created reports, this list is empty.

- The *Script* tab manages the report script. Newly created reports already have a predefined class with the necessary methods.
- The layout template is managed in the *Template* view. As with the report script, a basic layout already exists for new reports.



Note

For larger report templates it can be helpful to use an IDE (e.g. Eclipse) for the script and layout template. The created files can then be uploaded in Klaros-Test-management. To avoid errors, add the Klaros model libraries to the build path of your Eclipse project. The tutorial document has a chapter about creating report templates where this is explained in detail.

Instead of editing the report script and template directly in the application, they can also be imported from a file. Specify the file to be used by clicking the Browse button and selecting the file to be imported. A click on *OK* in the file dialog and then on Upload imports the selected file into the script or template field.

This page provides three actions to be executed:

Preview generates a preview of the report.

Save stores the report template.

Discard discard all changes to the report template.

To make reports more flexible, it is possible to pass parameters to the report script. These can be used, for example, to pass a time span for which data is determined.

To add arguments to the report, click the $+$	icon button on the parameters tab.
---	------------------------------------

= 👱 K L A R (o s test manac	GEMENT		Finance Tracker 👕 🛢	<u>م</u>	.× 0× ≗×
🖍 Define	RT00002 - Test Run Over	view Report				60
砦 Plan	Properties Parameters (5) Script Layout Temp	ate Revisions Changes	Preview Save	Discard	Back
🏟 Execute	Name	Label	Description	Туре	Default	Mandatory Action
	startDate	Start Date	The start date of the test runs to consider	Date		1
🕒 Evaluate	endDate	End Date	The end date of the test runs to consider	Date		1 🗊
	sortByTestRun	Test Run Sort Criteria	Choose an entry, how the test runs should be listed.	Multi List	Test Result	1 🗊
🔑 Configure	showDetailedTestRuns	Show Test Run Details	Select if detailed test run results should be included	Boolean	\checkmark	1 🗊
<i>a</i> configure	showPassedResults	Show Passed Results	Also include passed results	Boolean	\checkmark	1 🗊
	New Parameter			Preview Save	Discard	Back



The type of the parameters can be specified by selecting the appropriate value from the type list. Supported types are Text, Number, Date, Boolean, Selection and Multiple Selection. Default val-

ues can be specified and a mandatory flag can be set. Clicking the finthetaff icon removes the parameter from the list.

The passed parameters can be accessed by the report script by either calling the getParameter-Value(String name) or the getParameter(String name) method. These methods will return null if no parameter with the specified name could be found.

11.2.1. Supported Parameter Types

The following parameter types are supported:

Text Number Date Selection Multiple Selection

11.2.2. Dealing with Parameters

As mentioned before, the passed parameters can be retrieved from the context by the get-ParameterValue(String name) and getParameter(String name) methods.

The following code snippet shows how to access the parameters and how to use them in the report script:

```
StringBuffer query = new StringBuffer();
query.append("select tcr from KlarosTestCaseResult tcr where tcr.executionTime < ");
query.append(context.getParameterValue("executionTime"));
List<?> tcr = context.executeQuery(query.toString());
```

This query will retrieve all objects of type KlarosTestCaseResult that have an execution time smaller than the value passed with the parameter executionTime.

Alternately, the parameters can be directly accessed in the query as shown in the following code snippet:

```
query.append("select tcr from KlarosTestCaseResult tcr where tcr.executionTime <: executionTime");
List<?> tcr = context.executeParameterizedQuery(query.toString());
```

Parameters can be accessed from the layout template in the following way:

<p:text value="Test results for test runs with execution time < #{executionTime} ms" />



Note

Make sure to escape characters like &, < or > when using them in XML attributes in the layout template. In our example < is the escape character for <.

11.3. Generating a Report

To generate a report, click on the *Evaluate* icon and select *Reports* from the menu on the left side. A list of available report templates will be shown.

🔳 🞐 К L А F	ROS TEST MANAGE	EMENT	Finance Tracker 🧧 🗮	२ छ ∗ ≛∗
📝 Define	Report Templates			
📇 Plan		7 Entries - Page 1 of 1 🚺 🔌 1 🕨	▶ 10 ▼	7 & ≡
🔅 Execute	♀ Name ◆ △ Issue Overview Report △ Iteration Status Report △ Ide Excel Overview Report	Description Lists the issues in a project sorted by selectable parameters Summary of an iteration status in a project Lists the ions in a project	Revision Changed By ◆ Changed ◆ Status ◆ Forma 1.3 System Account 4/14/21,11:47 AM Approved PDF 1.2 System Account 4/14/21,11:47 AM Approved PDF 1.1 System Account 4/14/21,11:47 AM Approved PDF 1.3 System Account 4/14/21,11:47 AM Approved PDF	Action □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □
욙 Evaluate	 △ Job Status Report △ Job Status Report 	Lists the jobs in a project grouped by selectable parameters Lists the jobs in a project sorted by selectable parameters	1.2 System Account 4/14/21, 11:47 AM Approved PDF 1.3 System Account 4/14/21, 11:47 AM Approved PDF	
🔑 Configure	 System under Test Status Report Test Run Overview Report 	t Lists the test case results in a SuT sorted by selectable parameters Lists the test runs in a project sorted by selectable parameters	1.2 System Account 4/14/21, 11:47 AM Approved PDF 1.2 System Account 4/14/21, 11:47 AM Approved PDF	
	New			

Figure 11.5. Generating a Report

Clicking the [2] or [3] icon of a report causes it to be rendered as a PDF or Excel file. If the report script was defined with parameters, a pop-up window will prompt the user to enter the defined parameters before the report is generated.

= 👱 K L A R	OS TEST MANAGEMENT Finance Tracker 🖬 🔳 🔍 🔍	@~ _ ~
📝 Define	Report Templates	
🏝 Plan	7 Entries - Page 1 of 1 K 4 1 k 10 V	V & ≡
🏚 Execute	V Name Description Nerration Changed Status Pointal △ Issue Overview Report Lists the issues in a project sorted by selectable parameters 1.3 System Account 4/14/21, 11:47 AM Approved PDF △ Iteration Status Report Summary of an iteration status in a project 1.2 System Account 4/14/21, 11:47 AM Approved PDF △ Job Excel Overview Report Lists the jobs in a project 1.1 System Account 4/14/21, 11:47 AM Approved XLS	
🕒 Evaluate	A Job Status Report Lists the jobs in a project grouped by selectable parameters 1.2 System Account 4/14/21, 11:47 AM Approved PDF Discrete Approved PDF System Account 4/14/21, 11:47 AM Approved PDF Discrete Approved PDF	
🔑 Configure	A system under Test Status Report Lists the test case results in a SuT sorted by selectable parameters 1.2 System Account 4/14/21, 11:47 AM Approved PDF [A Test Run Overview Report Lists the test runs in a project sorted by selectable parameters 1.2 System Account 4/14/21, 11:47 AM Approved PDF [
	Please enter all required parameters Start Date End Date Test Run Sort Criteria Test Run Details Show Passed Results OK	



11.4. Example Report

This section provides an example report, which retrieves test case results and displays them depending on their status.

11.4.1. Creating the Report Script

The following code snippet shows the basic structure of a report script with all required imports. The code to retrieve the data will be implemented in the execute method. A more detailed description of the Klaros-Testmanagement API can be found in <u>de.verit.klaros.core.model</u>.

```
import de.verit.klaros.scripting.*;
import de.verit.klaros.core.model.*;
import java.util.*;
public class TestScript implements KlarosScript {
    public void execute(KlarosContext context) {...
    }
}
```

The next step is to retrieve the required data. The following code snippet shows how to build a query string to retrieve the data.

```
StringBuffer query = new StringBuffer();
query.append("select tcr from KlarosTestCaseResult tcr");
List<?> tcr = context.executeQuery(query.toString());
```

The data is returned in a List object. This list must be stored in the context so that it can be accessed from the layout template:

```
context.add("results", tcr);
```



Note

For more information on building queries please consult the HQL documentation.

The List object is stored in the context with the name results and can be accessed from the layout template by this name. If additional data is needed, execute the appropriate queries and store the processed data in the context using a different name.

11.4.2. Creating a PDF Report Template

The code snippets in this section show how to build a PDF layout template. For the official documentation on the layout language used, please refer to SeamPDF manual. More information regarding the object model can be found in <u>de.verit.klaros.core.model</u>.

The following code snippet shows the basic structure of a PDF layout template. Inside the <p:document> element, the content of the report is contained and grouped in chapters and sections.

```
<p:document xmlns:ui="http://java.sun.com/jsf/facelets" xmlns:f="http://java.sun.com/jsf/core"
xmlns:p="http://jboss.org/schema/seam/pdf" title="Klaros-Testmanagement Test Suite Report"
marginMirroring="true" author="#{user.name}" creator="#{user.name}" pageSize="A4">
...
```

Here the <u>de.verit.klaros.core.model.KlarosUser</u> parameter which is always present in the context is used via #{user.name} to specify the author of the PDF document.

The next code snippet shows how to define the report headers and footers. Here the current date and the <u>de.verit.klaros.core.model.KlarosUser</u> parameter in the context is used.

Next, the front page for the report is defined, showing a short summary of the report. To keep this example short, only a fragment with the user's name and email address is shown. For the complete script see <u>Section E.3, "Example Layout Template"</u>.

```
<p:font style="bold" size="16">
  <p:paragraph alignment="center" spacingAfter="5">
    <p:text value="#{user.name} (#{user.email})"/>
  </p:paragraph>
  </p:font>
```

11.5. Creating a Chart

Reports can be enhanced by providing a chart showing a graphical representation of the data retrieved. The data for these charts is also prepared by the report script and stored in the context. The layout template passed the data to the charting component of SeamPDF. This section explains how to create a pie chart as shown in Figure 11.7, "Pie Chart Example".





11.5.1. Pie Chart Report Script

The report script is not only used to retrieve the data, but also be to prepare the data for the layout template. The following snippet shows a way to prepare the data for a pie chart. For each possi-

ble result type, an empty List object is created. Then the <u>de.verit.klaros.core.model.KlarosTest-</u> <u>CaseResult</u> objects retrieved before are added to one of the lists depending on their result verdict. These lists can then later be accessed from the context by their corresponding key.

```
List<KlarosTestCaseResult> error = new ArrayList<KlarosTestCaseResult>();
List<KlarosTestCaseResult> failure = new ArrayList<KlarosTestCaseResult>();
List<KlarosTestCaseResult> success = new ArrayList<KlarosTestCaseResult>();
// Iterate over the results and retrieve the status
Iterator<KlarosTestCaseResult> iter = (Iterator<KlarosTestCaseResult>) tcr.iterator();
while (iter.hasNext()) {
    KlarosTestCaseResult result = iter.next();
    if (result.isError()) error.add(result);
    else if (result.isFailure()) failure.add(result);
    else if (result.isPassed()) success.add(result);
  }
  context.add("error", error);
  context.add("failure", failure);
  context.add("success", success);
```

11.5.2. Pie Chart Report Template

The following snippet shows the chart using the data that has been prepared by the report script in the previous section.

```
<p:paragraph horizontalAlignment="center">
<p:paragraph horizontalAlignment="center">
<p:paragraph horizontalAlignment="center">
<p:piechart title="Test Results" direction="anticlockwise" circular="true"
startAngle="30" labelGap="0.1" labelLinkPaint="#000000" plotBackgroundPaint="#ffffff"
labelBackgroundPaint="#ffffff" is3D="true">
<p:series key="results">
<p:series key="results">
<p:data key="results">
<p:data key="Error [#{error.size}]" value="#{error.size}" sectionPaint="#FF0A0A" />
<p:data key="Success [#{success.size}]" value="#{success.size}" sectionPaint="#FFC000"/>
<p:data key="Failure [#{failure.size}]" value="#{failure.size}" sectionPaint="#FFC00"/>
</p:series>
</p:piechart>
```

The piechart element defines the presentation and contents of the chart. For a pie chart a series element is required. The data is retrieved from the lists formerly stored in the context by the script.

<p:data key="Error [#{error.size}]" value="#{error.size}" sectionPaint="#FF0A0A" />

This code retrieves the list containing the results from the context and calls its size() method to determine the number of test cases. The pie chart is then rendered, using the three data sections, as seen in Figure 11.7, "Pie Chart Example".



Note

For detailed information on different chart types, please check the SeamPDF documentation.

11.5.3. Embedding Images

This section shows how to include an image into a report. To be accessible by the report template, the image file has to be stored in one of two possible locations:

• In a .jar file in the .klaros/resources folder

• On a web server, which can be accessed via a URL

When storing the image in a .jar file, it can be accessed using the following code snippet.

<p:image value="images/image.png"/>

The value attribute defines the image filename and the folder of the image location inside the jar file.

When providing an image via a URL it can be accessed using the following code snippet.

```
<p:html>
<img src="http://www.verit.de/images/logo-klaros-160.png" />
</p:html>
```

The src attribute defines the URL of the image location.

11.5.4. Creating an Excel Report Template

This section shows how to build an Excel report template for Klaros-Testmanagement. For detailed information please refer to SeamExcel manual. More information on the Klaros-Testmanagement object model can be found in <u>de.verit.klaros.core.model</u>.

The following snippet shows a basic Excel report template. The <e:workbook> element, contains all other report details.

Note the usage of the <u>de.verit.klaros.core.model.KlarosUser</u> and the <u>de.verit.klaros.core.model.K-</u> <u>larosTestCase</u> parameter from the context: #{user.name} and #{testCases}.

A worksheet uses a list of objects to generate the rows and columns of the Excel sheet. This list must be passed to the value attribute of the worksheet element.

In the var attribute, the name of the current object can be defined. For example: To use the test case name of the current object inside the worksheet block, the statement #{testCase.name} is used.

It is also possible to use multiple worksheet elements in one template, though in this example only one worksheet is used, with the username as the worksheet name.

Chapter 12. Import/Export

Klaros-Testmanagement provides several interfaces to import data from other tools and export its data into several formats.

User authentication is mandatory for all import operations. The user role required varies with the type of data being imported. For importing test results, the *Tester* role is sufficient as for requirements and test cases the *Manager* role is required.



LDAP Authentication

If the setting *Login default* is enabled in the LDAP configuration, then authentication is done exclusively via the LDAP directory. Local authentication information is always ignored in this case.

12.1. Importing Test Cases from Excel

Klaros-Testmanagement allows importing of test cases from Excel sheets structured in a predefined format which will be further detailed in <u>Section 12.1.1, "Import Format"</u>.

In addition to the standard fields provided for test cases, users of the Klaros-Testmanagement Enterprise Edition are also able to import data for user-defined fields. For this it is necessary to prepare the project according to what is listed in <u>Section 12.1.2, "Prerequisites"</u>.

The import interface consists of a simple REST interface which can be accessed by command line tools as well as custom applications. This is further detailed in <u>Section 12.1.3, "Execution"</u>.

12.1.1. Import Format

The Excel sheet used to import test cases has to follow this strict format:

- 1. Each test case is located on a separate sheet.
- 2. Each sheet is divided into three sections: general properties, test case steps and custom properties.
- 3. The fields A:1 to A:22 are reserved for the standard attributes of a test case. The corresponding values are entered in the fields B:1 to B:22. It is not mandatory to fill all fields.
- 4. Column G is for the names of any number of user-defined properties. In column H the corresponding values are entered.
- 5. The Step field in cell A:25 **must** remain. Changing or deleting the A:25 field will cause the import of this table to be rejected. From here the definition of the test steps begins:
 - Column A for the test step number,
 - Column B for the action,
 - Column C for the precondition,
 - Column D for the postcondition and

	Α	В	С	D	E	F	G	Н	1
1	ID	TC00051	Created automa	atically			Cycle	Major	
2	Name	Test Login					Category	Regression Tests	
3	Revision	1.0	Created automa	atically					
4							Custo	om Properties	
5	Description	Check the basic functionality							
6	Precondition	Server is installed							
7	Postcondition	User is authenticated							
8	Expected Result	Main page opens							
9									
10	Note	Required for basic operation.							
11	Area	FUNCTIONAL							
12	Design Method	BLACK_BOX	Seneral Prone	rties					
13	Variety	POSITIVE	beneral riope	THEO					
14	Execution	MANUAL							
15	Priority	HIGH							
16	State	APPROVED							
17	Team	QA							
18	Level	SYSTEM							
19	Document Base	Requirements 1.5 (12.07.2020)							
20	Dependency	-							
21	Evaluation	MANUAL							
22	Traceability	UC-002							
23	Estimated Duration (ms)	3000							
24									
25	Step	Action	Precondition	Postcondition	Expected Result				
26	1	Open Login Page			Login page displayed				
27	T 2	Enter Username	Test Steps						
28	3	Enter Password			Password should not be visible				
29	4	Press OK			User is logged in				
30									
31	Do not remove field								
32	A:25!								
33									
34									

• Column E for the expected result.

Figure 12.1. Test Case Excel Sheet Sample

12.1.1.1. General Properties

The table below lists the cell coordinates for the general test case properties. The test case ID does not have to be specified. The ID is generated during the import. None of these values are mandatory. Some fields can take only predefined values which are listed there.

Coordinate	Value
B2	Name
B3	Revision (only used in export)
B5	Description
B6	Precondition
B7	Postcondition
B8	Expected Result
B10	Note
B11	Area (FUNCTIONAL, NON_FUNCTIONAL, STRUCTURAL, REGRESSION, RE_TEST)
B12	Design Method (BLACK_BOX, WHITE_BOX)
B13	Variety (POSITIVE, NEGATIVE)
B14	Execution (MANUAL, AUTO)
B15	Priority (LOW, MEDIUM, HIGH)
B16	State (NEW, APPROVED, LOCKED, INVISIBLE)
B17	Team
B18	Level (COMPONENT, INTEGRATION, SYSTEM, ACCEPTANCE)

Coordinate	Value
B19	Document Base
B20	Dependency
B21	Evaluation
B22	Traceability
B23	Estimated Duration (ms)

Table 12.1. General Property Coordinates

12.1.1.2. Test Steps

Test cases may contain a variable number of test steps. These are listed one per row starting from cell A:26 downwards. The import parser will stop reading steps once it encounters an empty step number.

Coordinate	Value
A26 (ff)	Step Number
B26 (ff)	Action
C26 (ff)	Precondition
D26 (ff)	Postcondition
E26 (ff)	Expected Result

Table 12.2. Test Step Coordinates

12.1.1.3. Custom Properties

Only available in Klaros-Testmanagement Enterprise Edition

Test cases may contain custom properties. These may be listed as name/value pairs starting from cell G1 downwards.

For the import to succeed, the names of the listed properties must match with user defined properties defined for the project targeted by the import.

The value field for enumeration properties must exactly match the name of the enumeration property value definition to be parsed.

If an empty name cell is found, processing stops.

Coordinate	Value
G1 (ff)	Property name
H1 (ff)	Property value

Table 12.3. Custom Property Coordinates

12.1.2. Prerequisites

To import Excel based test cases the following prerequisites have to be met:

1. The file that is going to be imported has to be XLS or XLSX format.
- 2. The project that should contain the imported test cases has to be already created.
- 3. If user defined properties (only available in Klaros-Testmanagement Enterprise Edition) are to be imported they have to be created for the project prior to starting the import process.

12.1.3. Execution

To import Excel-based test cases a REST interface is available which allows you to upload your data from the command-line or other applications.

The following example shows how to import an Excel sheet containing test cases into the project named P00001 using the curl command line application. Curl should be available in every Linux distribution and as part of the Cygwin http://www.cygwin.com/ distribution or as a command line tool from http://curl.haxx.se/download.html for the Microsoft Windows operating system family.



Incompatible Windows Powershell curl Alias

Windows Powershell also defines an alias named **curl** which also can be used for this purpose but takes a completely different set of arguments. If you prefer to use the Linux compatible curl executable under Powershell, use **curl.exe** instead of **curl**.

```
curl -v -T TestCases.xls "<klaros-app-url>/seam/resource/rest/import/testcase/xls?config=P00001\
&username=user&password=secret"
```

Example 12.1. Excel Test Case Import via Command Line



ID Format

All objects which are referenced during import (like projects or test cases) contain five digits in their id.

For example, P00001 is a valid project ID, while P0001 and P000001 are not.

12.2. Importing Test Cases from XML

Klaros-Testmanagement allows importing of test cases from XML Files. The import process will always create new instances of the supplied test cases for each invocation.

The format of the XML file is described in <u>Appendix C, Test Case Import File Specification</u>. The XML schema is available at the following URL: https://www.klaros-testmanagement.com/files/ schema/klaros-testcases-1.0.xsd.

12.2.1. Prerequisites

To import XML based test cases the following prerequisites have to be met:

- 1. The file that is going to be imported has to in XML format.
- 2. The project that should contain the imported test cases has to be already created.
- 3. If user defined properties are to be imported, they have to be created in the project prior to starting the import process.

12.2.2. Execution

To import XML-based test cases a REST interface is available which allows you to upload your data from the command-line or other applications.

The following example shows how to import an XML file containing test cases into the project named P00001 using the curl command line application. Curl should be available in every decent Linux distribution and as part of the Cygwin http://www.cygwin.com/ distribution or as a command line tool from http://curl.haxx.se/download.html for the Microsoft Windows operating system family.

curl -v -T TestCases.xml "<klaros-app-url>/seam/resource/rest/import/testcase/xml?config=P00001\ &username=user&password=secret"

Example 12.2. XML Test Case Import via Command Line

12.3. Importing Requirements from Excel

In order to correctly import requirements from Excel spreadsheets, the data must exist in a specified format. This format is described in detail in the following section <u>Section 12.1.1, "Import</u> <u>Format"</u>.

In addition to the standard fields provided for requirements, users of the Klaros-Testmanagement Enterprise Edition are also able to import data for user-defined fields. For this it is necessary to prepare the project accordingly. You can find more information about this at <u>Section 12.1.2</u>, "Pre-requisites".

The import interface consists of a simple REST interface which can be accessed by command line tools as well as custom applications. This is further detailed in <u>Section 12.1.3, "Execution"</u>.

12.3.1. Import Format

The Excel sheet used to import requirements has to follow this strict format:

- 1. Each requirement is located on a separate sheet.
- 2. Each sheet is divided into two sections: general properties and custom properties.
- 3. The fields A:1 to A:9 are reserved for the standard attributes of a requirement. The corresponding values are entered in the fields B:1 to B:9. It is not mandatory to fill all fields.
- 4. Column G is for the names of any number of user-defined properties. In column H the corresponding values are entered.

	A	В	С	D	E	F	G	н	- I
1	ID	R00001	Created auto	omatically			Department	Internal	
2	Name	Require authorization					Category	Security	
3	Revision	1.0	Created auto	omatically			Phase	BETA	
4									
5	Summary	Unauthorized users must not be able to access services.					Custom F	Properties	
6	Description	During alpha development phase, it was possible to access services without providing a valid username/password combination. The server must now be secured against unauthorized access.							
7									
8	Priority	HIGH Gener	al Properti	es					
9	State	APPROVED							
10									
11									

Figure 12.2. Requirement Excel Sheet Sample

12.3.1.1. General Properties

The table below lists the cell coordinates for the general requirement properties. None of these values are mandatory. Some fields can take only predefined values which are listed there.

Coordinate	Value
B1	ID (only used in export)
B2	Name
B3	Revision (only used in export)
B5	Summary
B6	Description
B8	Priority (LOW, MEDIUM, HIGH)
B9	State (NEW, APPROVED, LOCKED, INVISIBLE)

Table 12.4. General Property Coordinates

12.3.1.2. Custom Properties

Requirements may contain custom properties. These may be listed as name/value pairs starting from cell G1 downwards.

For the import to succeed, the names of the listed properties must match with user defined properties defined for the project targeted by the import.

The name in the name column must exactly match the name of a user defined property for a value to be parsed correctly. The parsing stops once an empty name cell has been found. The value field for enumeration properties must exactly match the name of the enumeration property value definition to be parsed.

Coordinate	Value
G1 (ff)	Property name
H1 (ff)	Property value

Table 12.5. Custom Property Coordinates

12.3.2. Prerequisites

To import Excel based requirements the following prerequisites have to be met:

- 1. The file that is going to be imported has to be in XLS or XLSX format.
- 2. The project that should contain the imported requirements has to be already created.
- 3. If user defined properties (only available in Klaros-Testmanagement Enterprise Edition) are to be imported they have to be created for the project prior to starting the import process.

12.3.3. Execution

To import Excel-based requirements a REST interface is available which allows you to upload your data from the command-line or other applications.

The following example shows how to import an Excel sheet containing requirements into the project named P00001 using the curl command line application. curl is available in almost any Linux distribution and for the Microsoft Windows operating system family as part of the Cygwin http://www.cygwin.com/ distribution or as a command line tool from http://curl.haxx.se/download.html.

curl -v -T Requirements.xls "<klaros-app-url>/seam/resource/rest/import/requirement/xls?config=P00001\
&username=user&password=secret"

Example 12.3. Excel Requirement Import via Command Line

12.4. Importing and Synchronizing Requirements from XML

Klaros-Testmanagement allows to both import and synchronize requirements from XML Files.

An import is limited to a one-time action that always creates new instances of the delivered requirements. Synchronization, on the other hand, allows updating existing requirements with the content of the XML file and creating new revisions of them as needed.

The format of the XML file is described in <u>Appendix D, Requirement Import File Specification</u>. The XML schema is available at the following URL: https://www.klaros-testmanagement.com/files/ schema/klaros-requirements-1.0.xsd.

12.4.1. Prerequisites

To import XML based test cases the following prerequisites have to be met:

- 1. The file that is going to be imported has to be in *.XML format.
- 2. The project that should contain the imported requirement has to be already created.
- 3. If user defined properties are to be imported, they have to be created in the project prior to starting the import process.
- 4. If the imported requirements should be assigned to test cases via externalTestCaseId, Test cases with this external ID must already exist.

12.4.2. Importing

To import XML-based requirements a REST interface is available which allows you to upload your data from the command-line or other applications.

The following example shows how to import an XML file containing requirements into the project named P00001 using the curl command line application. curl is available in almost any Linux distribution and for the Microsoft Windows operating system family as part of the Cygwin http:// www.cygwin.com/ distribution or as a command line tool from http://curl.haxx.se/download.html.

curl -v -T Requirements.xml "<klaros-app-url>/seam/resource/rest/import/requirement/xml?config=P00001\
&username=user&password=secret"

Example 12.4. XML Requirement Import via Command Line

12.4.3. Synchronization

To synchronize XML-based requirements a different REST interface is available (ending with / sync/requirement/xml). It understands the same import format as defined for importing requirements but requires additional elements to be present for correct operation.

In contrast to an import a synchronization action is intended to update the set of requirements stored in Klaros-Testmanagement from arbitrary external applications on a regular basis.

The synchronization supports creating and updating as well as revisioning of requirements. Changes to the requirements will be reflected in the Klaros-Testmanagement database on a perfield basis for each synchronization action.



Synchronization Overrides Local Changes!

Be aware that all data received via synchronization will overwrite any changes made to the requirements stored in Klaros-Testmanagement.



externalId

Supplying the externalId element (See <u>Section D.11, "<externalId>"</u>) will identify a requirement for subsequent synchronisation attempts. It is required field when using this interface. If you change this value, a new requirement object will be created instead of replacing the content of the previously created requirement.



revision

Supplying the revision element (See <u>Section D.15, "<revision>"</u>) will identify the version of a requirement for subsequent synchronisation attempts.

- If the value is empty, the currently selected revision of the requirement is updated.
- If a new revision is entered, a new version of the requirement is created and updated.
- If an older revision is entered, the entered revision will be updated.

The following example shows how to synchronize an XML file containing requirements into the project named P00001 using the curl command line application. curl is available in almost any Linux distribution and for the Microsoft Windows operating system family as part of the Cygwin http://www.cygwin.com/ distribution or as a command line tool from http://curl.haxx.se/download.html.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<r:container xmlns:r="http://klaros-testmanagement.com/export-requirements-1.0">
<r:requirements>
<r:requirement>
<r:attributes/>
<r:externalId>RTM-00001</r:externalId>
<r:revision>1.0</r:revision>
<r:priority>MEDIUM</r:priority>
```

```
<pr:shortname>Remote-controlled door panels / unlocking</r:shortname>
<r:summary>Doors must me remote controllable.</r:summary>
</r:requirement>
</r:requirements>
</r:container>
```

Example 12.5. XML Requirement Synchronization via Command Line

curl -v -T Requirements.xml "<klaros-app-url>/seam/resource/rest/sync/requirement/xml?config=P00001\
&username=user&password=secret"

Example 12.6. XML Requirement Import via Command Line

12.5. Importing Test Results

Test automation tools typically generate test results in file form. These files can be imported via a REST interface and automatically converted into test results.

12.5.1. The JUnit XML++ format

JUnit XML++ is the generic format used internally by Klaros-Testmanagement for importing test case results from external sources. This format is upward compatible with the JUnit XML format.

By extending the JUnit XML format, the JUnit XML++ format enables you to specify individual test steps of a test, as well as to embed binary attachments in an XML document. In addition, with the result type inconclusive it allows to indicate ambiguous results.

12.5.1.1. The JUnit XML format

Unfortunately, a clear formal JUnit XML format specification does not exist, as JUnit itself does not create XML reports. XML reporting usually results from the processing of the JUnit invocation within a build tool such as the Ant JUnit task, the Maven Surefire plugin, or Gradle.

The JUnit XML format has established itself as a generic (quasi-)standard format for test case results across tool and programming language boundaries and is supported by a large number of automation tools. Despite the ambiguity of the format, there have been several attempts in the past to document the format in the form of an XML schema. Detailed commented schema definitions can be found, for example, here: https://llg.cubic.org/docs/junit/ and here: https://github.com/windyroad/Junit-Schema/blob/master/JUnit.xsd.

12.5.1.2. The JUnit XML++ format

The JUnit XML format unfortunately does not offer all the possibilities that are supported in Klaros-Testmanagement, especially it lacks:

Das JUnit XML Format bietet leider nicht alle Möglichkeiten, die in Klaros-Testmanagement unterstützt werden, insbesondere fehlt hier:

- Describing and evaluating tests at the test step level
- · Adding binary attachments (images, documents, etc.) to the test results
- Support additional test ratings (unclear)

We have added this to the JUnit XML format without changing the existing syntax. In the following we call this new format JUnit XML++.

12.5.1.2.1. Test step results

Within a testcase element there can now occur any number of teststep elements. These must contain a time attribute which specifies the execution time of the test step in seconds.

As with test cases, error, failure or skipped elements within a test step control their rating.

Also likewise to the test cases, system-out as well as system-err elements are supported. If these contain content, they are automatically added as binary attachments to the test case result generated during import.

Optionally, the elements action, precondition, postcondition and expectedResult can be mapped to the test step result.

If these are available, they are transferred to the corresponding fields of a test step result in Klaros-Testmanagement. This information is not mandatory, but it is recommended at least for the action field in order to better identify individual test steps.



Example 12.7. Sample Test Test-step-result.xml

12.5.1.2.2. Attachments

For the documentation of a test (step) result, it may be useful to supplement it with a file (screenshot, document, etc.) in addition to the textual description. To enable this, additional content can be defined in the system-out and system-err elements.

After successful extraction and saving, this content is not transferred to any additionally created system-out/system-err attachments. The content is defined according to the Jenkins plugin JUnit Attachments and may appear in three different forms:

Base64 Encoded Attachment	<pre>Syntax: [[ATTACHMENT_DATA «attached-file.xyz» «encoded data»]]</pre>
	In "encoded data" a base64-encoded binary content is expect- ed. After successful decoding, this is stored under the file name attached-file.xyz as an attachment to the test case result or test step result.

So this method saves the entire attachment as part of the result file. This requires more storage space, but allows access to the content at any time.

URL Attachment Syntax: [[ATTACHMENT_URL|/«url/to/attached-file.xyz»]] An attempt will be made to download the contents of the URL url/to/attached-file.xyz and to save it as an attachment under the filename attached-file.xyz. This method expects the attachment to be available on the network. It is mandatory that the Klaros instance has access to this URL at import time for the extraction to succeed. **File Attachment** Syntax: [[ATTACHMENT|/«path/to/attached-file.xyz»]] An attempt is made to download the contents of the file path/ to/attached-file.xyz and save it as an attachment under the file name attached-file.xyz. With this method it is expected that the attachment is provided in the local file system. It is mandatory that the Klaros instance has access to this file at the time of import for the extraction to succeed.

12.5.1.2.3. Test results

For test case results (testcase) as well as for test step results (teststep), it is possible to set inconclusive (unclear) as result type besides error, failure or skipped, since this is supported in Klaros-Testmanagement but not in JUnit.

12.5.2. Prerequisites

To import test results the following prerequisites have to be met:

1. The project, iteration, test environment and system under test that should contain the imported test case or test suite results have to be created.

12.5.3. Execution

The default URL of the import interface is located at http://localhost:18080/klaros-web/seam/ resource/rest/importer. The content is transferred via an HTTP PUT request using the above URL and various URL query parameters.



The <klaros-app-url> Term

The http://localhost:18080/klaros-web URL shown above is the default klaros application url when accessing Klaros-Testmanagement from the host it is installed on and may vary with your setup. Throughout this chapter the term <klaros-app-url> will be used instead to reflect this.

The following parameters are supported:

config	The ID of the project to import the	e results into (e.g. P0001).			
iteration	The ID of the iteration to relate th This parameter is optional.	ne results to (e.g. ITR00001).			
env	The ID of the test environment in which the tests have been run (e.g. ENV00001). Please make sure that this test environment already exists in the project before starting the import.				
sut	The ID of the system under test in which the tests have been run (e.g. SUT00001). Please make sure that this system under test already exists in the project before starting the import.				
type	The type of the import format. The following types are supported:				
	aunit	AUnit			
	boost	Boost Test			
	Check	Check			
	cpptest	CppTest			
	cppunit	CppUnit			
	ctest	ctest (CMake)			
	cunit	CUnit			
	fitnesse	Fitnesse			
	fpunit	Free Pascal Unit			
	gtester	GLib/gtester			
	googletest	GoogleTest			
	jbehave	JBehave			
	jmeter	JMeter			
	jsunit	JsUnit			
	jubula	Jubula / GUIdancer			
	junit	JUnit / Gauge / Robot Framework / Selenium / Squish / TestNG			
	mbunit	MBUnit			
	mstest	MSTest			
	nunit	NUnit			
	phpunit	PHPUnit			

	qftest	QF-Test		
	qtestlib	QTestLib		
	tessy	TESSY		
	testcomplete	TestComplete		
	tusar	TUSAR		
	uft	Unified Functional Testing (UFT) / QuickTest Profes- sional (QTP)		
	cpptestunit	UnitTest++		
	valgrind	Valgrind		
	xUnit.net	xunitdotnet		
job	The optional ID of a job this test run JOB00001). If given, the job must before starting the import.	a should be assigned to (e.g. already exist in the project		
time	The time of the test execution. Please make sure the fixed for- mat for the time is dd.MM.yyyy_HH:mm. This parameter can be used to override the test run date which is otherwise bound to the time of import.			
createTestSuiteResults	If set to true test suite results are automatically created for the corresponding test suite information contained in the result file. This information may vary with the import format. Additionally, a corresponding test suite is created for the test suite result if it did not yet exist.			
autoCreateTestCases	If set to false test cases are no longer automatically created for test case results for which no associated test case can be found. The default setting for this parameter is true.			
username	The username for the import.			
password	The password for the import.			

A complete example for a QF-Test import URL would look like this:

http://localhost:18080/klaros-web/seam/resource/rest/importer?\
config=P00001&env=ENV00001&sut=SUT00001&type=qftest&time=01.03.2011_12:00&username=me&password=secret

Example 12.8. QF-Test import URL sample

The result file is contained in the HTTP request body.

The **curl** command line tool can be used on Linux or Windows/Cygwin to trigger an import in a single command line.

curl -v -H "Content-Type: text/xml" -T <test result file> \

"<klaros-app-url>/seam/resource/rest/importer?config=P00001&env=ENV00001&sut=SUT00001&type=junit\
&time=23.05.2011_14:55"

Example 12.9. Curl Command Line Example



Incompatible Windows Powershell curl alias

Windows Powershell also defines an alias named **curl** which also can be used for this purpose but takes a completely different set of arguments. If you prefer to use the Linux compatible curl executable under Powershell, use **curl.exe** instead of **curl**.

```
curl -Method put -InFile <test result file> -Uri \
"<klaros-app-url>/seam/resource/rest/importer?config=P00001&env=ENV00001&sut=SUT00001&type=junit\
&time=23.05.2011_14:55"
```

Example 12.10. Powershell Curl Alias Command Line Example

12.5.4. Jenkins Plugin

This plugin integrates a continuous integration server with Klaros-Testmanagement by publishing the test results of a Jenkins build to the Klaros-Testmanagement application. The test results will be stored in the Klaros-Testmanagement database for further evaluation and reporting purposes. You can find the installation and configuration guide for the plugin in the Jenkins GitHub page.

12.6. Exporting Test Results

Klaros-Testmanagement provides a JUnit XML export interface that enables the export of test results.

12.6.1. Execution

The default URL of the export interface is located at http://localhost:18080/klaros-web/seam/ resource/rest/export/result/junit. The content is transferred via an HTTP GET request using the above URL and various URL query parameters.



The <klaros-app-url> Term

The http://localhost:18080/klaros-web URL shown above is the default klaros application url when accessing Klaros-Testmanagement from the host it is installed on and may vary with your setup. Throughout this chapter the term <klaros-app-url> will be used instead to reflect this.

The following parameters are supported:

config	The ID of the project to export the results from (e.g. P0001).
env	The ID of the test environment in which the tests have been run (e.g. ENV00001).
sut	The ID of the system under test in which the tests have been run (e.g. SUT00001).
job	The ID of the job to export the results of (e.g. JOB00001).

testRun	The ID of the test run to export the results of (e.g. TRU0000001).
username	The username for the export.
password	The password for the export.

Both individual test results of a job or a test run and all test results of a system under test or a combination of system under test and test environment can be exported. This can be controlled by specifying the parameters.

The **curl** command line tool can be used on Linux or Windows/Cygwin to trigger an export in a single command line.

curl \ "<klaros-app-url>/seam/resource/rest/export/result/junit?config=P00001&job=JOB00001&user=me&password=secret"

In this example, the results of the job JOB00001 from the project P00001 are exported.

Example 12.11. Curl Command Line Example



Incompatible Windows Powershell curl alias

Windows Powershell also defines an alias named **curl** which also can be used for this purpose but takes a completely different set of arguments. If you prefer to use the Linux compatible curl executable under Powershell, use **curl.exe** instead of **curl**.

curl -Method get -Uri \

"<klaros-app-url>/seam/resource/rest/export/result/junit?config=P00001&testRun=TRU0000001&username=me&password=secret"

In this example, the results of the test run TRU0000001 from the project P00001 are exported.

Example 12.12. Powershell Curl Alias Command Line Example

12.7. Exporting Table Content

enterprise Only available in Klaros-Testmanagement Enterprise Edition

With Klaros-Testmanagement it is possible to export the content of all tables to an Excel file. The current filter and sort settings will be considered.

🔳 👱 К L A R (os test man		Finance Tracker 🔲 📑	۹ 🚱 ۲ 🛓 ۲
📝 Define	Test Suites			
	New			Save Discard
Plan 🔁	+ ๙ Ѻ ӱ Ә ӥ	7 Entries - Page 1 of 1 🛛 🖌 🔰 🖡	▶ ▶ 10 🔍 초∨	Q × 丞 =
🚔 Eveeute	ID			Excel Export
	TS00008 台 1.0	Check the connection to the server	Finance Tracker 2.1.0	PDF Export
	TS00007 台 1.0	Edit standing orders		ML Export
🕒 Evaluate	TS00006 台 1.0	Overview		7 🕜 🖸 🖨 🔟
	TS00004 1.0	Connection	Finance Tracker 1.0.0	2 2000
🖌 Configure	TS00003 🔒 1.0	Display activities of different time periods		4 🖉 🖸 🖨 🔟
Je configure	TS00002 台 1.0	Bank Transfer		· CCQ
	TS00001 台 1.0	Update Dashboard		4 🖸 🖸 🖨 🗓
	ID Revision	Name	System under Test	Test Cases Action
	New			Save Discard

Figure 12.3. Export Table Content to Excel

12.8. Backup/Recovery

To move data between different database installations, or to selectively import data, Klaros-Testmanagement provides the functionality to import and export database content via XML files. <u>Section 10.6, "Backup"</u> explains the import and export functionality in detail.

12.8.1. Backup via REST

Klaros-Testmanagement Enterprise Edition contains a REST-based interface that grant access to back up files of single projects. The following URI for example returns the backup xml file of the project P00004: http://{host}:{port}/klaros-web/seam/resource/rest/io/backup/project/P00004.



Note

For more information on how to access REST-based interfaces, see <u>Chapter 13, The</u> <u>Remote API</u>.

Chapter 13. The Remote API



Only available in Klaros-Testmanagement Enterprise Edition

13.1. Overview

Klaros-Testmanagement Enterprise Edition comes with a REST-based interface which can be used to retrieve data like test cases, results or information about users. The data can be retrieved either in XML or JSON format. The API is accessible from the URL http://{host}:{port}/klaros-web/api/v1/resources/.

http://localhost:8080/klaros-web/api/v1/resources/users/name	e/Harald Martens	
ET OPOST OPUT OPATCH ODELETE OHEAD OO	PTIONS Other	
thorization	Basic YWRtaW46YWRtaW4=	
ne	value	
NEW HEADER		
		Clear Se
200: OK 🕢 Loading time:289ms		U.S. Contraction of the second s
latus: 200. OK Eduling time.209113		
Response headers (0)	Request headers (0)	Redirects (0)
Response headers (0) Response Y TO CLIPBOARD SAVE AS FILE	Request headers (0)	Redirects (0)

Figure 13.1. JSON Output after Searching for a User by its Name.

13.2. REST Clients

There are numerous ways to connect to the remote API. The following lists give an overview of some freely available clients.

Browser Plugins

Advanced REST client plugin for Google Chrome

Google Chrome Web Store

RESTED for Firefox

Firefox Add-Ons

RestMan Add-On for Opera

Opera Add-Ons

Standalone Clients

• Wget

https://www.gnu.org/software/wget/

• cURL

http://curl.haxx.se/

13.3. Documentation

Klaros-Testmanagement comes with an interactive documentation for the Remote API which can be accessed from a browser at http:/{host}:{port}/klaros-web/api-doc/resources/.

Klaros-Testmanagement REST Service API: /api/v1/resources/systemsUnderTest/{id}/{testRuns} Resources Data Type									
GET /ani/v1/resources/systems11	Home	/ Reso	urces / /api/v1/resources/systems	sUnderTest/{id}/{testRuns}					
Back to Top	/api	i/v1	/1/resources/systemsUnderTest/{id}/{testRuns}						
	Returns	the test	runs executed for this system unde	r test.					
	GET Returns	/api/\	/1/resources/systems	sUnderTest/{id}/{testRuns} 🗹					
	Request	Parame	ters						
name type description			default	constraints					
	id	path	the system under test id						
	accept	query	The Media Type information acce and application/xml.	epted by the client. Currently supported values are: applic	ation/json	application/json			
	limit	query	Page limit when retrieving paged	lists of objects. The default value is 0 and disables pagin	g.	0	int		
	offset	query	Paging offset when retrieving page	ged lists of objects.		0	int		
	Respons	se Body							
	media	type		data type	descript	ion			
	applica	tion/json		TestRuns (JSON)	the test r	runs			
	applica	tion/xml		testRuns (XML)					

Figure 13.2. Remote API Documentation

Appendix A. Access Permissions

A.1. Role Permission Overview – Community Edition

Actions/Roles	Administrator	Manager	Tester	Guest
Execute test cases	0	\bigcirc	\bigcirc	×
Generate reports	0	\bigcirc	\bigcirc	0
View Test Results	S	0	\bigcirc	0
Backup and Restore ^a	0	\bigcirc	×	×
Create, edit and delete test objects	S	0	×	×
Create, edit and delete users with the role <i>Tester</i> ^b	•	0	×	×
Create, edit and delete users with the role Administrator or Manager	•	×	×	×
Edit system preferences ^b	0	×	×	×

^a Global managers are able to restore any project. However, they may only export projects which have no project-specific roles defined or in which they have the role *Manager*.

^bNot changed by project role.

Table A.1. Role Permission Overview – Community Edition

A.2. Role Permission Overview – Enterprise Edition

Actions/Roles	Administrator	Manager	Tester	Guest
Configure dashboard	0	0	\bigcirc	0
Execute jobs	0	0	0	×
Export Excel tables	0	0	0	0
Generate custom reports	0	0	0	0
Generic full text search	O	0	0	0
Track test run execution	0	0	0	×
Create and assign jobs	0	0	×	×
Create Custom Reports ^a	0	0	×	×
Define custom fields	0	0	×	×
Configure LDAP authentication ^a	0	×	×	×

^aNot changed by project role

Table A.2. Role Permission Overview - Enterprise Edition

Appendix B. The Klaros Object Model API Reference

B.1. Klaros Object Model API Reference

- B.1.1. de.verit.klaros.core.model
- B.1.1.1. IKlarosAttachment

This interface provides access to a binary attachment.

B.1.1.1.1. Synopsis

```
public interface IKlarosAttachment
    extends IKlarosRepositoryEntity<KlarosAttachment> {
```

```
// Public Methods
public abstract long getSize();
}
```



B.1.1.1.2. getSize()

public abstract long getSize();

Gets the size in bytes of this attachment.

Parameters

return

the size

B.1.1.2. IKlarosCategoryNode

A Klaros Category Node.

B.1.1.2.1. Synopsis

public interface IKlarosCategoryNode extends IKlarosPersistentObject {

```
// Public Methods
public abstract CategoryTree getCategoryTree();
public abstract String getDescription();
public abstract KlarosCategoryNode getParent();
public abstract String getShortname();
public abstract List<KlarosCategoryNode> getSubCategories();
}
```



B.1.1.2.2. getCategoryTree()

public abstract CategoryTree getCategoryTree();

Gets the category tree.

Parameters

return

the category tree

B.1.1.2.3. getDescription()

public abstract String getDescription();

Gets the description. Parameters the description return B.1.1.2.4. getParent() public abstract KlarosCategoryNode getParent(); Gets the parent category node. Parameters the parent return B.1.1.2.5. getShortname() public abstract String getShortname(); Gets the shortname. Parameters the shortname return B.1.1.2.6. getSubCategories() public abstract List<KlarosCategoryNode> getSubCategories(); Gets the sub categories. Parameters the sub categories return B.1.1.3. IKlarosCategoryTree A category tree. B.1.1.3.1. Synopsis public interface IKlarosCategoryTree extends IKlarosNamedEntity<KlarosCategoryTree> { // Public Methods

public abstract String getDescription(); public abstract KlarosCategoryNode getRootNode(); public abstract String getShortname();

}



B.1.1.3.2. getDescription()

public abstract String getDescription();

Gets the description.

return

the description

B.1.1.3.3. getRootNode()

public abstract KlarosCategoryNode getRootNode();

Gets the root category node.

Parameters

return

the root

Gets the shortname.

Parameters

return

the shortname

B.1.1.4. IKlarosConfiguration

This interface provides access to data of a test project.

B.1.1.4.1. Synopsis

public interface IKlarosConfiguration
 extends IKlarosLabeledObject<KlarosConfiguration> {

```
// Public Methods
public abstract String getDescription();
public abstract Set<KlarosTestEnvironment> getEnvs();
public abstract Set<KlarosIssueManagement> getIssueManagementSystems();
public abstract Set<KlarosIssue> getIssues();
public abstract Set<KlarosCategoryTree> getIterationCategoryTrees();
public abstract Set<KlarosIteration> getIterations();
public abstract Set<KlarosJob> getJobs();
public abstract Set<KlarosCategoryTree> getRequirementCategoryTrees();
public abstract Set<KlarosRequirement> getRequirements();
public abstract KlarosRequirementsManagement getRequirementsManagementSystem();
public abstract Set<KlarosSUTImplementation> getSuts();
public abstract Set<KlarosCategoryTree> getSystemUnderTestCategoryTrees();
public abstract Set<KlarosCategoryTree> getTestCaseCategoryTrees();
public abstract Set<KlarosTestCase> getTestCases();
public abstract Set<KlarosCategoryTree> getTestEnvironmentCategoryTrees();
public abstract Set<KlarosTestRun> getTestRuns();
public abstract Set<KlarosCategoryTree> getTestSegmentCategoryTrees();
public abstract Set<KlarosCategoryTree> getTestSuiteCategoryTrees();
public abstract Set<KlarosTestSuite> getTestSuites();
public abstract Boolean isSecured();
```

}



B.1.1.4.2. getDescription()

public abstract String getDescription();

Returns the project description.

Parameters

return

The description of the project.

B.1.1.4.3. getEnvs()

public abstract Set<KlarosTestEnvironment> getEnvs();

Returns the project test environments.

Parameters

return

Set containing the test environments of the project.

B.1.1.4.4. getlssueManagementSys public abstract Set <klarosi< th=""><th>tems() ssueManagement> getIssueManagementSystems();</th></klarosi<>	tems() ssueManagement> getIssueManagementSystems();			
Returns the related issue management systems.				
	Parameters			
return	Set containing the related issue management systems of the project.			
B.1.1.4.5.getlssues() public abstract Set <klarosi< td=""><td><pre>ssue> getIssues();</pre></td></klarosi<>	<pre>ssue> getIssues();</pre>			
Returns the issues.				
	Parameters			
return	KlarosSet containing the issue objects of the project.			
B.1.1.4.6. getIterationCategoryTrees() public abstract Set <klaroscategorytree> getIterationCategoryTrees();</klaroscategorytree>				
Returns the project iteration cat	egory trees.			
	Parameters			
return	Set containing the iteration category trees of the project.			
B.1.1.4.7.getIterations() public abstract Set <klarosi< td=""><td><pre>teration> getIterations();</pre></td></klarosi<>	<pre>teration> getIterations();</pre>			
Returns the project iteration.				
	Parameters			
return	KlarosSet containing the iteration objects of the project.			
<pre>B.1.1.4.8. getJobs() public abstract Set<klarosjob> getJobs();</klarosjob></pre>				
Returns the project jobs.				
	Parameters			
return	KlarosSet containing the job objects of the project.			
B.1.1.4.9. getRequirementCategory public abstract Set <klarosc< td=""><td>Trees() ategoryTree> getRequirementCategoryTrees();</td></klarosc<>	Trees() ategoryTree> getRequirementCategoryTrees();			
Returns the project requirement category trees.				
Parameters				
return	Set containing the requirement category trees of the project.			
B.1.1.4.10. getRequirements()				

	Returns the project requirements.		
	Parameters		
	return	Set containing the requirement objects of the project.	
B.1.1.4.11. getRequirementsManagementSystem() public abstract KlarosRequirementsManagement getRequirementsManagementSystem();			
	Returns the related requirement r	nanagement system.	
		Parameters	
	return	Set containing the related requirement management system of the project.	
<pre>B.1.1.4.12. getSuts() public abstract Set<klarossutimplementation> getSuts();</klarossutimplementation></pre>			
	Returns the project SUTs (system	ns under test).	
		Parameters	
	return	KlarosSet containing the SUT objects of the project.	
B.1.1.4.13. getSystemUnderTestCategoryTrees() public abstract Set <klaroscategorytree> getSystemUnderTestCategoryTrees();</klaroscategorytree>			
	Returns the project system under	test category trees.	
		Parameters	
	return	Set containing the system under test category trees of the project.	
B.1.1.4.14. getTestCaseCategoryTrees() public abstract Set <klaroscategorytree> getTestCaseCategoryTrees();</klaroscategorytree>			
	Returns the project test case cate	egory trees.	
		Parameters	
	return	Set containing the test case category trees of the project.	
<pre>B.1.1.4.15. getTestCases() public abstract Set<klarostestcase> getTestCases();</klarostestcase></pre>			
Returns the project test cases.			
		Parameters	
	return	Set containing the test case objects of the project.	
B.1.1.4.16. getTestEnvironmentCategoryTrees() public abstract Set <klaroscategorytree> getTestEnvironmentCategoryTrees();</klaroscategorytree>			
	Returns the project test environment category trees.		

Parameters

	return	Set containing the test environment category trees of the project.
<pre>B.1.1.4.17. getTestRuns() public abstract Set<klarostestrun> getTestRuns();</klarostestrun></pre>		
	Returns the project test runs.	
		Parameters
	return	Set containing the test run objects of the project.
B.1.1	.4.18. getTestSegmentCategory public abstract Set <klaroscat< td=""><td>Trees() .egoryTree> getTestSegmentCategoryTrees();</td></klaroscat<>	Trees() .egoryTree> getTestSegmentCategoryTrees();
	Returns the project test segment of	category trees.
		Parameters
	return	Set containing the test segment category trees of the project.
B.1.1	.4.19. getTestSuiteCategoryTree public abstract Set <klaroscat< td=""><td>S() egoryTree> getTestSuiteCategoryTrees();</td></klaroscat<>	S() egoryTree> getTestSuiteCategoryTrees();
	Returns the project test suite cate	gory trees.
		Parameters
	return	Set containing the test suite category trees of the project.
<pre>B.1.1.4.20. getTestSuites() public abstract Set<klarostestsuite> getTestSuites();</klarostestsuite></pre>		
	Returns the project test suites.	
		Parameters
	return	Set containing the test suite objects of the project.
B.1.1	.4.21. isSecured() public abstract Boolean isSec	ured();
	Returns whether this project is see	cured or not.
		Parameters
	return	true if this project is secured, false if not.
B.1.1	.5. IKlarosEnumValue	
	A Custom property enumeration v	alue.
B.1.1	.5.1. Synopsis	
<pre>public interface IKlarosEnumValue extends IKlarosPersistentObject {</pre>		
	// Public Methods	

```
public abstract String getValue();
}
```



B.1.1.5.2. getValue()

public abstract String getValue();

Gets the enumeration value.

Parameters

return

B.1.1.6. IKlarosExternalLink

This interface provides access to externally stored information about an object.

the value

B.1.1.6.1. Synopsis

public interface IKlarosExternalLink extends IKlarosPersistentObject {

```
// Public Methods
public abstract String getReference();
}
```



B.1.1.6.2. getReference()

public abstract String getReference();

Get the reference to the externally stored information.

Parameters

return

The reference to the information.

B.1.1.7. IKlarosExternalServer

This interface provides access to external systems.

B.1.1.7.1. Synopsis

}

public interface IKlarosExternalServer<T> extends IKlarosLabeledObject<T> {

```
// Public Methods
public abstract String getDescription();
public abstract String getType();
public abstract String getUrl();
```



B.1.1.7.2. getDescription()

public abstract String getDescription();

Get the description of the external system.

Parameters

return

The description of the external system.

B.1.1.7.3. getType()
 public abstract String getType();

Get the type of this system.

Parameters

return

The type

B.1.1.7.4. getUrl()

public abstract String getUrl();

Get the base url of this system.

Parameters

return

The url

B.1.1.8. IKlarosIssue

This interface provides access to a software issue.

B.1.1.8.1. Synopsis

}

public interface IKlarosIssue extends IKlarosNamedEntity<KlarosIssue> {

```
// Public Methods
@Deprecated(since="5.3.3")
public abstract KlarosSUTImplementation getAcceptedIn();
public abstract Date getCreationDate();
public abstract String getDescription();
public abstract String getExternalId();
public abstract KlarosIssueManagement getIssueManagement();
public abstract Date getLastSynched();
public abstract String getOwner();
public abstract String getPriority();
public abstract String getRemoteBrowseUrl();
public abstract String getReporter();
public abstract String getResolution();
public abstract String getState();
public abstract String getSubject();
public abstract Set<KlarosSUTImplementation> getSystemsUnderTest();
public abstract Set<KlarosTestCase> getTestCases();
public abstract boolean isResolved();
```



B.1.1.8.2. getAcceptedIn()

@Deprecated(since="5.3.3")
public abstract KlarosSUTImplementation getAcceptedIn();

Gets the system under test this issue has been accepted/detected in.

Parameters

return

the system under test

B.1.1.8.3. getCreationDate()

public abstract Date getCreationDate();

Gets the creation date of this issue.

Parameters

return

the creation date of this issue.

B.1.1.8.4. getDescription()
 public abstract String getDescription();

Get the description.

Parameters

return

The description of this issue.

B.1.1.8.5. getExternalId()

public abstract String getExternalId();

Gets the external id of this issue, if available. This id is generated by the external issue management system upon creation of the issue.

Parameters

return

the external id

B.1.1.8.6. getIssueManagement()

public abstract KlarosIssueManagement getIssueManagement();

Gets the issue management this issue belongs to.

Parameters

return

the issue management

Gets the last synchronization date of this issue.

Parameters

return

the last synchronization date of this issue.

B.1.1.8.8. getOwner()

public abstract String getOwner();

Gets the name of the owner/assignee of this issue.

Parameters

return

the owner name of this issue.

B.1.1.8.9. getPriority()

public abstract String getPriority();

Gets the priority of this issue. The format and content of this value varies with the issue management system and its configuration.

Parameters

return

the priority of this issue.

B.1.1.8.10. getRemoteBrowseUrl()

public abstract String getRemoteBrowseUrl();

Returns an url to browse the issue in the remote issue management system.

Parameters

return

the remote browse url

B.1.1.8.11. getReporter()

public abstract String getReporter();

Gets the name of the reporter of this issue.

Parameters

return

the reporter name of this issue.

B.1.1.8.12. getResolution()

public abstract String getResolution();

Gets the resolution of this issue. The format and content of this value varies with the issue management system and its configuration.

Parameters

return

the resolution of this issue.

B.1.1.8.13. getState()

public abstract String getState();

Gets the state of this issue. The format and content of this value varies with the issue management system and its configuration.

Parameters

return

the state string of this issue.

B.1.1.8.14. getSubject()

public abstract String getSubject();

Gets the subject of this issue.

Parameters

return

the subject string of this issue.

B.1.1.8.15. getSystemsUnderTest()

public abstract Set<KlarosSUTImplementation> getSystemsUnderTest();

Get the related systems under test.

Parameters

return

Set of systems under test which have detected this issue.

B.1.1.8.16. getTestCases()

public abstract Set<KlarosTestCase> getTestCases();

Get the related test cases.

Parameters

return

Set of test cases which have detected this issue.

B.1.1.8.17. isResolved()

public abstract boolean isResolved();

Checks if this issue is resolved.

Parameters

return

true, if resolved

B.1.1.9. IKlarosIssueManagement

The interface for accessing issue management systems.

B.1.1.9.1. Synopsis

public interface IKlarosIssueManagement
 extends IKlarosExternalServer<KlarosIssueManagement> {

```
// Public Methods
public abstract Set<KlarosIssue> getIssues();
public abstract String getProject();
}
```



B.1.1.9.2. getIssues()

public abstract Set<KlarosIssue> getIssues();

Gets the referenced issues of this system.

Parameters

return

the issues

B.1.1.9.3. getProject()

public abstract String getProject();

Get the optional project of this system.

Parameters

return

The project

B.1.1.10. IKlarosIteration

An iteration in a project.

B.1.1.10.1. Synopsis

public interface IKlarosIteration extends IKlarosNamedEntity<KlarosIteration> {

```
// Public Methods
public abstract Set<KlarosAttachment> getAttachments();
public abstract Set<KlarosCategoryNode> getCategories();
public abstract StateDef getCurrentState();
public abstract String getDescription();
public abstract Date getDueDate();
public abstract Set<KlarosTestEnvironment> getEnvs();
public abstract Set<KlarosJob> getJobs();
public abstract String getShortname();
public abstract String getSuccessCriteria();
public abstract Set<KlarosTestRun> getSuts();
public abstract Set<KlarosTestRun> getTestRuns();
}
```



B.1.1.10.2. getAttachments()

public abstract Set<KlarosAttachment> getAttachments();

Gets the attachments associated with this iteration.

Parameters

return

the attachments

B.1.1.10.3. getCategories()

public abstract Set<KlarosCategoryNode> getCategories();

Gets the categories this object belongs to. Each category node will belong to a different category tree.

		Parameters
	return	the category nodes
B.1.1	0.4.getCurrentState() <pre>public abstract StateDef getCurrentState();</pre>	
	Gets the current state.	
		Parameters
	return	the current state
B.1.1	<pre>0.5. getDescription() public abstract String getDescription();</pre>	
	Gets the description.	
		Parameters
	return	the description
B.1.1	.1.10.6. getDueDate() public abstract Date getDueDate();	
	Gets the due date.	
		Parameters
	return	the due date
B.1.1	1.1.10.7.getEnvs() public abstract Set <klarostestenvironment> getEnvs();</klarostestenvironment>	
	Gets the test environments associ	iated with this iteration.
		Parameters
	return	the test environments
B.1.1	1.10.8.getJobs() public abstract Set <klarosjob> getJobs();</klarosjob>	
	iteration.	
		Parameters
	return	the jobs
<pre>B.1.1.10.9. getShortname() public abstract String getShortname();</pre>		rtname();
	Gets the short name.	
		Parameters
	return	the short name
B.1.1.10.10. getStartDate()

public abstract Date getStartDate();

Gets the start date.

Parameters

return

the start date

Gets the success criteria.

Parameters

return

the success criteria

B.1.1.10.12. getSuts()

public abstract Set<KlarosSUTImplementation> getSuts();

Gets the systems under test associated with this iteration.

Parameters

return

the systems under test

B.1.1.10.13. getTestRuns()

public abstract Set<KlarosTestRun> getTestRuns();

Gets the test runs associated with this iteration.

Parameters

the test runs

return

B.1.1.11. IKlarosJob

This interface provides access to the data of a job.

B.1.1.11.1. Synopsis

public interface IKlarosJob extends IKlarosNamedEntity<KlarosJob> {

// Public Methods
public abstract Set<KlarosAttachment> getAttachments();
public abstract KlarosConfiguration getConfiguration();
public abstract Set<KlarosJobDependency> getDependentJobs();
public abstract String getDescription();
public abstract Date getDueDate();
public abstract KlarosTestEnvironment getEnv();
public abstract String getEstimatedTime();
public abstract Long getEstimatedTimeInMilliseconds();
public abstract JobPriority getJobPriority();
public abstract JobStatus getJobStatus();
public abstract KlarosJob getParent();
public abstract KlarosJob getParent();
public abstract Integer getProgress();

public abstract Set<KlarosJobDependency> getRequiredJobs(); public abstract Date getStartDate(); public abstract List<KlarosJob> getSubJobs(); public abstract Integer getSuccessRate(); public abstract String getSummary(); public abstract KlarosSUTImplementation getSut(); public abstract KlarosTestCase getTestCase(); public abstract Set<KlarosTestRun> getTestRuns(); public abstract KlarosTestSuite getTestSuite(); public abstract List<KlarosJobUpdateAction> getUpdateAction(); public abstract List<KlarosJobTimeBlock> getWork(); }



B.1.1.11.2. getAttachments()

public abstract Set<KlarosAttachment> getAttachments();

Gets the attachments associated with this job.

Parameters

return

the attachments

B.1.1.11.3. getDependentJobs()

public abstract Set<KlarosJobDependency> getDependentJobs();

Gets the jobs depending on this jobs state.

Parameters

return

the dependent jobs

B.1.1.11.4. getRequiredJobs()

public abstract Set<KlarosJobDependency> getRequiredJobs();

Gets the jobs on which states this job is depending on.

Parameters

return

the required jobs

B.1.1.11.5. getTestRuns()

public abstract Set<KlarosTestRun> getTestRuns();

Gets the test runs executed with this job.

Parameters

return

the test runs

Gets the update action list.

Parameters

return

the update action

B.1.1.11.7. getWork()

public abstract List<KlarosJobTimeBlock> getWork();

Gets the list of work items done on this job.

Parameters

return

the work items

B.1.1.12. IKlarosJobDependency

This interface provides access to the data of a job.

B.1.1.12.1. Synopsis

public interface IKlarosJobDependency extends IKlarosPersistentObject {

```
// Public Methods
public abstract Set<JobStatus> getJobStates();
public abstract UUID getKey();
public abstract Integer getProgress();
public abstract KlarosJob getRequiredBy();
public abstract KlarosJob getRequires();
public abstract Integer getSuccessRate();
public abstract Set<Verdict> getVerdicts();
}
```



B.1.1.12.2. getJobStates()

public abstract Set<JobStatus> getJobStates();

Gets the job states.

Parameters

return

the job states

B.1.1.12.3. getKey()

public abstract UUID getKey();

Get the internal key of the object. This key is globally unique.

Parameters

The internal key.

Gets the job this dependency is required by.

Parameters

return

return

the job this dependency is required by

B.1.1.12.5. getRequires()

public abstract KlarosJob getRequires();

Returns the job this dependency requires for evaluation.

Parameters

return

the job this dependency requires

B.1.1.12.6. getVerdicts()

public abstract Set<Verdict> getVerdicts();

Gets the verdicts.

Parameters

return

the verdicts

B.1.1.13. IKlarosJobTimeBlock

The job time block defines a certain amount of time the job has been worked on.

B.1.1.13.1. Synopsis

```
public interface IKlarosJobTimeBlock
  extends IKlarosLabeledObject<KlarosJobTimeBlock> {
```

```
// Public Methods
public abstract String getDescription();
public abstract Date getDoneAt();
public abstract String getDuration();
public abstract Long getDurationInMilliseconds();
public abstract String getEstimatedTimeLeft();
public abstract Long getEstimatedTimeLeftInMilliseconds();
public abstract KlarosJob getJob();
public abstract KlarosTestRun getTestRun();
}
```



B.1.1.13.2. getDescription()

public abstract String getDescription();

Gets the work description.

Parameters

return

the work description

B.1.1.13.3. getDoneAt()

public abstract Date getDoneAt();

Gets the date this work was done at.

Parameters

return

the start date

B.1.1.13.4. getDuration()

public abstract String getDuration();

Gets the work duration as a String.

Parameters

return

the duration

B.1.1.13.5. getDurationInMilliseconds()

public abstract Long getDurationInMilliseconds();

Gets the work duration in milliseconds.

Parameters

	return	the work duration in hours	
B.1.1	.13.6.getEstimatedTimeLeft() public abstract String getEstimatedTimeLeft();		
	Gets the estimated time left for thi	is job as a String.	
		Parameters	
	return	the estimated time left	
B.1.1	.13.7. getEstimatedTimeLeftInM public abstract Long getEstim	illiseconds() atedTimeLeftInMilliseconds();	
	Gets the estimated time left for this job in milliseconds.		
		Parameters	
	return	the estimated time left in hours	
B.1.1	.13.8. getJob() public abstract KlarosJob get	Job();	
	Gets the job this job time block belongs to.		
		Parameters	
	return	the job	
B.1.1	.13.9.getTestRun() public abstract KlarosTestRun	getTestRun();	
	Gets the test run this job time block belongs to.		
		Parameters	
	return	the test run	
B.1.1	.14. IKlarosJobUpdateAction		
	This interface provides access to t	the job update action.	
B.1.1	.14.1. Synopsis		

public interface IKlarosJobUpdateAction
 extends IKlarosLabeledObject<KlarosJobUpdateAction> {

// Public Methods



```
// Public Methods
public abstract Iterable<T> asIterable();
public abstract Date getCreated();
public abstract KlarosUser getCreator();
public abstract UUID getKey();
public abstract KlarosUser getLastEditor();
public abstract Date getLastUpdated();
public abstract String getName();
public abstract boolean isEnabled();
```





B.1.1.15.2. asIterable()

public abstract Iterable<T> asIterable();

Return this object as an iterable containing just this object.

Parameters

return

the iterable container

B.1.1.15.3. getKey()

public abstract UUID getKey();

Get the internal key of the object. This key is globally unique.

Parameters

return

The internal key.

B.1.1.15.4. getName()
 public abstract String getName();

Get the name of the object. This matches the id field visible in the UI.

Parameters

return

The name of the object.

B.1.1.15.5. isEnabled()

public abstract boolean isEnabled();

Returns the value of the enabled flag of this object.

Parameters

return

true if the entity is enabled, else false

B.1.1.16. IKlarosNamedEntity

This interface provides access to data of a properties owner.

B.1.1.16.1. Synopsis

}

public interface IKlarosNamedEntity<T> extends IKlarosLabeledObject<T> {

```
// Public Methods
public abstract KlarosUser getAssignee();
public abstract List<IKlarosProperty> getProperties();
public abstract String getPropertyValue(String propertyName);
public abstract boolean isDefinedProperty(String propertyName);
```



B.1.1.16.2. getProperties()

public abstract List<IKlarosProperty> getProperties();

Gets the list of properties.

Parameters

	return	the properties		
	B.1.1.16.3. getPropertyValue(String) public abstract String getPropertyValue(String propertyName);			
	Gets the property value	Gets the property value for the given property name.		
		Parameters		
	propertyName	the property name		
	return	the property value or null if not name is not present		
B.1.1.16.4. isDefinedProperty(String) public abstract boolean isDefinedProperty(String propertyName);				
	Check if a property identified by given name is a defined property.			
		Parameters		
	propertyName	The name of the property to check.		
	return	true if the identified property is a defined property, false else		

B.1.1.17. IKlarosPersistentObject

Base interface.

B.1.1.17.1. Synopsis

public interface IKlarosPersistentObject {
}

«interface» IKlarosPersistentObject

B.1.1.18. IKlarosProperty

A user defined property.

B.1.1.18.1. Synopsis

public interface IKlarosProperty extends IKlarosPersistentObject {

```
// Public Methods
public abstract String getName();
public abstract String getValue();
}
```



B.1.1.18.2. getName()

public abstract String getName();

Gets the property name.

Parameters

return

the property name

B.1.1.18.3. getValue()
 public abstract String getValue();

Gets the property value.

Parameters

return

the property value

B.1.1.19. IKlarosRepositoryEntity

This interface provides access to binary content.

B.1.1.19.1. Synopsis

public interface IKlarosRepositoryEntity<T> extends IKlarosLabeledObject<T> {

// Public Methods
public abstract String getMimeType();
public abstract String getUuid();
public abstract String getVersion();

}



B.1.1.19.2. getMimeType()

public abstract String getMimeType();

Gets the mime type of the attachment.

Parameters

return

the mime type

B.1.1.19.3. getUuid()

public abstract String getUuid();

Get the unique id of this attachment. This may be resolved by calling KlarosContext.getAttachmentURL() to an URL pointing to the attachment data.

Parameters

return

The uuid of this attachment.

B.1.1.19.4. getVersion()

public abstract String getVersion();

Gets the version of this attachment.

Parameters

return

the version

B.1.1.20. IKlarosRequirement

This interface provides access to data of a requirement.

B.1.1.20.1. Synopsis

}

```
public interface IKlarosRequirement<T,S extends Revision<?>>
    extends IKlarosRevision<T, S> {
```

```
// Public Methods
public abstract Set<KlarosAttachment> getAttachments();
public abstract Set<KlarosCategoryNode> getCategories();
public abstract KlarosConfiguration getConfiguration();
public abstract Set<KlarosTestCase> getCoveringTestCases();
public abstract String getDescription();
public abstract Set<String> getExternalNames();
public abstract Set<KlarosIteration> getIterations();
public abstract Set<KlarosIteration> getIterations();
public abstract String getShortname();
public abstract String getState();
public abstract String getState();
public abstract String getSummary();
```



B.1.1.20.2. getAttachments()

public abstract Set<KlarosAttachment> getAttachments();

Gets the attachments associated with this requirement.

		Parameters
	return	the attachments
B.1.1	.20.3. getCategories() public abstract Set <klaroscat< td=""><td>tegoryNode> getCategories();</td></klaroscat<>	tegoryNode> getCategories();
	Gets the categories this object bel tree.	longs to. Each category node will belong to a different category
		Parameters
	return	the category nodes
B.1.1	.20.4. getConfiguration() public abstract KlarosConfigu	uration getConfiguration();
	Get configuration.	
		Parameters
	return	The related configuration.
B.1.1	.20.5. getCoveringTestCases() public abstract Set <klarostes< td=""><td><pre>stCase> getCoveringTestCases();</pre></td></klarostes<>	<pre>stCase> getCoveringTestCases();</pre>
	Get test cases covering the requir	ement.
		Parameters
	return	Set of test cases which cover this test requirement.
B.1.1	.20.6. getDescription() public abstract String getDes	<pre>scription();</pre>
	Get the description.	
		Parameters
	return	the string
B.1.1	.20.7. getExternalNames() public abstract Set <string> g</string>	getExternalNames();
	Gets the external requirement nar names are used to match importe	mes this requirement is associated with. External requirement ed requirements to existing requirements.
		Parameters
	return	the external names
B.1.1	.20.8. getExternalStatus() public abstract String getExt	cernalStatus();
	Gets the external status.	
		Parameters
	return	the external status

B.1.1	.20.9. getIterations() public abstract Set <klarosite< td=""><td>eration> getIterations();</td></klarosite<>	eration> getIterations();	
	Get the iterations this requirement is assigned to.		
		Parameters	
	return	Set of iterations this requirement is assigned to.	
B.1.1	.20.10. getPriority() public abstract RequirementPr	riority getPriority();	
	Gets the priority.		
		Parameters	
	return	the priority	
B.1.1	B.1.1.20.11.getShortname() public abstract String getShortname();		
	Gets the shortname.		
		Parameters	
	return	the shortname	
B.1.1	<pre>B.1.1.20.12. getState() public abstract String getState();</pre>		
	Gets the state.		
		Parameters	
	return	the state	
B.1.1	.20.13. getSummary() public abstract String getSum	<pre>imary();</pre>	
	Gets the summary.		
		Parameters	
	return	the summary	
B.1.1	B.1.1.21. IKlarosRequirementsManagement		
	The interface for accessing requir	rements management systems.	
B.1.1	.21.1. Synopsis		
	<pre>public interface IKlarosRequir extends IKlarosExternalServer</pre>	ementsManagement ~ <klarosrequirementsmanagement> {</klarosrequirementsmanagement>	
	<pre>// Public Methods public abstract String getPro }</pre>	<pre>>ject();</pre>	



B.1.1.21.2. getProject()

public abstract String getProject();

Get the optional project of this system.

Parameters

return

The project

B.1.1.22. IKlarosResult

This interface provides access to data of generic result.

B.1.1.22.1. Synopsis

public interface IKlarosResult<T> extends IKlarosNamedEntity<T> {

```
// Public Methods
public abstract KlarosConfiguration getConfiguration();
public abstract String getDescription();
public abstract long getExecutionTime();
public abstract KlarosJob getJob();
public abstract String getSummary();
public abstract KlarosTestRun getTestRun();
public abstract String getVerdict();
```

```
public abstract String getVerdict(Locale locale);
public abstract boolean isError();
public abstract boolean isFailure();
public abstract boolean isInconclusive();
public abstract boolean isPassed();
public abstract boolean isSkipped();
}
```



B.1.1.22.2. getConfiguration()

public abstract KlarosConfiguration getConfiguration();

Get configuration.

Parameters

return

The related configuration

B.1.1.22.3. getDescription()

public abstract String getDescription();

Get the test result description. This is usually set for failed/error status results.

Parameters

return

The test result description.

Get the test execution time in ms.

Parameters

return

The test execution time.

B.1.1.22.5. getJob()

public abstract KlarosJob getJob();

Gets the job this result belongs if one exists.

Parameters

return

the associated job

Get the test result summary. This is usually set for failed/error status results.

Parameters

return

The test result summary.

B.1.1.22.7. getTestRun()

public abstract KlarosTestRun getTestRun();

Get the associated test run.

Parameters

return

The test run that created this result.

B.1.1.22.8. getVerdict()

public abstract String getVerdict();

Returns the Test Case Result verdict as String. P = Passed U = Unknown S = Skipped E = Error F = Failed

Parameters

return

The Test Case Result verdict as String.

B.1.1.22.9. getVerdict(Locale)

public abstract String getVerdict(Locale locale);

Returns the Test Case Result verdict as String with localization. P = Passed U = Unknown S = Skipped E = Error F = Failed

Parameters

locale set location Information

return The Test Case Result verdict as String.

B.1.1.22.10. isError()

public abstract boolean isError();

Check if this is an error result. It is assumed, that error results have a property 'type' with the value 'E' or 'error'.

Parameters

return

true if this results represents an error.

B.1.1.22.11. isFailure()

public abstract boolean isFailure();

Check if this is a failure result. It is assumed, that failure results have a property 'type' with the value 'F' or 'failure'.

Parameters

return

true if this results represents a failure.

B.1.1.22.12. isInconclusive()

public abstract boolean isInconclusive();

Check if this is an inconclusive result. It is assumed, that inconclusive results have a property 'type' with the value 'l' or 'inconclusive'.

Parameters

return

true if this results represents an inconclusive result.

B.1.1.22.13. isPassed()

public abstract boolean isPassed();

Check if this is a result of a passed test case. It is assumed, that passed results have a property 'type' with the value 'P' or 'error'.

Parameters

return

true if this results represents an error.

B.1.1.22.14. isSkipped()

public abstract boolean isSkipped();

Check if this is a skipped result. It is assumed, that skipped results have a property 'type' with the value 'S' or 'skipped'.

```
Parameters
```

return

true if this results represents a skipped result.

```
B.1.1.23. IKlarosRevision
```

This interface provides access to a revisionable Klaros object.

```
B.1.1.23.1. Synopsis
```

```
public interface IKlarosRevision<T,S extends Revision<?>>
    extends IKlarosNamedEntity<T> {
```

```
// Public Methods
public abstract KlarosRevision<T, S> getPredecessor();
public abstract String getRevisionComment();
public abstract String getRevisionId();
public abstract KlarosRevision<T, S> getRoot();
public abstract KlarosRevision<T, S> getSuccessor();
}
```



B.1.1.23.2. getPredecessor() public abstract KlarosRe	evision <t, s=""> getPredecessor();</t,>		
Get the predecessor of the r	Get the predecessor of the revision.		
	Parameters		
return	The revision object that is the predecessor of this revision.		
B.1.1.23.3. getRevisionCommen public abstract String g	t() etRevisionComment();		
Get comment.			
	Parameters		
return	The comment of the revision.		
<pre>B.1.1.23.4. getRevisionId() public abstract String getRevisionId();</pre>			
Get the revision id.			
	Parameters		
return	The revision id.		
B.1.1.23.5. getRoot() public abstract KlarosRe	evision <t, s=""> getRoot();</t,>		
Get the root of the revision h	Get the root of the revision hierarchy.		
	Parameters		
return	The root revision object.		
B.1.1.23.6. getSuccessor() public abstract KlarosRevision <t, s=""> getSuccessor();</t,>			
Get the successor of the rev	Get the successor of the revision.		
	Parameters		
return	The revision object that is the successor of this revision.		
B.1.1.24. IKlarosSUTImplementa	ation		
This interface provides acce	ss to data of a system under test version.		
B.1.1.24.1. Synopsis			
public interface IKlarosS extends IKlarosNamedEnti	JTImplementation ty <klarossutimplementation> {</klarossutimplementation>		

```
// Public Methods
public abstract Set<KlarosAttachment> getAttachments();
public abstract Set<KlarosCategoryNode> getCategories();
public abstract KlarosConfiguration getConfiguration();
public abstract Set<KlarosIssue> getIssues();
public abstract Set<KlarosIteration> getIterations();
public abstract String getProductversion();
public abstract Set<KlarosTestRun> getTestRuns();
}
```



B.1.1.24.2. getAttachments()

public abstract Set<KlarosAttachment> getAttachments();

Gets the attachments associated with this system under test.

Parameters

return

the attachments

B.1.1.24.3. getCategories()

public abstract Set<KlarosCategoryNode> getCategories();

Gets the categories this system under test belongs to. Each category node will belong to a different category tree.

Parameters

return

the category nodes

B.1.1.24.4. getConfiguration()

public abstract KlarosConfiguration getConfiguration();

Get configuration.

Parameters

return

The related configuration.

B.1.1.24.5. getIssues()

public abstract Set<KlarosIssue> getIssues();

Gets the issues related to this system under test.

Parameters

return

the issues

B.1.1.24.6. getIterations()

public abstract Set<KlarosIteration> getIterations();

Get the iterations this system under test is assigned to.

Parameters

return

Set of iterations this system under test is assigned to.

B.1.1.24.7. getProductversion()

public abstract String getProductversion();

Get product version.

Parameters

return

The version id of the system under test.

B.1.1.24.8. getTestRuns()

public abstract Set<KlarosTestRun> getTestRuns();

Get the test runs performed with this system under test.

Parameters

return

Set of test runs performed for this system under test.

B.1.1.25. IKlarosTestCase

This class provides access to data of a test case.

B.1.1.25.1. Synopsis

public interface IKlarosTestCase extends IKlarosRevision<KlarosTestCase, TestCase> {

```
// Public Methods
public abstract Set<KlarosAttachment> getAttachments();
public abstract Set<KlarosCategoryNode> getCategories();
public abstract KlarosConfiguration getConfiguration();
public abstract Set<KlarosRequirement> getCovers();
public abstract String getDependency();
public abstract String getDescription();
public abstract TestDesignTechnique getDesignTechnique();
public abstract List<KlarosIssue> getDetectedIssues();
public abstract String getDocbase();
public abstract Long getEstimatedDuration();
public abstract TestEvaluationMethod getEvaluation();
public abstract TestExecutionMethod getExecution();
public abstract String getExpectedResult();
public abstract Set<String> getExternalNames();
public abstract List<KlarosJob> getJobs();
public abstract TestLevel getLevel();
public abstract String getNote();
public abstract String getPostcondition();
public abstract String getPrecondition();
public abstract TestPriority getPriority();
public abstract Set<KlarosTestCaseResult> getResults();
public abstract String getShortname();
public abstract String getState();
public abstract String getTeam();
public abstract List<KlarosTestCaseStep> getTestCaseSteps();
public abstract TestAreatopic getTestType();
public abstract String getTraceability();
public abstract TestVariety getVariety();
```

}



B.1.1.25.2. getAttachments()

public abstract Set<KlarosAttachment> getAttachments();

Gets the attachments associated with this test case.

Parameters

return

the attachments

B.1.1.25.3. getCategories()

public abstract Set<KlarosCategoryNode> getCategories();

Gets the categories this object belongs to. Each category node will belong to a different category tree.

Parameters

return

the category nodes

B.1.1.25.4. getConfiguration()

public abstract KlarosConfiguration getConfiguration();

Get the project configuration this test case revision belongs to.

Parameters

return

The related configuration.

B.1.1.25.5. getCovers()

public abstract Set<KlarosRequirement> getCovers();

Get covered requirements.

Parameters

return

Set of requirements which are covered by this test case.

The dependency of this test case.

Parameters

return

The dependency.

The description of this test case.

Parameters

return

The description.

B.1.1.25.8. getDesignTechnique()

public abstract TestDesignTechnique getDesignTechnique();

The design technique of this test case.

Parameters

return

The type.

Get detected issues.

Parameters

return

List of issues which have been detected by this test case.

The docbase of this test case.

Parameters

return

The docbase.

The estimated duration of this test case.

Parameters

return

The estimatedDuration.

The evaluation of this test case.

Parameters

return

The evaluation.

B.1.1.25.13. getExecution()

public abstract TestExecutionMethod getExecution();

The execution method of this test case.

Parameters

return

The execution method.

B.1.1.25.14. getExpectedResult()

public abstract String getExpectedResult();

The expected result of this test case.

Parameters

return

The expected result.

B.1.1.25.15. getExternalNames()

public abstract Set<String> getExternalNames();

Gets the external test case names this test case is associated with. External test case names are used to match imported test case result to existing test cases.

Parameters

return

the external names

B.1.1.25.16. getJobs()

public abstract List<KlarosJob> getJobs();

Gets the jobs associated with this test case.

Parameters

return

the jobs

The level of this test case.

Parameters

return

The level.

B.1.1.25.18. getNote()
 public abstract String getNote();

The note of this test case.

Parameters

return

The note.

The postcondition of this test case.

Parameters

return

B.1.1.25.20. getPrecondition()

public abstract String getPrecondition();

The precondition of this test case.

Parameters

return

The precondition.

The priority of this test case.

Parameters

return

The priority.

B.1.1.25.22. getResults()

public abstract Set<KlarosTestCaseResult> getResults();

Get test case results.

Parameters

return

Set of results of executions of this test case.

The short name (title) of this test case.

Parameters

return

The short name.

The type of this test case.

Parameters

return

The type.

The team.

B.1.1.25.25.getTeam()
 public abstract String getTeam();

The team responsible for this test case.

Parameters

return

B.1.1.25.26. getTestCaseSteps() public abstract List<KlarosTestCaseStep> getTestCaseSteps(); Get test case steps. Parameters return list of steps of this test case. B.1.1.25.27. getTestType() public abstract TestAreatopic getTestType(); The test type of this test case. Parameters return The test type. B.1.1.25.28. getTraceability() public abstract String getTraceability(); The traceability of this test case. Parameters return The traceability.

The variety of this test case.

Parameters

return

The variety.

B.1.1.26. IKlarosTestCaseResult

This interface provides access to data of a test case result.

B.1.1.26.1. Synopsis

}

public interface IKlarosTestCaseResult extends IKlarosResult<KlarosTestCaseResult> {

// Public Methods
public abstract Set<KlarosAttachment> getAttachments();
public abstract List<KlarosTestCaseStepResult> getStepResults();
public abstract KlarosTestCase getTestCase();
public abstract Integer getTestSuitePosition();
public abstract KlarosTestSuiteResult getTestSuiteResult();
public abstract boolean isPending();



B.1.1.26.2. getAttachments()

public abstract Set<KlarosAttachment> getAttachments();

Gets the attachments associated with this result.

Parameters

return

the attachments

B.1.1.26.3. getStepResults()

public abstract List<KlarosTestCaseStepResult> getStepResults();

Get the associated step results.

Parameters

return

List of test case step results.

B.1.1.26.4. getTestCase()

public abstract KlarosTestCase getTestCase();

Get the associated test case.

Parameters

return

The test case that has been executed to get this result.

B.1.1.26.5. getTestSuitePosition()

public abstract Integer getTestSuitePosition();

Return the position in the test suite result this test case result belongs to or null if this test case has not been executed by executing a test suite.

Parameters

return the position of the test case in the test suite, or null

B.1.1.26.6. getTestSuiteResult()

public abstract KlarosTestSuiteResult getTestSuiteResult();

Return the test suite result this test case result belongs to or null if this test case has not been executed by executing a test suite.

Parameters

return

the test suite result, or null

B.1.1.26.7. isPending()

public abstract boolean isPending();

Checks if the result is pending. This means that it has not yet been completely executed.

Parameters

return

true, if the result is pending

B.1.1.27. IKlarosTestCaseStep

This interface provides access to a test case step.

B.1.1.27.1. Synopsis

}

public interface IKlarosTestCaseStep<T> extends IKlarosTestStepContainer<T> {

// Public Methods
public abstract Set<KlarosAttachment> getAttachments();



B.1.1.27.2. getAttachments()

public abstract Set<KlarosAttachment> getAttachments();

Gets the attachments associated with this test case.

Parameters

return

the attachments

B.1.1.28. IKlarosTestCaseStepResult

This interface provides access to data of a test case step result.

B.1.1.28.1. Synopsis

public interface IKlarosTestCaseStepResult
 extends IKlarosResult<KlarosTestCaseStepResult> {
```
// Public Methods
public abstract String getAction();
public abstract Set<KlarosAttachment> getAttachments();
public abstract String getExpectedResult();
public abstract String getPostcondition();
public abstract KlarosTestCase getTestCase();
public abstract KlarosTestCaseResult getTestCaseResult();
}
```



B.1.1.28.2 pl	2.getAction() blic abstract String getActi	on();
Get	the action.	
		Parameters
ret	ırn	The action of this test case step.
B.1.1.28.3 pl	3. getAttachments() blic abstract Set <klarosatta< td=""><td>chment> getAttachments();</td></klarosatta<>	chment> getAttachments();
Get	s the attachments associated w	vith this test case.
		Parameters
ret	ırn	the attachments
B.1.1.28.4 pu	4.getExpectedResult() blic abstract String getExpe	ctedResult();
Get	the expected result.	
		Parameters
ret	Jrn	The expected result of this test case step.
B.1.1.28. pl	5.getPostcondition() blic abstract String getPost	condition();
Get	the postcondition.	
		Parameters
ret	urn	The postcondition of this test case step.
B.1.1.28.0 pt	5.getPrecondition() blic abstract String getPrec	ondition();
Get	the precondition.	
		Parameters
ret	ırn	The precondition of this test case step.
B.1.1.28. ⁻ pu	7.getTestCase() blic abstract KlarosTestCase	e getTestCase();
Get	test case.	
		Parameters
ret	ırn	The test case that has been executed to get this result.
B.1.1.28.8 pu	3. getTestCaseResult() blic abstract KlarosTestCase	Result getTestCaseResult();
Get	test case result.	

```
Parameters
```

return

The test case result that has been executed to get this result.

B.1.1.29. IKlarosTestEnvironment

This interface provides access to data of a test environment.

B.1.1.29.1. Synopsis

}

```
public interface IKlarosTestEnvironment
    extends IKlarosNamedEntity<KlarosTestEnvironment> {
```

```
// Public Methods
public abstract Set<KlarosAttachment> getAttachments();
public abstract Set<KlarosCategoryNode> getCategories();
public abstract String getDescription();
public abstract Set<KlarosIteration> getIterations();
public abstract Set<KlarosTestRun> getTestRuns();
```



B.1.1.29.2. getAttachments()

public abstract Set<KlarosAttachment> getAttachments();

Gets the attachments associated with this test environment.

Parameters

return

the attachments

B.1.1.29.3. getCategories()

public abstract Set<KlarosCategoryNode> getCategories();

Gets the categories this test environment belongs to. Each category node will belong to a different category tree.

Parameters

return

the category nodes

B.1.1.29.4. getIterations()

public abstract Set<KlarosIteration> getIterations();

Get the iterations this test environment is assigned to.

Parameters

return

Set of iterations this test environment is assigned to.

B.1.1.29.5. getTestRuns()

public abstract Set<KlarosTestRun> getTestRuns();

Gets the test runs associated with this test environment.

Parameters

return

Collection of test runs executed in the environment.

B.1.1.30. IKlarosTestExecutable

This class provides access to a test executable. This may wrap a test case or a test suite.

B.1.1.30.1. Synopsis

public interface IKlarosTestExecutable extends IKlarosPersistentObject {
}



B.1.1.31. IKlarosTestRun

This interface provides access to data of a test run.

B.1.1.31.1. Synopsis

}

public interface IKlarosTestRun extends IKlarosNamedEntity<KlarosTestRun> {

```
// Public Methods
public abstract KlarosConfiguration getConfiguration();
public abstract KlarosTestEnvironment getEnv();
public abstract KlarosIteration getIteration();
public abstract KlarosJob getJob();
public abstract int getNumberErrors();
public abstract int getNumberFailures();
public abstract int getNumberInconclusive();
public abstract int getNumberPassed();
public abstract int getNumberSkipped();
public abstract String getRelatedSummary();
public abstract Set<KlarosTestCaseResult> getResults();
public abstract String getRunId();
public abstract KlarosSUTImplementation getSut();
public abstract KlarosTestSuite getTestSuite();
public abstract Date getTimestamp();
public abstract Set<KlarosJobTimeBlock> getWork();
public abstract boolean isPending();
```



B.1.1.31.2. getConfiguration()

public abstract KlarosConfiguration getConfiguration();

Get configuration.

Parameters

return

The related configuration

B.1.1.31.3. getEnv()

public abstract KlarosTestEnvironment getEnv();

Get test environment.

Parameters

return

The test environment in which the test cases have been executed.

B.1.1.31.4. getIteration()

public abstract KlarosIteration getIteration();

Get the iteration in which the test run has been created, if available.

Parameters

return

The iteration in which the test run has been created

B.1.1.31.5. getJob()

public abstract KlarosJob getJob();

Get the job that initiated the test run, if available.

Parameters

return

The job that initiated the test run

Get number of test cases with errors of this test run. It is assumed, that error results have a property 'type' with the value 'E'.

Parameters

return

The number of error test cases

B.1.1.31.7. getNumberFailures()

public abstract int getNumberFailures();

Get number of failed test cases of this test run. It is assumed, that failed results have a property 'type' with the value 'F'.

Parameters

return

The number of failed test cases

B.1.1.31.8. getNumberInconclusive()

public abstract int getNumberInconclusive();

Get number of inconclusive test cases of this test run. It is assumed, that inconclusive results have a property 'type' with the value 'l'.

Parameters

The number of skipped test cases

B.1.1.31.9. getNumberPassed()

public abstract int getNumberPassed();

Get number of passed test cases of this test run. It is assumed, that passed results have a property 'testCasePassed' with value 'true'.

Parameters

return

return

The number of passed test cases

B.1.1.31.10. getNumberSkipped()

public abstract int getNumberSkipped();

Get number of skipped test cases of this test run. It is assumed, that skipped results have a property 'type' with the value 'S'.

Parameters

return

The number of skipped test cases

B.1.1.31.11. getRelatedSummary()

public abstract String getRelatedSummary();

Gets the summary of the job, test suite or test case related to this test run.

Parameters

return

the related summary

B.1.1.31.12. getResults()

public abstract Set<KlarosTestCaseResult> getResults();

Get results.

Parameters

return

Set of results of test case executions.

Get id of test run.

Parameters

return

The id of the test run

B.1.1.31.14. getSut()

public abstract KlarosSUTImplementation getSut();

Get the tested system	version.
	Parameters
return	The SUT version which has been tested
B.1.1.31.15. getTestSuite() public abstract Kla) arosTestSuite getTestSuite();
Get the test suite exec	uted with this test run, if available.
	Parameters
return	The test suite executed with this test run
B.1.1.31.16. getTimestamp public abstract Dat)() ce getTimestamp();
Get timestamp.	
	Parameters
return	The time the test run has been executed as Date object
B.1.1.31.17.getWork() public abstract Set	: <klarosjobtimeblock> getWork();</klarosjobtimeblock>
Get the job time blocks	s of this test run.
	Parameters
return	Set of job time blocks.
B.1.1.31.18. isPending() public abstract boo	plean isPending();
Checks if the test run i	s pending. This means that it has not yet been completely executed.
	Parameters
return	true, if the test run is pending
B.1.1.32. IKlarosTestSegm	ent
This interface provides	access to a test segment.
B.1.1.32.1. Synopsis	
public interface IKl extends IKlarosRevi	arosTestSegment <t,s extends="" revision<s="">> .sion<t, s=""> {</t,></t,s>
// Public Methods	

public abstract String getDescription();
public abstract String getExpectedResult();

```
public abstract String getNote();
public abstract String getPostcondition();
public abstract String getPrecondition();
public abstract String getShortname();
public abstract List<KlarosTestCaseStep> getSteps();
}
```



B.1.1.32.2. getDescription()

public abstract String getDescription();

Get the description.

Parameters

		Farameters
	return	The description of this test segment.
B.1.1	.32.3. getExpectedResult() public abstract String getExp	<pre>pectedResult();</pre>
	Get the expected result.	
		Parameters
	return	The expected result of this test case step.
B.1.1	.32.4. getNote() public abstract String getNot	re();
	Get the note.	
		Parameters
	return	The note of this test segment.
B.1.1	.32.5. getPostcondition() public abstract String getPos	<pre>stcondition();</pre>
	Get the postcondition.	
		Parameters
	return	The postcondition of this test case step.
B.1.1	.32.6. getPrecondition() public abstract String getPre	<pre>econdition();</pre>
	Get the precondition.	
		Parameters
	return	The precondition of this test case step.
B.1.1	.32.7.getShortname() public abstract String getSho	ortname();
	Get the shortname.	
		Parameters
	return	The shortname of this test case segment.
B.1.1	.32.8.getSteps() public abstract List <klaroste< td=""><td>estCaseStep> getSteps();</td></klaroste<>	estCaseStep> getSteps();

Get the test case steps.

Parameters

return

list of steps of this test segment.

```
B.1.1.33. IKlarosTestStepContainer
```

This interface provides access to a test step containers.

```
B.1.1.33.1. Synopsis
```

public interface IKlarosTestStepContainer<T> extends IKlarosNamedEntity<T> {

```
// Public Methods
public abstract String getAction();
public abstract String getExpectedResult();
public abstract String getPostcondition();
public abstract String getPrecondition();
}
```



B.1.1.33.2. getAction()

public abstract String getAction();

Get the action.	
	Parameters
return	The action of this test case step.
B.1.1.33.3. getExpectedRes public abstract Stri	ult() .ng getExpectedResult();
Get the expected result.	
	Parameters
return	The expected result of this test case step.
B.1.1.33.4. getPostcondition public abstract Stri	n() .ng getPostcondition();
Get the postcondition.	
	Parameters
return	The postcondition of this test case step.
B.1.1.33.5. getPrecondition(public abstract Stri) .ng getPrecondition();
Get the precondition.	
	Parameters
return	The precondition of this test case step.
B.1.1.34. IKlarosTestSuite	
This interface provides	access to data of a test suite.
B.1.1.34.1. Synopsis	
public interface IKla S> {	rosTestSuite <t,s extends="" revision<?="">> extends IKlarosRevision<t,< td=""></t,<></t,s>
<pre>// Public Methods public abstract Set< public abstract Set< public abstract Klar public abstract List public abstract Stri public abstract Klar public abstract Klar</pre>	<pre>%KlarosAttachment> getAttachments(); %KlarosCategoryNode> getCategories(); oosConfiguration getConfiguration(); %KlarosTestSuiteResult> getResults(); .ng getShortname(); rosSUTImplementation getSut(); %KlarosTestCase> getTestCases();</pre>



B.1.1.34.2. getAttachments()

public abstract Set<KlarosAttachment> getAttachments();

Gets the attachments associated with this test suite.

Parameters

return

the attachments

B.1.1.34.3. getCategories()

public abstract Set<KlarosCategoryNode> getCategories();

	Gets the categories this object bel tree.	ongs to. Each category node will belong to a different category		
		Parameters		
	return	the category nodes		
B.1.1	.34.4. getConfiguration() public abstract KlarosConfigu	<pre>uration getConfiguration();</pre>		
	Get configuration.			
		Parameters		
	return	The related configuration.		
B.1.1	.34.5. getResults() public abstract List <klaroste< td=""><td>estSuiteResult> getResults();</td></klaroste<>	estSuiteResult> getResults();		
	Get test suite results.			
		Parameters		
	return	Set of results of executions of this test suite.		
B.1.1	.34.6. getShortname() public abstract String getSho	rtname();		
	Get the short name of the test suit	te.		
		Parameters		
	return	String of the description of the test suite.		
B.1.1	1.1.34.7.getSut() public abstract KlarosSUTImplementation getSut();			
	Get the description of the test suit	e.		
		Parameters		
	return	String of the description of the test suite.		
B.1.1	.34.8. getTestCases() public abstract List <klaroste< td=""><td>estCase> getTestCases();</td></klaroste<>	estCase> getTestCases();		
Get the executables of this test suite.		ite.		
		Parameters		
	return	Set of test cases of this test suite.		
B.1.1	.34.9. getTestSuiteResultCount(public abstract Integer getTe) stSuiteResultCount();		
Return the number of test suite results in this test suite.				
	Parameters			
	return	The number of test suite results in this test suite.		

B.1.1.35. IKlarosTestSuiteResult

This interface provides access to data of a test suite result.

B.1.1.35.1. Synopsis

}

```
public interface IKlarosTestSuiteResult
  extends IKlarosResult<KlarosTestSuiteResult> {
```

```
// Public Methods
public abstract Set<KlarosTestCaseResult> getResults();
public abstract KlarosTestSuite getTestSuite();
public abstract boolean isPending();
```



B.1.1.35.2. getResults()

public abstract Set<KlarosTestCaseResult> getResults();

Gets the test case results for this test suite result. The list is sorted by the position in which the test cases appear in the test suite.

Parameters

return

the results

B.1.1.35.3. getTestSuite()

public abstract KlarosTestSuite getTestSuite();

Get the related test suite.

Parameters

return

The test suite that has been executed to get this result.

B.1.1.35.4. isPending()

public abstract boolean isPending();

Checks if the result is pending. This means that it has not yet been completely executed.

Parameters

return

true, if the result is pending

B.1.1.36. IKlarosUser

The user object.

B.1.1.36.1. Synopsis

public interface IKlarosUser extends IKlarosLabeledObject<KlarosUser> {

```
// Public Methods
public abstract String getEmail();
public abstract UserRole getRole();
public abstract String getUsername();
}
```



B.1.1.36.2. getEmail()

public abstract String getEmail();

The email address of this user.

Parameters

return

the email address

B.1.1.36.3. getRole()

public abstract UserRole getRole();

The role name of this user.

Parameters

return

the role name

B.1.1.36.4. getUsername()

public abstract String getUsername();

The username of this user as used when logging in.

Parameters

return

the username

B.1.1.37. KlarosAttachment

This class provides access to binary attachment.

B.1.1.37.1. Synopsis

}

implements IKlarosAttachment {

```
// Public Methods
public long getSize();
```

Methods inherited from de.verit.klaros.core.model.KlarosRepositoryEntity: getMimeType, get-Name, getUuid, getVersion

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.38. KlarosCategoryNode

A generic category node.

B.1.1.38.1. Synopsis

public class KlarosCategoryNode extends KlarosWrapper<KlarosCategoryNode, CategoryNode>

```
implements IKlarosCategoryNode {
    // Public Constructors
    public KlarosCategoryNode();
    // Public Methods
    public CategoryTree getCategoryTree();
    public String getDescription();
    public KlarosCategoryNode getParent();
    public String getShortname();
    public List<KlarosCategoryNode> getSubCategories();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



Constructor.

B.1.1.39. KlarosCategoryTree

A category tree.

B.1.1.39.1. Synopsis

public class KlarosCategoryTree extends

KlarosNamedEntity<KlarosCategoryTree, CategoryTree>

```
implements IKlarosCategoryTree {
    // Public Constructors
    public KlarosCategoryTree();
    // Public Methods
    public String getDescription();
    public KlarosCategoryNode getRootNode();
    public String getShortname();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



Serialization constructor.

B.1.1.40. KlarosConfiguration

This class provides access to the information stored for project's configuration.

B.1.1.40.1. Synopsis

}

public class KlarosConfiguration extends

KlarosLabeledObject<KlarosConfiguration,

Configuration>

```
// Public Methods
public int compareTo(KlarosConfiguration o);
public boolean equals(Object o);
public String getDescription();
public Set<KlarosTestEnvironment> getEnvs();
public Set<KlarosIssueManagement> getIssueManagementSystems();
public Set<KlarosIssue> getIssues();
public Set<KlarosCategoryTree> getIterationCategoryTrees();
public Set<KlarosIteration> getIterations();
public Set<KlarosJob> getJobs();
public Set<KlarosCategoryTree> getRequirementCategoryTrees();
public Set<KlarosRequirement> getRequirements();
public KlarosRequirementsManagement getRequirementsManagementSystem();
public Set<KlarosSUTImplementation> getSuts();
public Set<KlarosCategoryTree> getSystemUnderTestCategoryTrees();
public Set<KlarosCategoryTree> getTestCaseCategoryTrees();
public Set<KlarosTestCase> getTestCases();
public Set<KlarosCategoryTree> getTestEnvironmentCategoryTrees();
public Set<KlarosTestRun> getTestRuns();
public Set<KlarosCategoryTree> getTestSegmentCategoryTrees();
public Set<KlarosCategoryTree> getTestSuiteCategoryTrees();
public Set<KlarosTestSuite> getTestSuites();
public int hashCode();
public Boolean isSecured();
```

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.41. KlarosEnumValue

A Custom property enumeration value.

```
B.1.1.41.1. Synopsis
```

```
public final class KlarosEnumValue extends KlarosWrapper<KlarosEnumValue, EnumValue>
    implements IKlarosEnumValue {
```

// Public Methods
public String getValue();
}

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.42. KlarosExternalServer

This class encapsulates the (dynamic) properties of a klaros object.

B.1.1.42.1. Synopsis

}

public abstract class KlarosExternalServer<T extends KlarosExternalServer<T, S>,S
 extends ExternalServer> extends

```
KlarosLabeledObject<T, S>
```

implements IKlarosExternalServer<T> {

// Public Methods
public String getDescription();
public String getType();
public String getUrl();

Direct known subclasses: de.verit.klaros.core.model.KlarosIssueManagement , de.verit.klaros.core.model.KlarosRequirementsManagement

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.43. KlarosIssue

This class provides access to the information stored for detected issues.

B.1.1.43.1. Synopsis

```
public final class KlarosIssue extends
                                 KlarosNamedEntity<KlarosIssue, Issue>
  implements IKlarosIssue {
 // Public Methods
 public KlarosSUTImplementation getAcceptedIn();
 public Date getCreationDate();
 public String getDescription();
 public String getExternalId();
 public KlarosIssueManagement getIssueManagement();
 public Date getLastSynched();
 public String getOwner();
 public String getPriority();
 public String getRemoteBrowseUrl();
 public String getReporter();
 public String getResolution();
 public String getState();
```

```
public String getSubject();
public Set<KlarosSUTImplementation> getSystemsUnderTest();
public Set<KlarosTestCase> getTestCases();
public boolean isResolved();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.44. KlarosIssueManagement

An issue management system.

B.1.1.44.1. Synopsis

public class KlarosIssueManagement extends

KlarosExternalServer<KlarosIssueManagement,</pre>

IssueManagement>

implements IKlarosIssueManagement {

// Public Methods

```
public Set<KlarosIssue> getIssues();
public String getProject();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosExternalServer: getDescription , getType , getUrl

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.45. KlarosIteration

An iteration in a project.

B.1.1.45.1. Synopsis

```
public Set<KlarosCategoryNode> getCategories();
public StateDef getCurrentState();
public String getDescription();
public Date getDueDate();
public Set<KlarosTestEnvironment> getEnvs();
public Set<KlarosJob> getJobs();
public String getShortname();
public Date getStartDate();
public String getSuccessCriteria();
public Set<KlarosSUTImplementation> getSuts();
public Set<KlarosTestRun> getTestRuns();
public int hashCode();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.46. KlarosJob

This class provides access to the information stored for jobs.

B.1.1.46.1. Synopsis

public final class KlarosJob extends

KlarosNamedEntity<KlarosJob, Job>

// Public Methods public int compareTo(KlarosJob o); public boolean equals(Object o); public Set<KlarosAttachment> getAttachments(); public KlarosConfiguration getConfiguration(); public Set<KlarosJobDependency> getDependentJobs(); public String getDescription(); public Date getDueDate(); public KlarosTestEnvironment getEnv(); public String getEstimatedTime(); public Long getEstimatedTimeInMilliseconds(); public JobPriority getJobPriority(); public JobStatus getJobStatus(); public JobType getJobType(); public KlarosJob getParent(); public Integer getProgress(); public Set<KlarosJobDependency> getRequiredJobs(); public Date getStartDate(); public List<KlarosJob> getSubJobs(); public Integer getSuccessRate(); public String getSummary(); public KlarosSUTImplementation getSut(); public KlarosTestCase getTestCase(); public Set<KlarosTestRun> getTestRuns(); public KlarosTestSuite getTestSuite(); public List<KlarosJobUpdateAction> getUpdateAction(); public List<KlarosJobTimeBlock> getWork(); public int hashCode(); 3

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.47. KlarosJobDependency

This class provides access to the information stored for jobs.

```
B.1.1.47.1. Synopsis
```

```
public final class KlarosJobDependency extends KlarosWrapper<KlarosJobDependency,
    JobDependency>
    implements IKlarosJobDependency,
        Comparable<KlarosJobDependency> {
    // Public Methods
    public int compareTo(KlarosJobDependency o);
    public boolean equals(Object o);
    public Set<JobStatus> getJobStates();
    public UUID getKey();
```

```
public Integer getProgress();
public KlarosJob getRequiredBy();
public KlarosJob getRequires();
public Integer getSuccessRate();
public Set<Verdict> getVerdicts();
public int hashCode();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.48. KlarosJobTimeBlock

The implementation of a job time block, measuring a unit of work done for a job.

B.1.1.48.1. Synopsis

```
public final class KlarosJobTimeBlock extends
                                 KlarosLabeledObject<KlarosJobTimeBlock,</pre>
 JobTimeBlock>
  implements IKlarosJobTimeBlock,
          Comparable<KlarosJobTimeBlock> {
 // Public Methods
 public int compareTo(KlarosJobTimeBlock o);
 public boolean equals(Object o);
 public String getDescription();
 public Date getDoneAt();
 public String getDuration();
 public Long getDurationInMilliseconds();
 public String getEstimatedTimeLeft();
 public Long getEstimatedTimeLeftInMilliseconds();
 public KlarosJob getJob();
 public KlarosTestRun getTestRun();
```

```
public int hashCode();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.49. KlarosJobUpdateAction

The model of a job comment.

B.1.1.49.1. Synopsis

public final class KlarosJobUpdateAction extends

KlarosLabeledObject<KlarosJobUpdateAction,

JobUpdateAction>

```
implements IKlarosJobUpdateAction,
        Comparable<KlarosJobUpdateAction> {
        // Public Methods
        public int compareTo(KlarosJobUpdateAction o);
```

```
public int compareTo(KlarosJobUpdateAction o);
public boolean equals(Object o);
public String getChanges();
public String getDescription();
public int hashCode();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.50. KlarosLabeledObject

This class encapsulates the (dynamic) properties of a klaros object.

B.1.1.50.1. Synopsis

```
public abstract class KlarosLabeledObject<T,S extends LabeledObject>
extends KlarosWrapper<T, S>
    implements IKlarosLabeledObject<T> {
    // Public Methods
    public Iterable<T> asIterable();
    public Date getCreated();
    public KlarosUser getCreator();
    public UUID getKey();
    public KlarosUser getLastEditor();
    public Date getLastUpdated();
    public String getName();
    public boolean isEnabled();
}
```

Direct known subclasses: de.verit.klaros.core.model.KlarosConfiguration , de.verit.klaros.core.model.KlarosExternalServer , de.verit.klaros.core.model.KlarosJobTimeBlock , de.verit.klaros.core.model.KlarosJobUpdateAction , de.verit.klaros.core.model.Klaros-NamedEntity , de.verit.klaros.core.model.KlarosCore.model.KlarosCore.model.KlarosUser

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.51. KlarosNamedEntity

This class encapsulates the (dynamic) properties of a klaros object.

B.1.1.51.1. Synopsis

```
implements IKlarosNamedEntity<T> {
```

```
// Public Methods
public KlarosUser getAssignee();
public List<IKlarosProperty> getProperties();
public String getPropertyValue(String propertyName);
public boolean isDefinedProperty(String propertyName);
}
```

```
Direct known subclasses: de.verit.klaros.core.model.KlarosCategoryTree , de.verit.klaros.core.model.KlarosIssue , de.verit.klaros.core.model.KlarosIteration , de.verit.klaros.core.model.KlarosResult , de.verit.klaros.core.model.KlarosSUTImplementation , de.verit.klaros.core.model.Klaros.core.model.KlarosTestEnvironment , de.verit.klaros.core.model.KlarosTestRun , de.verit.klaros.core.model.KlarosTestStepContainer
```

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.52. KlarosProperty

The Klaros property class.

B.1.1.52.1. Synopsis

```
public final class KlarosProperty extends KlarosWrapper<KlarosProperty, Property>
implements IKlarosProperty {
```

```
// Public Methods
public String getName();
public String getValue();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code


B.1.1.53. KlarosRepositoryEntity

This class encapsulates the (dynamic) properties of a klaros object.

```
B.1.1.53.1. Synopsis
```

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals , getWrapped , hash-Code



B.1.1.54. KlarosRequirement

This class provides access to the information stored for requirements.

B.1.1.54.1. Synopsis

```
public final class KlarosRequirement extends
                                KlarosRevision<KlarosRequirement, Requirement>
  implements IKlarosRequirement<KlarosRequirement, Requirement> {
 // Public Methods
 public Set<KlarosAttachment> getAttachments();
 public Set<KlarosCategoryNode> getCategories();
 public KlarosConfiguration getConfiguration();
 public Set<KlarosTestCase> getCoveringTestCases();
 public String getDescription();
 public Set<String> getExternalNames();
 public String getExternalStatus();
 public Set<KlarosIteration> getIterations();
 public KlarosRequirement getPredecessor();
 public RequirementPriority getPriority();
 public KlarosRequirement getRoot();
 public String getShortname();
```

```
public String getState();
public KlarosRequirement getSuccessor();
public String getSummary();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosRevision: getRevisionComment , get-RevisionId

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



 The external status of this requirement.

Parameters

return

the external status

B.1.1.55. KlarosRequirementsManagement

A requirement management system.

B.1.1.55.1. Synopsis

public class KlarosRequirementsManagement extends

KlarosExternalServer<KlarosRequirementsManagement,

RequirementsManagement>

implements IKlarosRequirementsManagement {

```
// Public Methods
public String getProject();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosExternalServer: getDescription , getType , getUrl

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.56. KlarosResult

This class provides access to the information stored for generic results.

B.1.1.56.1. Synopsis

```
implements IKlarosResult<T> {
    // Public Methods
    public KlarosConfiguration getConfiguration();
    public String getDescription();
    public long getExecutionTime();
    public KlarosJob getJob();
    public String getSummary();
    public String getVerdict();
    public String getVerdict(Locale locale);
    public boolean isError();
    public boolean isFailure();
    public boolean isInconclusive();
    public boolean isPassed();
    public boolean isSkipped();
```

```
}
```

Direct known subclasses: de.verit.klaros.core.model.KlarosTestCaseResult , de.verit.klaros.core.model.KlarosTestCaseStepResult , de.verit.klaros.core.model.KlarosTestSuiteResult

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.57. KlarosRevision

This class encapsulates the revision related information of a klaros object.

B.1.1.57.1. Synopsis

implements IKlarosRevision<T, S> {

// Public Methods

```
public final String getRevisionComment();
public final String getRevisionId();
}
```

Direct known subclasses: de.verit.klaros.core.model.KlarosRequirement , de.verit.klaros.core.model.KlarosTestCase , de.verit.klaros.core.model.KlarosTestSegment , de.verit.klaros.core.model.KlarosTestSuite

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.58. KlarosSUTImplementation

This class provides access to the information stored for systems under tests (SUT).

B.1.1.58.1. Synopsis

}

public final class KlarosSUTImplementation extends

KlarosNamedEntity<KlarosSUTImplementation,

SystemUnderTest>

```
// Public Methods
public int compareTo(KlarosSUTImplementation o);
public boolean equals(Object o);
public Set<KlarosAttachment> getAttachments();
public Set<KlarosCategoryNode> getCategories();
public KlarosConfiguration getConfiguration();
public Set<KlarosIssue> getIssues();
public Set<KlarosIteration> getIterations();
public String getProductversion();
public Set<KlarosTestRun> getTestRuns();
public int hashCode();
```

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.59. KlarosTestCase

This class provides access to the information stored for a test case.

B.1.1.59.1. Synopsis

public final class KlarosTestCase extends

KlarosRevision<KlarosTestCase, TestCase>

// Public Methods public int compareTo(KlarosTestCase o); public boolean equals(Object o); public Set<KlarosAttachment> getAttachments(); public Set<KlarosCategoryNode> getCategories(); public KlarosConfiguration getConfiguration(); public Set<KlarosRequirement> getCovers(); public String getDependency(); public String getDescription(); public TestDesignTechnique getDesignTechnique(); public List<KlarosIssue> getDetectedIssues(); public String getDocbase(); public Long getEstimatedDuration(); public TestEvaluationMethod getEvaluation(); public TestExecutionMethod getExecution(); public String getExpectedResult(); public Set<String> getExternalNames(); public List<KlarosJob> getJobs(); public TestLevel getLevel(); public String getNote(); public String getPostcondition();

```
public String getPrecondition();
public KlarosTestCase getPredecessor();
public TestPriority getPriority();
public Set<KlarosTestCaseResult> getResults();
public KlarosTestCase getRoot();
public String getShortname();
public String getState();
public KlarosTestCase getSuccessor();
public KlarosTestCase getSuccessor();
public List<KlarosTestCaseStep> getTestCaseSteps();
public TestAreatopic getTestType();
public String getTraceability();
public TestVariety getVariety();
public int hashCode();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosRevision: getRevisionComment , get-RevisionId

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.60. KlarosTestCaseResult

This class provides access to the information stored for test case results.

B.1.1.60.1. Synopsis

```
// Public Methods
public int compareTo(KlarosTestCaseResult o);
public boolean equals(Object o);
public Set<KlarosAttachment> getAttachments();
public List<KlarosTestCaseStepResult> getStepResults();
public KlarosTestCase getTestCase();
public KlarosTestRun getTestRun();
public Integer getTestSuitePosition();
public KlarosTestSuiteResult getTestSuiteResult();
public int hashCode();
public boolean isPending();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosResult: getConfiguration , get-Description , getExecutionTime , getJob , getSummary , getVerdict , isError , isFailure , isInconclusive , isPassed , isSkipped

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.61. KlarosTestCaseStep

This class provides access to the information stored for test case steps.

B.1.1.61.1. Synopsis

```
public final class KlarosTestCaseStep extends
```

KlarosTestStepContainer<KlarosTestCaseStep,</pre>

TestStep>

}

implements IKlarosTestCaseStep<KlarosTestCaseStep> {

```
// Public Methods
public String getAction();
public Set<KlarosAttachment> getAttachments();
public String getExpectedResult();
public String getPostcondition();
public String getPrecondition();
```

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.62. KlarosTestCaseStepResult

This class provides access to the information stored for test case step results.

B.1.1.62.1. Synopsis

```
public final class KlarosTestCaseStepResult extends
                                 KlarosResult<KlarosTestCaseStepResult,
 TestCaseStepResult>
  implements IKlarosTestCaseStepResult,
          Comparable<KlarosTestCaseStepResult> {
 // Public Methods
 public int compareTo(KlarosTestCaseStepResult o);
 public boolean equals(Object o);
 public String getAction();
 public Set<KlarosAttachment> getAttachments();
 public String getExpectedResult();
 public String getPostcondition();
 public String getPrecondition();
 public KlarosTestCase getTestCase();
 public KlarosTestCaseResult getTestCaseResult();
 public KlarosTestRun getTestRun();
 public int hashCode();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosResult: getConfiguration , get-Description , getExecutionTime , getJob , getSummary , getVerdict , isError , isFailure , isInconclusive , isPassed , isSkipped

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.63. KlarosTestEnvironment

This class provides access to the information stored for test environments.

B.1.1.63.1. Synopsis

// Public Methods

```
public int compareTo(KlarosTestEnvironment o);
public boolean equals(Object o);
public Set<KlarosAttachment> getAttachments();
public Set<KlarosCategoryNode> getCategories();
public String getDescription();
public Set<KlarosIteration> getIterations();
public Set<KlarosTestRun> getTestRuns();
public int hashCode();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.64. KlarosTestExecutable

This class provides access to the information stored for test cases.

B.1.1.64.1. Synopsis

```
public final class KlarosTestExecutable extends KlarosWrapper<KlarosTestExecutable,
TestExecutable>
```

```
implements IKlarosTestExecutable {
```

```
}
```

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.65. KlarosTestRun

This class provides access to the information stored for test runs.

B.1.1.65.1. Synopsis

```
public final class KlarosTestRun extends
                                 KlarosNamedEntity<KlarosTestRun, TestRun>
  implements IKlarosTestRun,
          Comparable<KlarosTestRun> {
 // Public Methods
 public int compareTo(KlarosTestRun o);
 public boolean equals(Object o);
 public KlarosConfiguration getConfiguration();
 public KlarosTestEnvironment getEnv();
 public KlarosIteration getIteration();
 public KlarosJob getJob();
 public int getNumberErrors();
 public int getNumberFailures();
 public int getNumberInconclusive();
 public int getNumberPassed();
 public int getNumberSkipped();
 public String getRelatedSummary();
 public Set<KlarosTestCaseResult> getResults();
 public String getRunId();
 public KlarosSUTImplementation getSut();
 public KlarosTestSuite getTestSuite();
 public Date getTimestamp();
 public Set<KlarosJobTimeBlock> getWork();
 public int hashCode();
 public boolean isPending();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty **Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject**: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.66. KlarosTestSegment

This class provides access to the information stored for test segments.

```
B.1.1.66.1. Synopsis
```

```
public String getPostcondition();
public String getPrecondition();
public KlarosRevision<KlarosTestSegment, TestSegment> getPredecessor();
public KlarosRevision<KlarosTestSegment, TestSegment> getRoot();
public String getShortname();
public List<KlarosTestCaseStep> getSteps();
public KlarosRevision<KlarosTestSegment, TestSegment> getSuccessor();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosRevision: getRevisionComment , get-RevisionId

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code





This class encapsulates the (dynamic) properties of a klaros object.

B.1.1.67.1. Synopsis

}

```
implements IKlarosTestStepContainer<T> {
```

Direct known subclasses: de.verit.klaros.core.model.KlarosTestCaseStep

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.68. KlarosTestSuite

This class provides access to the information stored for test suites.

B.1.1.68.1. Synopsis

implements IKlarosTestSuite<KlarosTestSuite, TestSuite> {

```
// Public Methods
public Set<KlarosAttachment> getAttachments();
public Set<KlarosCategoryNode> getCategories();
public KlarosConfiguration getConfiguration();
public KlarosTestSuite getPredecessor();
public List<KlarosTestSuiteResult> getResults();
public KlarosTestSuite getRoot();
public String getShortname();
public KlarosTestSuite getSuccessor();
public KlarosSUTImplementation getSut();
public List<KlarosTestCase> getTestCases();
public Integer getTestSuiteResultCount();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosRevision: getRevisionComment , get-RevisionId

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code



B.1.1.69. KlarosTestSuiteResult

This class provides access to the information stored for test suite results.

B.1.1.69.1. Synopsis

```
public boolean isPending();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosResult: getConfiguration , get-Description , getExecutionTime , getJob , getSummary , getVerdict , isError , isFailure , isInconclusive , isPassed , isSkipped

Methods inherited from de.verit.klaros.core.model.KlarosNamedEntity: getAssignee , get-Properties , getPropertyValue , isDefinedProperty

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.1.70. KlarosUser

The user object.

B.1.1.70.1. Synopsis

public class KlarosUser extends

KlarosLabeledObject<KlarosUser, User>

```
// Public Methods
public int compareTo(KlarosUser o);
public boolean equals(Object o);
public String getEmail();
public UserRole getRole();
public String getUsername();
public int hashCode();
public String toString();
}
```

Methods inherited from de.verit.klaros.core.model.KlarosLabeledObject: asIterable , get-Created , getCreator , getKey , getLastEditor , getLastUpdated , getName , isEnabled

Methods inherited from de.verit.klaros.core.model.KlarosWrapper: equals, getWrapped, hash-Code

Methods inherited from java.lang.Object: getClass, notify, notifyAll, toString, wait



B.1.2. Deprecated API

B.1.2.1. Deprecated Methods

de.verit.klaros.core.model.IK- use getSystemsUnderTest laroslssue.getAcceptedIn()

B.2. Scripting API Reference

B.2.1. de.verit.klaros.scripting

B.2.1.1. IKlarosQueryFactory

A factory for executing Klaros queries.

B.2.1.1.1. Synopsis

```
public interface IKlarosQueryFactory extends Serializable {
```

```
// Public Methods
 public abstract List<?> execute(String query);
 public abstract List<?> execute(String query, ParameterContext params);
}
```



B.2.1.1.2. execute(String)

public abstract List<?> execute(String query);

Prepares the given query string with our KlarosOdaQuery and returns a KlarosList with the results of the query.

	Parameters	
query	the query	
return	the list of query results	
B.2.1.1.3. execute(String, ParameterContext)		

```
public abstract List<?> execute(String query, ParameterContext params);
```

Prepares the given query string with our KlarosOdaQuery and returns a KlarosList with the results of the query.

Parameters

query

the query

params

the parameters

return

the list

B.2.1.2. KlarosContext

Context to provide all methods to the user to add own objects to the event context.

B.2.1.2.1. Synopsis

public class KlarosContext implements IKlarosContext {

```
// Public Constructors
 public KlarosContext(Context context,
                   UUID authenticatedUserId,
                   UUID activeProjectId,
                   UUID activeIterationId,
                   String applicationUrl,
                   IKlarosQueryFactory factory,
                   ParameterContext parameters);
 public KlarosContext(Context context,
                   UUID authenticatedUserId,
                   UUID activeProjectId,
                   UUID activeIterationId,
                   String applicationUrl,
                   IKlarosQueryFactory factory,
                   ParameterContext parameters,
                   Locale locale);
 // Public Methods
 public void add(String name, Object value);
 public boolean containsParameter(String name);
 public List<?> executeParameterizedQuery(String query);
 public List<?> executeQuery(String query);
 public KlarosIteration getActiveIteration();
 public KlarosConfiguration getActiveProject();
 public String getAttachmentUrl(String attachmentId);
 public String getAttachmentUrl(String attachmentId, String version);
 public String getAttachmentUrl(UUID attachmentId);
 public String getAttachmentUrl(UUID attachmentId, String version);
 public String getBrowsePageUrl(KlarosLabeledObject<?, ?> artifact);
 public Locale getLocale();
 public Parameter getParameter(String name);
 public Object getParameterValue(String name);
 public String getPrintPageUrl(KlarosLabeledObject<?, ?> artifact);
 public void setLocale(Locale locale);
}
```

Methods inherited from java.lang.Object: equals,getClass,hashCode,notify,notifyAll,to-String,wait



B.2.1.2.2. KlarosContext(Context, UUID, UUID, UUID, String, IKlarosQueryFactory, Parameter-Context)

public KlarosContext(Context context,

UUID authenticatedUserId, UUID activeProjectId, UUID activeIterationId, String applicationUrl, IKlarosQueryFactory factory, ParameterContext parameters);

Create a KlarosContext.

Parameters

context

Gets passed by the relating servlet.

authenticatedUserId

the authenticated user id

activeProjectId	the active project id
activeIterationId	the active iteration id if available
applicationUrl	the application url
factory	the factory
parameters	the parameters

B.2.1.2.3. KlarosContext(Context, UUID, UUID, UUID, String, IKlarosQueryFactory, Parameter-Context, Locale)

Create a KlarosContext.

Parameters

context	Gets passed by the relating servlet.
authenticatedUserId	the authenticated user id
activeProjectId	the active project id
activeIterationId	the active iteration id
applicationUrl	the application url
factory	the factory
parameters	the parameters
locale	the locale

B.2.2. de.verit.klaros.scripting.context

B.2.2.1. IKlarosContext

The context object encapsulates the input/output interface a report script can access. <orderedlist> <listitem>

The parameters given before starting the report. </listitem> An HQL query interface for retrieving model objects from the database. </listitem> <listitem>

A writable context for storing values for template access. </listitem> <listitem>

Access to current Klaros settings (current project, current iteration, locale settings) </listitem>

URL conversions for generating URLs to attachments, page and print page objects inside the application. </listitem> </orderedlist>

</orderedlist>

The following entries are available

B.2.2.1.1. Synopsis

public interface IKlarosContext {

```
// Public Methods
public abstract void add(String name, Object value);
public abstract boolean containsParameter(String parameterName);
public abstract List<?> executeParameterizedQuery(String query);
public abstract List<?> executeQuery(String query);
public abstract KlarosIteration getActiveIteration();
public abstract KlarosConfiguration getActiveProject();
public abstract String getAttachmentUrl(String attachmentId);
public abstract String getAttachmentUrl(String attachmentId, String revision);
public abstract String getAttachmentUrl(UUID attachmentId);
public abstract String getAttachmentUrl(UUID attachmentId, String revision);
public abstract String getBrowsePageUrl(KlarosLabeledObject<?, ?> artifact);
public abstract Locale getLocale();
public abstract Parameter getParameter(String parameterName);
public abstract Object getParameterValue(String name);
public abstract String getPrintPageUrl(KlarosLabeledObject<?, ?> artifact);
public abstract void setLocale(Locale locale);
```

«interface» IKlarosContext
 + add(name: String, value: Object): void + getActiveProject(): KlarosConfiguration + getActiveIteration(): KlarosIteration + executeQuery(query: String): List<? > + executeParameterizedQuery(query: String): List<? > + containsParameter(parameterName: String): boolean + getParameter(parameterName: String): Parameter + getParameterValue(name: String): Object + getLocale(): Locale + setLocale(locale: Locale): void + getAttachmentUrl(attachmentld: UUID): String + getAttachmentUrl(attachmentld: String): String + getAttachmentUrl(attachmentld: String): String + getAttachmentUrl(attachmentld: String, revision: String): String + getBrowsePageUrl(artifact: KlarosLabeledObject<?, ?>): String + getPrintPageUrl(artifact: KlarosLabeledObject<?, ?>): String

B.2.2.1.2. add(String, Object)

public abstract void add(String name, Object value);

Add a new object with the give key to the event context.

Parameters

name

the name

value

the value

B.2.2.1.3. containsParameter(String)

public abstract boolean containsParameter(String parameterName);

Checks if the parameter is available in the context.

Parameters

the parameter name parameterName

return

true, if successful

B.2.2.1.4. executeParameterizedQuery(String)

public abstract List<?> executeParameterizedQuery(String query);

Execute the given query with the Parameters from the ParameterContext.

Parameters

query

The HQL query to execute.

	return	A KlarosList of the selected objects.	
B.2.2	<pre>B.2.2.1.5. executeQuery(String) public abstract List<?> executeQuery(String query);</pre>		
	Execute the given query.		
		Parameters	
	query	The HQL query to execute.	
	return	A KlarosList of the selected objects.	
B.2.2.1.6. getActiveIteration() public abstract KlarosIteration getActiveIteration();			
	Gets the active iteration.		
		Parameters	
	return	the active iteration	
<pre>B.2.2.1.7. getActiveProject() public abstract KlarosConfiguration getActiveProject();</pre>			
	Gets the active project.		
		Parameters	
	return	the active project	
B.2.2.1.8. getAttachmentUrl(String) public abstract String getAttachmentUrl(String attachmentId);			
	Return an URL string to retrieve the attachment with given attachment id.		
		Parameters	
	attachmentId	the attachment id	
	return	the attachment url	
B.2.2.1.9. getAttachmentUrl(String, String) public abstract String getAttachmentUrl(String attachmentId, String revision);			
	Return an URL string to retrieve the attachment with given attachment id string and revision.		
		Parameters	
	attachmentId	the attachment id	
	revision	the revision	
	return	the attachment url	

B.2.2.1.10. getAttachmentUrl(UUID)

public abstract String getAttachmentUrl(UUID attachmentId);

Return an URL string to retrieve the attachment with given attachment UUID.

Parameters

attachmentId the attachment id

return the attachment url

B.2.2.1.11. getAttachmentUrl(UUID, String)

public abstract String getAttachmentUrl(UUID attachmentId, String revision);

Return an URL string to retrieve the attachment with given attachment UUID and revision.

	Parameters
attachmentId	the attachment id
revision	the revision string
return	the attachment url

B.2.2.1.12. getBrowsePageUrl(KlarosLabeledObject<?, ?>)

public abstract String getBrowsePageUrl(KlarosLabeledObject<?, ?> artifact);

Gets the browse page url to the given klaros artifact. Will return null if the artifact type is not supported.

Parameters

artifact	the artifact

return

the browse page url

B.2.2.1.13. getLocale()

public abstract Locale getLocale();

/** Returns the active Java Locale setting.

Parameters

return	
i c cui ii	

the Java Locale

B.2.2.1.14. getParameter(String)

public abstract Parameter getParameter(String parameterName);

Gets the parameter.

Parameters

parameterName	the parameter name
return	The parameter with the given name
	Exceptions
Klaros- Illegal-	if the parameter is not available.

Argument-Exception

Gets the parameter value.

Parameters

name

the name

return

The parameter with the given name

if the parameter is not available.

Exceptions

Klaros-Illegal-Argument-Exception

B.2.2.1.16. getPrintPageUrl(KlarosLabeledObject<?, ?>)

public abstract String getPrintPageUrl(KlarosLabeledObject<?, ?> artifact);

Gets the print page of the given klaros artifact. Will return null if the artifact type is not supported.

Parameters

artifact

the artifact

return

the print page url

B.2.2.1.17. setLocale(Locale)

public abstract void setLocale(Locale locale);

Sets the active Java Locale setting.

Parameters

locale

the new locale

B.2.3. de.verit.klaros.scripting.model

B.2.3.1. Parameter

The Parameter class encapsulates a report query parameter.

B.2.3.1.1. Synopsis

public class Parameter implements Serializable {

// Public Constructors
public Parameter(ScriptParameter source);

// Public Methods
public String getDefaultValue();

```
public List<String> getDefaultValues();
public String getDescription();
public String getLabel();
public String getName();
public List<String> getOptions();
public String getOptionsString();
public ScriptParameterType getType();
public String getValue();
public List<String> getValues();
public List<String> getValues();
public boolean isMandatory();
}
```

Methods inherited from java.lang.Object: equals,getClass,hashCode,notify,notifyAll,to-String,wait



B.2.3.1.2. Parameter(ScriptParameter)

public Parameter(ScriptParameter source);

Instantiates a new parameter.

Parameters

source

the source script parameter

Gets the default value.

		Parameters
	return	the default value
B.2.3	.1.4. getDefaultValues() public List <string> getDefaul</string>	tValues();
	Gets the default values.	
		Parameters
	return	the default values
B.2.3	.1.5. getDescription() public String getDescription();
	Gets the description.	
		Parameters
	return	the description
B.2.3	.1.6.getLabel() <pre>public String getLabel();</pre>	
	Gets the label.	
		Parameters
	return	the label
B.2.3	.1.7.getName() public String getName();	
	Gets the name.	
		Parameters
	return	the name
B.2.3	.1.8.getOptions() public List <string> getOption</string>	s();
	Gets the options.	
		Parameters
	return	the options
B.2.3	.1.9.getOptionsString() public String getOptionsStrin	g();
	Gets the options string.	
	Gets the options string.	Parameters

B.2.3.1.10.getType() public ScriptParameterType g	getType();
Gets the type.	
	Parameters
return	the type
B.2.3.1.11.getValue() public String getValue();	
Gets the value.	
	Parameters
return	the value
B.2.3.1.12.getValues() public List <string> getValue</string>	es();
Gets the value.	
	Parameters
return	the value
B.2.3.1.13. isMandatory() public boolean isMandatory()	;
Checks if is mandatory.	
	Parameters
return	true, if is mandatory

return
Appendix C. Test Case Import File Specification

Name	Description
<areatopic></areatopic>	(Test Type) A group of test activities based on specific test objectives with the purpose of testing a component or system for specific charac- teristics (FUNCTIONAL, NON_FUNCTIONAL, STRUCTURAL, REGRESSIO- N or RE_TEST).
<attachment></attachment>	An attachment.
<attachments></attachments>	The list of attachments.
<attribute></attribute>	A custom field.
<attributes></attributes>	The list of user-defined custom fields.
<category></category>	A node of a category tree.
< <u>categoryDescrip-</u> tion>	The description of the category.
<categoryname></categoryname>	The name of the category.
<categorytree></categorytree>	A category tree.
<container></container>	The root element of this schema, which contains all the test cases and optionally a test suite/category tree.
<content></content>	The base64 encoded content of the attachment.
<depends></depends>	The dependencies of the test case in text form.
<description></description>	The textual description of the test case.
<docbase></docbase>	The reference document containing the requirement on which the test case is based.
<estimatedduration></estimatedduration>	The estimated duration in milli seconds.
<evaluation></evaluation>	The evaluation method of this test case (MANUAL or AUTO).
<execution></execution>	The execution type of this test case (MANUAL or AUTO).
<expectedresult></expectedresult>	The expected result.
<externalid></externalid>	The external id of this test case.
<level></level>	The test level of this test case (COMPONENT, INTEGRATION, SYSTEM or ACCEPTANCE).
<method></method>	(Design Technique) The test design view on the system under test (BLA-CK_BOX or WHITE_BOX).
<u><name></name></u>	The name of the user-defined custom field.
<note></note>	A placeholder for any short notes about the test case.
<postcondition></postcondition>	A text describing the postcondition.
<precondition></precondition>	A text describing the precondition.
<priority></priority>	The priority of this test case (LOW, MEDIUM or HIGH).
<revision></revision>	The revision of this test case.

The following table gives a short overview of the available elements:

Name	Description
<shortname></shortname>	The name of the test case.
<state></state>	The test case state (not used, reserved for future use).
<step></step>	A test case step.
<steps></steps>	The list of test case steps.
<team></team>	The team that is responsible for this test case.
<testcase></testcase>	A test case.
<testcases></testcases>	The list of test cases.
<testsuite></testsuite>	If this element is present, a test suite with the name given here will be generated upon import.
<traceability></traceability>	A reference to the associated requirement, use case, or work package.
<value></value>	The value of the user-defined custom field.
<variety></variety>	The result expectation of the test (POSITIVE or NEGATIVE).

Table C.1. Element summary

C.1. <step>

A test case step.

This element may occur inside the following elements: <steps> .

Name	Multiplicity	Description
<pre><description></description></pre>	01	The action to take in this test case step.
<expectedresult></expectedresult>	01	The expected result.
<postcondition></postcondition>	01	The postcondition.
<precondition></precondition>	01	The precondition.

Table C.2. <step> elements

C.2. <steps>

The list of test case steps.

This element may occur inside the following elements: <a><testcase>

Name	Multiplicity	Description
<step></step>	0n	A test step.

Table C.3. <steps> elements

C.3. <attachments>

The list of attachments.

This element may occur inside the following elements: <a> .

Name	Multiplicity	Description
<attachment></attachment>	0n	A binary attachment.

Table C.4. <attachments> elements

C.4. <attachment>

An attachment.

This element may occur inside the following elements: sattachments .

Name	Multiplicity	Description
<u><name></name></u>	11	The name of the attachment.
<content></content>	11	The attachment binary content, encoded in - Base-64 format.

Table C.5. <attachment> elements

C.5. <attribute>

A custom field.

This element may occur inside the following elements: <a href="https://www.attributes-complexity-co

Name	Multiplicity	Description
<u><name></name></u>	11	The name of the custom field.
<value></value>	11	The value of the custom field.

Table C.6. <attribute> elements

C.6. <attributes>

The list of user-defined custom fields.

This element may occur inside the following elements: <a><testcase>

Name	Multiplicity	Description
<attribute></attribute>	0n	

Table C.7. <attributes> elements

C.7. <category>

A node of a category tree. It can contain further subcategories.

Name	Multiplicity	Description
<categoryname></categoryname>	11	The name of the category node.
<categorydescription></categorydescription>	11	The description of the category node.
<category></category>	01	The parent category node.

Table C.8. <category> elements

C.8. <categoryTree>

A category tree. It contains the root category node with an arbitrary nomber of sub categories.

This element may occur inside the following elements: <a> <a> <a> <a> <a> <a> <a> <a> <a><

Name	Multiplicity	Description
<categoryname></categoryname>	11	The name of the category tree.
<category></category>	11	The root category tree node.

Table C.9. <categoryTree> elements

C.9. <container>

The root element of this schema, which contains all the test cases and optionally a test suite/ category tree.

Name	Multiplicity	Description
<u><testsuite></testsuite></u>	01	An optional test suite containing the listed test cases.
<testcases></testcases>	01	The list of test cases.
< <u>categoryTree></u>	01	The optional category tree of these require- ments.

Table C.10. <container> elements

C.10. <testcases>

The list of test cases.

This element may occur inside the following elements: <a> <a> <a> <a> <a> <a> <a> <a> <a><

Name	Multiplicity	Description
<testcase></testcase>	0n	A test case.

Table C.11. <testcases> elements

C.11. <testcase>

A test case.

Name	Multiplicity	Description
<areatopic></areatopic>	01	The test case type (FUNCTIONAL, NON_FUNC- TIONAL, STRUCTURAL, REGRESSION or RE_T- EST).
<attachments></attachments>	01	The list of binary attachments.
<attributes></attributes>	01	The custom fields this test case has.
<category></category>	01	The category node to which this test case is assigned.
<depends></depends>	01	The dependencies of the test case in text form.
<description></description>	01	The textual description of the test case.
<pre><docbase></docbase></pre>	01	The reference document containing the re- quirement on which the test case is based.
<pre><estimatedduration></estimatedduration></pre>	01	The estimated duration in milli seconds for executing this test case.

Name	Multiplicity	Description
<evaluation></evaluation>	01	The evaluation method of this test case (MAN-UAL or AUTO).
<execution></execution>	01	The execution type of this test case (MANUAL or AUTO).
<expectedresult></expectedresult>	01	The expected result.
<externalid></externalid>	01	The external id of this test case. This is re- quired when referencing test cases in require- ment imports.
<level></level>	01	The test level of this test case (COMPONENT, - INTEGRATION, SYSTEM or ACCEPTANCE).
<method></method>	01	(Design Technique) The test design view on the system under test (BLACK_BOX or WHIT- E_BOX).
<note></note>	01	Any short note about the test case.
<postcondition></postcondition>	01	The postcondition.
<precondition></precondition>	01	The precondition.
<priority></priority>	01	The priority of this test case (LOW, MEDIUM or HIGH).
<revision></revision>	01	The revision of this test case.
<shortname></shortname>	01	The name of the test case.
<u><state></state></u>	01	The test case state (not used, reserved for fu- ture use).
<steps></steps>	01	The list of test case steps.
<team></team>	01	The team that is responsible for this test case.
<traceability></traceability>	01	A reference to the associated requirement, use case, or work package.
<variety></variety>	01	The result expectation of the test (POSITIVE or NEGATIVE).

Table C.12. <testcase> elements

C.12. <testsuite>

If this element is present, a test suite with the name given here will be generated upon import. This test suite will contain all the test cases that are part of this import container.

This element may occur inside the following elements: <a>container> .

C.13. <externalId>

The external id of this test case. This is required when referencing test cases in requirement imports.

This element may occur inside the following elements: <a><testcase>

C.14. <areatopic>

(Test Type) A group of test activities based on specific test objectives with the purpose of testing a component or system for specific characteristics (FUNCTIONAL, NON_FUNCTIONAL, STRUCTURAL, REGRESSION or RE_TEST).

This element may occur inside the following elements: <a><testcase>

C.15. <depends>

The dependencies of the test case in text form.

This element may occur inside the following elements: <a> .

C.16. <description>

The textual description of the test case.

This element may occur inside the following elements: <a>step>, <a>testcase>.

C.17. <docbase>

The reference document containing the requirement on which the test case is based.

C.18. <estimatedDuration>

The estimated duration in milli seconds.

This element may occur inside the following elements: <a> .

C.19. <evaluation>

The evaluation method of this test case (MANUAL or AUTO).

This element may occur inside the following elements: <a><testcase>

C.20. <execution>

The execution type of this test case (MANUAL or AUTO).

C.21. <expectedResult>

The expected result.

This element may occur inside the following elements: <a>step>, <a>testcase>.

C.22. <level>

The test level of this test case (COMPONENT, INTEGRATION, SYSTEM or ACCEPTANCE).

C.23. <method>

(Design Technique) The test design view on the system under test (BLACK_BOX or WHITE_BOX).

This element may occur inside the following elements: <a><testcase>

C.24. <note>

A placeholder for any short notes about the test case.

This element may occur inside the following elements: <a> .

C.25. <postcondition>

A text describing the postcondition.

This element may occur inside the following elements: <a>step>, <a>testcase>.

C.26. <precondition>

A text describing the precondition.

This element may occur inside the following elements: <a>step>, <a>testcase>.

C.27. <revision>

The revision of this test case. This element is used only when exporting test cases, imported test cases always get the fixed revision Id "1.0".

This element may occur inside the following elements: <a><testcase>

C.28. <priority>

The priority of this test case (LOW, MEDIUM or HIGH).

C.29. <shortname>

The name of the test case.

This element may occur inside the following elements: <a><testcase>

C.30. <state>

The test case state (not used, reserved for future use).

This element may occur inside the following elements: <a><testcase>

C.31. <team>

The team that is responsible for this test case.

This element may occur inside the following elements: <a> .

C.32. <traceability>

A reference to the associated requirement, use case, or work package.

This element may occur inside the following elements: <a><testcase>

C.33. <variety>

The result expectation of the test (POSITIVE or NEGATIVE).

C.34. <name>

The name of the user-defined custom field.

This element may occur inside the following elements: <a href="https://www.attributes-cattachments-cattachmen

C.35. <value>

The value of the user-defined custom field.

This element may occur inside the following elements: sattributes .

C.36. <content>

The base64 encoded content of the attachment.

This element may occur inside the following elements: sattachment> .

C.37. <categoryName>

The name of the category.

C.38. <categoryDescription>

The description of the category.

This element may occur inside the following elements: <a> <a> <a> <a> <a> <a> <a> <a> <a><

Appendix D. Requirement Import File Specification

Name	Description
<attachment></attachment>	An attachment.
<attachments></attachments>	The list of binary attachments.
<attribute></attribute>	A custom field.
<attributes></attributes>	The list of custom fields.
<category></category>	A node of a category tree.
< <u>categoryDescrip-</u> tion>	The description of the category.
<categoryname></categoryname>	The name of the category.
<categorytree></categorytree>	A category tree.
<container></container>	The root element of this schema, which contains all requirements.
<content></content>	The base64 encoded content of the attachment.
<description></description>	The textual description of the requirement.
<externalid></externalid>	The external id of this requirement, the format of which can be freely chosen.
<externalrevision></externalrevision>	The external revision of this requirement (not used, reserved for future use).
< <u>externalTest-</u> Caselds>	The list of external test case ids covering this requirement.
<name></name>	The name of a field.
<priority></priority>	The priority of this requirement (LOW, MEDIUM or HIGH).
<requirement></requirement>	A requirement.
<requirements></requirements>	The list of requirements.
<revision></revision>	The revision of this requirement.
<shortname></shortname>	The name of the requirement.
<summary></summary>	A short summary of the requirement.
<value></value>	The value of a field.

The following table gives a short overview of the available elements:

Table D.1. Element summary

D.1. <attachments>

The list of binary attachments.

This element may occur inside the following elements: <a>

<u><requirement></u> .

Name	Multiplicity	Description
<attachment></attachment>	0n	An attachment.

Table D.2. <attachments> elements

D.2. <attachment>

An attachment.

This element may occur inside the following elements: sattachments .

Name	Multiplicity	Description
<name></name>	11	The name of the attachment.
<content></content>	11	The attachment binary content, encoded in - Base-64 format.

Table D.3. <attachment> elements

D.3. <attributes>

The list of custom fields.

This element may occur inside the following elements: <a>

<u><requirement></u> .

Name	Multiplicity	Description
<u><attribute></attribute></u>	0n	A custom field.

Table D.4. <attributes> elements

D.4. <attribute>

A custom field.

This element may occur inside the following elements: $\underline{\langle attributes \rangle}$.

Name	Multiplicity	Description
<u><name></name></u>	11	The name of the custom field.
<value></value>	11	The value of the custom field.

Table D.5. <attribute> elements

D.5. <category>

A node of a category tree. It can contain further subcategories.

This element may occur inside the following elements: <a href="https://www.scategory-category

Name	Multiplicity	Description
<categoryname></categoryname>	11	The name of the category node.
<categorydescription></categorydescription>	11	The description of the category node.
<category></category>	01	The parent category node.

Table D.6. <category> elements

D.6. <categoryTree>

A category tree. It contains the root category node with an arbitrary nomber of sub categories.

This element may occur inside the following elements: <a> <a> <a> <a> <a> <a> <a> <a> <a><

Name	Multiplicity	Description
<categoryname></categoryname>	11	The name of the category tree.
<category></category>	11	The root category tree node.

Table D.7. <categoryTree> elements

D.7. <container>

The root element of this schema, which contains all requirements.

Name	Multiplicity	Description
<requirements></requirements>	01	The list of requirements.
<categorytree></categorytree>	01	The optional category tree of these require- ments.

Table D.8. <container> elements

D.8. <requirements>

The list of requirements.

This element may occur inside the following elements: <a> .

Name	Multiplicity	Description
<requirement></requirement>	0n	A requirement.

Table D.9. <requirements> elements

D.9. <requirement>

A requirement.

This element may occur inside the following elements: <a> <a> <a> <a> <a> <a> <a> <a> <a><

Name	Multiplicity	Description
<attributes></attributes>	01	The custom fields this requirement has.
<category></category>	01	The category node to which this requirement is assigned.
<pre><description></description></pre>	01	The detailed description of the requirement in text form.
<externalid></externalid>	01	The external id of this requirement.
<priority></priority>	01	The priority of this requirement.
<revision></revision>	01	The revision of this requirement.
<u><shortname></shortname></u>	01	The name of the requirement.
<u><summary></summary></u>	01	A short summary of the requirement.
<attachments></attachments>	01	The list of binary attachments.
<externaltestcaseids></externaltestcaseids>	01	The list of external test case ids covering this requirement.

Table D.10. <requirement> elements

D.10. <externalTestCaseIds>

The list of external test case ids covering this requirement.

This element may occur inside the following elements: <a>

<u><requirement></u> .

Name	Multiplicity	Description
<externalid></externalid>	0n	An id of an external test case covering the re- quirement.

Table D.11. <externalTestCaseIds> elements

D.11. <externalId>

The external id of this requirement, the format of which can be freely chosen. This element is used only for requirement synchronization and is ignored during import. If the id of a requirement is changed before a synchronization, a new requirement with this id will be created in Klaros-Testmanagement. Otherwise, the existing requirement is overwritten with the data from this XML file.

This element may occur inside the following elements: <a>

<u><requirement></u> .

D.12. <externalRevision>

The external revision of this requirement (not used, reserved for future use).

D.13. <description>

The textual description of the requirement.

This element may occur inside the following elements: $\underline{\langle requirement \rangle}$.

D.14. <priority>

The priority of this requirement (LOW, MEDIUM or HIGH).

This element may occur inside the following elements: <a>

<u><requirement></u> .

D.15. <revision>

The revision of this requirement. This element is used only when exporting and synchronising requirements, imported requirements always get the fixed revision Id "1.0".

This element may occur inside the following elements: <a>

<u><requirement></u> .

D.16. <shortname>

The name of the requirement.

This element may occur inside the following elements: <a>

<u><requirement></u> .

D.17. <summary>

A short summary of the requirement.

This element may occur inside the following elements: <a>

<u><requirement></u> .

D.18. <name>

The name of a field.

This element may occur inside the following elements: <a href="https://www.attributes-cattachments-cattachmen

D.19. <value>

The value of a field.

This element may occur inside the following elements: sattributes .

D.20. <content>

The base64 encoded content of the attachment.

This element may occur inside the following elements: <a href="mailto:sattachments-

D.21. <categoryName>

The name of the category.

D.22. <categoryDescription>

The description of the category.

This element may occur inside the following elements: $\underline{<\!category\!>}$.

Appendix E. The Reporting Context API

E.1. The Klaros Report Context

Name	Description	Тур
date	The time of report execu- tion.	java.util.Date
locale	The localization selected by the user.	java.util.Locale
activeProject	The currently selected project or null if no project is selected.	de.verit.klaros.model.KlarosConfiguration
activelteration	The currently selected itera- tion or null if no iteration is selected.	de.verit.klaros.model.KlarosIteration
user	The current user who cre- ates the report.	de.verit.klaros.model.KlarosUser
parameters	A map of parameter names to parameter objects con- taining the user defined pa- rameters entered when run- ning the report.	java.util.Map java.lang.String,de.verit. klaros.scripting.model.Parameter

Table E.1. Context Variables

The context variables can be accessed via SeamPDF by e.g.:

<p:text value="#{date}" />



Note

#{user.name} und #{user.username} contain different values. The first provides the user's real name, while the latter provides the name the user is logged in with.

E.2. KlarosScript Interface

```
package de.verit.klaros.scripting;
/**
 * Public interface for generating user defined reports.
 */
public interface KlarosScript {
    /**
    * This method starts the report generation.
    *
    * @param context
    * The event context to provide all needed functions, properties
    * and objects.
    * @throws KlarosScriptingException if executing the script fails
```

```
*/
void execute(KlarosContext context) throws KlarosScriptingException;
```

E.3. Example Layout Template

}

```
<p:document xmlns:ui="http://java.sun.com/jsf/facelets"
 xmlns:f="http://java.sun.com/jsf/core" xmlns:p="http://jboss.org/schema/seam/pdf"
 title="Klaros-Testmanagement Test Suite Report" marginMirroring="true"
 author="#{user.name}" creator="#{user.name}" pageSize="A4">
 <f:facet name="header">
   <p:font size="8">
     <p:header borderWidthBottom="0.1" borderColorBottom="black" borderWidthTop="0" alignment="center">
       <p:text value="Example report - generated #{date} by #{user.name}"/>
     </p:header>
     <p:footer borderWidthTop="0.1" borderColorTop="black" borderWidthBottom="0" alignment="center">
       <p:text value="Page " />
       <p:pageNumber />
     </p:footer>
   </p:font>
 </f:facet>
 <!-- print the frontpage -->
 <p:paragraph alignment="center" spacingAfter="100">
   <p:text value="" />
 </p:paragraph>
 <p:font style="bold" size="32">
   <p:paragraph alignment="center" spacingAfter="75">
     <p:text value="Test Case Report" />
   </p:paragraph>
 </p:font>
 <p:font style="normal" size="12">
   <p:paragraph alignment="center" spacingAfter="5">
     <p:text value="Created by" />
   </p:paragraph>
 </p:font>
 <p:font style="bold" size="16">
   <p:paragraph alignment="center" spacingAfter="5">
     <p:text value="#{user.name} (#{user.email})"/>
   </p:paragraph>
 </p:font>
 <p:font style="normal" size="12">
   <p:paragraph alignment="center" spacingAfter="5">
     <p:text value="at" />
   </p:paragraph>
 </p:font>
 <p:font style="bold" size="16">
   <p:paragraph alignment="center" spacingAfter="75">
     <p:text value="#{date}" />
   </p:paragraph>
 </p:font>
 <p:newPage/>
 <ui:fragment rendered="#{results != null}">
   <p:font style="normal" size="12">
```

```
<p:paragraph alignment="left" spacingAfter="10">
      <p:text value="The test results for " />
   </p:paragraph>
  </p:font>
</ui:fragment>
<!-- Test result table -->
<p:table columns="4" widths="1 1 3 3">
  <!-- create the headline with bold characters -->
  <p:font size="10" style="bold">
    <p:cell horizontalAlignment="center" verticalAlignment="top">
      <p:paragraph>
        <p:text value="Name" />
      </p:paragraph>
   </p:cell>
   <p:cell horizontalAlignment="center" verticalAlignment="top">
      <p:paragraph>
        <p:text value="Result" />
      </p:paragraph>
   </p:cell>
    <p:cell horizontalAlignment="center" verticalAlignment="top">
      <p:paragraph>
        <p:text value="Test run description" />
      </p:paragraph>
   </p:cell>
   <p:cell horizontalAlignment="center" verticalAlignment="top">
      <p:paragraph>
        <p:text value="Summary" />
      </p:paragraph>
   </p:cell>
  </p:font>
  <!-- display the attributes of the test results -->
  <p:font size="8">
    <ui:repeat value="#{results}" var="tcr">
      <p:cell verticalAlignment="top" horizontalAlignment="left">
        <p:paragraph>
          <p:text value="#{tcr.testCase.name}" />
        </p:paragraph>
      </p:cell>
      <!-- decide which color has to be displayed, based on the test result -->
      <ui:fragment rendered="#{tcr.error}">
        <p:cell backgroundColor="rgb(255,0,0)" verticalAlignment="top" horizontalAlignment="center">
          <p:paragraph>
            <p:text value="error" />
          </p:paragraph>
        </p:cell>
      </ui:fragment>
      <ui:fragment rendered="#{tcr.failure}">
        <p:cell backgroundColor="rgb(255,215,0)" verticalAlignment="top" horizontalAlignment="center">
          <p:paragraph>
            <p:text value="failure" />
          </p:paragraph>
        </p:cell>
      </ui:fragment>
```

```
<ui:fragment rendered="#{tcr.passed}">
         <p:cell backgroundColor="rgb(0,255,0)" verticalAlignment="top" horizontalAlignment="center">
           <p:paragraph>
             <p:text value="passed" />
           </p:paragraph>
         </p:cell>
       </ui:fragment>
       <p:cell verticalAlignment="top" horizontalAlignment="left">
         <p:paragraph>
           <p:text value="#{tcr.description}" />
         </p:paragraph>
       </p:cell>
       <p:cell verticalAlignment="top" horizontalAlignment="left">
         <p:paragraph>
           <p:text value="#{tcr.summary}" />
         </p:paragraph>
       </p:cell>
       <!-- Print the test case description below the result row.
            To differ from the next row use a bigger border for the bottom. -->
       <p:cell colspan="4" verticalAlignment="top" horizontalAlignment="left"
               borderWidthBottom="1" paddingBottom="3">
         <p:paragraph>
           <p:font size="6" style="bold">
             <p:text value="Testcase description:" />
           </p:font>
           <p:font size="6">
             <p:text value="#{tcr.testCase.description}" />
           </p:font>
         </p:paragraph>
       </p:cell>
     </ui:repeat>
   </p:font>
 </p:table>
</p:document>
```

Appendix F. Icon Index

F.1. Sections

lcon	Description
ľ	Define
***	Plan
\$	Execute
¢	Evaluate
يكي	Configure

Table F.1. Sections

F.2. Actions

F.2.1. Actions

lcon	Description
Ο	Activate object
Q	View object
ľ	Edit object
D	Duplicate object
⑪	Delete object
Ð	Restore object
\diamond	Purge object
Ð	Print object
S	Refresh object
ષ	Create revision
+	Add item
_	Remove item
×	Clear item
	Assign to category
0	Limit display to active iteration
Ľ	Open external link
ଜ	Connection check
٦	URL validation
8	Change the role of the user in a project
	Authentication check
Д	Bookmark the page

lcon	Description
샰	Start a download
	Send Email
V	Open the filter section
8	Merge objects
žΞ	Remove duplicate objects
	Assign a test case or test suite to a job
(Show all entries
ø	Show only already executed entries
20	Show only entries that have not yet been executed
U	Show only the most recently executed entries

Table F.2. Common Actions

F.2.2. Execution Actions

lcon	Description
\$	Execute object
.	Manually execute object
_ 0	Execution pending
*	Execute object review
20	Reopen or update the object
-5	Import test result

Table F.3. Execution Actions

F.2.3. Test Execution Actions

lcon	Description
G	Previous step
M	Skip step
M	Skip all remaining steps and finish test
0	Pause execution
ľ	Edit step result
۲	Request test review
Ð	Create new issue
()	Link existing issue
\odot	Step passed
⚠	Step failed
۲	Error during step execution
*	Step result inconclusive

lcon	Description
?	Step skipped

Table F.4. Test Execution Actions

F.2.4. Arranging Actions

lcon	Description
<u>م</u>	Move objects up on the same hierarchy level
▽	Move objects down on the same hierarchy level
1	Move objects to a selected position on the same hierarchy level
D	Move objects one hierarchy level down
٥	Move objects one hierarchy level up
KN	Inserts a test segment before test case step
KX	Inserts a test segment after test case step
2	Expand Window
۶ ^K	Compress Window
[]	Expand items
ן ר יר	Collapse items
>	Insert a test step to the right
<	Insert a test step to the left
»	Insert a test segment to the right
«	Insert a test segment to the left

Table F.5. Arranging Actions

F.3. Table Operations

lcon	Description
an a	Opens the category section
\mathbf{V}	Filter / Sort
Q	Apply quick filter
公	Export table
Ξ	Configure columns

Table F.6. Table Operations

F.4. Information

lcon	Description
ନ	Information
Δ	Warning

lcon	Description
8	Error
A	Delete disabled
I	Linked with other objects
()	Issues exist for this object
	Object is synchronized from external source
0	Synchronized object has pending updates
Ŷ	Hint
ු	Job is blocked
00	Job has a pending dependency

Table F.7. Information

F.5. Properties

F.5.1. Results

lcon	Description
\odot	Passed
⚠	Failed
Θ	Error
*	Step result inconclusive
0	Skipped

Table F.8. Results

F.5.2. Priorities

lcon	Description
\otimes	Blocker
*	Critical
^	High
≈	Low
~	Trivial

Table F.9. Priorities

F.6. Document Formats

lcon	Description
57	CSV
×	Excel

lcon	Description
\$	HTML
ß	PDF
¢	XML

Table F.10. Document Formats

Glossary

А

Administrator	A user role that has full access to all functions.
	This includes installation and maintenance of the software, updates, system settings, user administration, backups and all rights to create, execute and evaluate tests.
Authentication	Users are usually authenticated via user name and password.
	In Klaros-Testmanagement the user authentication can be done direct- ly in the software or via an external LDAP or Active Directory directory.
Automated Test Cases	The test results from automation tools are imported via result files in- to Klaros-Testmanagement. If both manual and automated automated test results are available, they can be evaluated together.
В	
Bugzilla	Bugzilla is an open source issue management system. Detailed infor- mation can be found on the Bugzilla Web Site.
	Klaros-Testmanagement provides an interface to Bugzilla.
С	
Change Tracking	Complete tracking of all changes to managed objects through auto- matic logging of modifications.
	In Klaros-Testmanagement, all changes are highlighted in color on the detail page of the corresponding Object over time.
Category	All objects can be arranged in categories for better overview. Grouping of objects (requirements, iterations, test systems, test environments, test cases or test suites) into a category tree can be done according to self-selected criteria. Multiple categorizations are also possible.
Community Edition	The free edition of Klaros test management. It has a limited feature set, but can also be used freely for commercial purposes. See also Enter- prise">Enterprise Edition.
Compliance	The capability of the software product to adhere to standards, conven- tions or regulations in laws and similar prescriptions [ISO9126].
Compliance Rate	In Klaros Test Management, the conformance rate shows how many test cases assigned to a requirement have been executed with the re- sult "Passed".

Continuous Integration		Continuous Integration (CI) is the automated integration of code changes from multiple contributors into a single software project. It is an important DevOps best practice that enables developers to regular- ly merge code changes into a central repository where builds and tests are then executed.
		A plug-in for the Jenkins Continuous Integration Server automatically transfers automatically transfers the test results of a build to Klaros Test Management, where they can be are evaluated.
Coverage, Coverage	Requirements	The requirement coverage represents the ratio of requirements cov- ered by tests to the total set of defined requirements.
		In Klaros Test Management, requirements can be directly entered and assigned to test cases. assigned to test cases. Individual requirements can be covered by several tests and one test can cover more than one requirement.
D		
Dashboard		A dashboard displays a collection of information, typically in the form of charts.
		In Klaros-Testmanagement dashboards display the evaluations for a specific test project. The displayed diagrams can be individually combined to form a dashboard. Each user can compile as many dashboards as desired.
		Dashboards can be private (visible only to the creator) or public. Ad- ministrators can create default dashboards.
Data Base		A database is a collection of information organized in interrelated ta- bles of data and specifications of data objects.
		Klaros-Testmanagement requires a database system to store the objects and is delivered with a preconfigured Apache Derby database. Other supported database systems are MariaDB, Microsoft SQLServer, MySQL and PostgresSQL.
Details Page		Each object (such as test cases, requirements, iterations, etc.) has its own detail page with various other subviews. These vary depending on the object and show detailed information for example "Properties", "Attachments", "Changes" and "Results" an.
Docker		Docker is free software for isolating applications through container vir- tualization. The containers contain all necessary packages and can thus be easily transported and installed as files.
		Klaros Test Management can also be run as a container within a Dock- er-based environment. Ready-to-use Docker images for various data- bases are already available. already available. Detailed documentation is available at GitHub.

E	
Enterprise Edition	The commercial edition with full support is licensed per user. The mini- mum installation supports 3 users. A free 30-day demo can be request- ed Demo.
Environment Variables	Environment variables are dynamically named values that can affect the behavior of running processes on a computer. They are part of the environment in which a process runs.
	By setting the environment variable KLAROS-HOME before the start of the application, the destination of the Klaros home directory can be moved to another location.
Error (Verdict)	A test execution error occurs when the system cannot execute the test correctly. It should not be confused with a failure.
Evaluation	The type of test result evaluation: manual or automated.
F	
Failure (Verdict)	A test is deemed to fail if its actual result does not match its expected result. (ISTQB). It should not be be confused with an error.
Functional Test Design Tech- nique	Procedure to derive and select test cases based on an analysis of the specification of the functionality of a component or system without reference to its internal structure. (ISTQB), also known as black box and white box tests.
G	
Guest	A user role with full, but read-only access to all data. A guest can view and save reports in various formats. This role is intended for e.g. project managers, customers or reviewers/reviewers.
I	
ID	The ID of an object is created automatically when it is created and can- not be changed later. All IDs consist of an abbreviation for the respec- tive object (e.g. TC for test case, ITR for iteration) and an ascending number. number. "SEG00025" would therefore be the test segment 25.
Inconclusive (Verdict)	A test is inconclusive if it is unclear, if its actual result matches its expected result.
Integration	Integration refers to the connection of external systems.
	Klaros-Testmanagement has interfaces for integration with issue and requirements management systems, test automation and load testing tools, and and continuous integration servers. There are also interfaces to authentication and e-mail servers.

lssue	An issue is a partial aspect that is required to complete a piece of work. This can be a bug, a requested feature, a task, missing documentation or something similar. An issue is tracked in an issue management sys- tem.
	Klaros-Testmanagement has interfaces to several issue management systems. This means that errors found can be transferred directly to the connected IMS, without having to leave the application.
Iteration	An iteration is a complete development loop resulting in a release (in- ternal or external) of an executable product, a subset of the final prod- uct under development, which grows from iteration to iteration to be- come the final product. (ISTQB)
	In Klaros test management supports agile development principles and manages iterations as dedicated objects.
J	
Java	In order to run Klaros-Testmanagement, a Java runtime environment is required. This is delivered with the installation file and used automatically.
JavaScript	JavaScript is a programming language used in web browsers and is required for running Klaros test management. When using JavaScript blockers such as NoScript or similar. When using JavaScript blockers such as NoScript or similar, the execution of JavaScript for Klaros- Testmanagement must be allowed as an exception.
JIRA	JIRA is a popular application by the company Atlassian Software for defect management, troubleshooting, and project management.
	Klaros-Testmanagement has an integration with JIRA.
Job	Test activities are planned and logged by means of jobs. The execution and results of the executed jobs are automatically logged and used for evaluation.
L	
Latest Success Rate, Report Diagram	The "Last Success Rate" report displays the latest results of executed test cases for the selected combination of test system (one or more) and test environment (one or more) as a chart on the dashboard.
Layout Template	Besides the script, the layout template is the other elementary part of a report template. It describes the structure of the generated report document in an XML-based description language. Layout templates can be adapted to individual requirements.
Log Panel	The log panel displays status messages such as warnings or infor- mation are displayed. By default, only the last status message is dis- played. Clicking on the + icon opens the history.

Μ	
Mantis	Mantis (MantisBT, Mantis Bug Tracker). is an open source application for defect management and issue tracking in software development.
	Klaros test management provides integration with Mantis.
Ν	
Notification Event	Notification events may trigger an automatic e-mail notification to specific users. These are sent out as soon as the designated event has has occurred. Possible notification events are, "Job assigned", "User account created", "Test execution failed", "User account password changed" and "Job ready for execution".
0	
Object	The objects defined in Klaros-Testmanagement are project, test case, test step, test segment, test step result, test case result, test suite, test suite result, test environment, system under test, test run, requirement, iteration and job.
Overview Page	Each object type has an overview page. There the individual objects, such as test cases or iterations, are displayed in tabular form. New objects are also created here created.
P	
Passed (Verdict)	A test is passed if its actual result matches its expected result. (ISTQB)
Precondition	Environmental and state conditions that must be fulfilled before the component or system can be executed with a particular test or test procedure. (ISTQB).
Postcondition	Environmental and state conditions that must be fulfilled after the execution of a test or test procedure. (ISTQB)
Print View	All objects can be rendered in a printout-optimized representation. The level of detail can be specified with various parameters.
Project	A project is the main object in Klaros Test Management and contains all other objects assigned to a test project.
Project Overview, Report Dia- gram	The "Project Overview" report displays the number of executed/not ex- ecuted test cases or execution definitions and the number of sched- uled/executed jobs as a chart on the dashboard.
Q	
QF-Test	QF-Test from Quality First Software is a cross-platform GUI test au-

tomation tool.

Klaros-Testmanagement integrates with QF-Test and can import the automatically generated test results for further processing.

R	
Redmine	Redmine is an open source application for defect management and issue tracking in software development.
	Klaros-Testmanagement integrates with Redmine.
Reports	Reports provide information on the status of testing activities, the con- dition of the software, indications of critical components and enable forward planning. For this purpose, the data from the database are sort- ed according to certain criteria and prepared in textual and graphical overviews. Examples of reports are "Project overview" and "Test activ- ity" as well as overviews of test progress, test results and test runs.
	Klaros-Testmanagement contains numerous predefined, basic re- ports. Individual reports can be created by the user. All reports are ex- portable to various formats such as PDF or Excel (also for further edit- ing).
Report Template	A report template predefines a report that can take report parameters and be customized by the user as needed. It consists of a <i>Script</i> and a <i>Layout Template</i> .
	With report templates, ready-made reports are available which can be customized by the user as needed.
REST (Representational State Transfer)	Representational State Transfer (REST) is a software architecture style that uses a subset of HTTP. It is commonly used to provide web services.
	Klaros-Testmanagement offers several integration interfaces based on REST. These include read access to all stored information as well as import interfaces for test results, test cases and requirements.
Requirement	A user-required property or capability that software must meet or pos- sess in order to Comply with a contract, standard, specification, or oth- er formal document. [According to IEEE 610].
	Requirements can be managed either directly in Klaros Test Manage- ment or or synchronized with external sources such as JIRA. A ref- erence to the corresponding requirement, e.g. to a document, can be added to test cases and test suites (traceability).
Result, expected	The behavior predicted by the specification, or another source, of a component or system under specified conditions (ISTQB).
Revision	If a major change is made to an object and the previous version is still needed, a new revision should be created. Only selected objects such as test cases or requirements support this mechanism.

	Glossary
Role	A user role defines the rights of the user. Roles can be defined globally and project project-specific.
	The following user roles are provided: "Administrator", "Test Manager", "Tester" and "Guest".
S	
Selenium	Selenium is a web browser automation tool and is mainly used for test- ing web apps.
	Selenium produces JUnit XML compliant test results that can be imported into Klaros test management for further processing.
Script	The script is an elementary component of a report template along with the layout template. It extracts the data intended for the report and is available as a Java class. Scripts can be adapted to individual require- ments.
Skipped, Verdict	A test case or a test step was omitted during execution.
System Account	A system access reserved for automated operations such as importing test results. A login via the login page is not possible with this.
Т	
Test Activity, Report Diagram	The "Test Activity" report shows the test case results for a selected combination of one or more test systems and test environments during a selected in a selected time period as a graph on the dashboard.
Test Case	A test case comprises a set of input values, execution conditions, ex- pected results, and postconditions defined for a specific system or test object under test. It is developed with respect to a specific goal or test condition, such as conformance to specific requirements.
Test Case History, Report Di- agram	The "Test History - Overview" report displays the rate of defined test cases versus executed test cases of a project for one or more test en- vironments and one or more test systems in a given time period as a graph on the dashboard.
Test Case Priority	The priority of the test case: "Low", "Medium" or "High".
Test Case Progress, Report Diagram	The Test Progress - Overview report shows the rate of the executed versus the defined test cases of a project project for a given set of test environments and test systems as a graph on the dashboard.
Test Case Status	The status of the test case. Possible values are "Locked", "Approved", "Draft" and "Omitted".
Test Step	A test step in Klaros-Testmanagement is the smallest action that is processed within a test case. It contains execution preconditions, ex-

	pected results, and execution postconditions for each individual action during test execution.
Test Case Result	The result of the execution of a test case including a verdict such as "Skipped", "Passed", "Failed" or "Failed".
Test Environment	A test environment represents the external circumstances that can in- fluence the test result. Examples for test environments are e.g. the op- erating system or an application server (e.g. Tomcat 10 with Ubuntu 20.04.2) or also parameters such as temperature or speed.
Test Environment Overview, Report Diagram	The "Test Environment - Overview" report displays the success and progress rate of test systems under a single test environment in a radar chart on the dashboard. If less than three test systems are configured, a bar chart is displayed instead.
Tester	A user role that is allowed to view objects and execute test cases and test suites.
Test Execution	The process of running a test on the component or system under test, producing actual result(s). (ISTQB)
	Klaros-Testmanagement automatically guides and logs test execution, see Testrunner.
Test Manager	A test manager has full read and write access to all data of a project. He can create, modify and delete all types of objects and manage project settings.
Test Run	A test run contains the results of the execution of a single test case or an entire test suite. It always refers to exactly one test system and exactly one test environment.
Testrunner	Klaros-Testmanagement contains a web-based client for the execution of manual tests. This guides the tester through the test steps, gives him the possibility to create annotations and attachments and auto- matically logs the test progress and results.
Test Step	A test step in Klaros Test Management is the smallest action that is processed within a test case. It contains execution preconditions, ex- pected results and execution postconditions for each individual action during test execution.
testSegment	A test segment in Klaros-Testmanagement is defined sequence of in- dividual test steps that can be inserted into a test case. A test segment can be used in several test cases at once. This makes it possible to realize a module concept based on test steps.
Test Suite	A test suite is a set of test cases that can be executed together in a group.
Test system, System under test, Unit under test, Test ob- ject	In Klaros-Testmanagement, a test system represents the component or system that is to be tested. Depending on the business sector, the

	terms system under test, SUT, DUT or test object are also commonly used.
System under Test Overview, Report Diagram	The Test System Overview report shows the success and progress rate of test progress rate of test environments under a single test system in a radar chart on the dashboard. If less than three test environments are configured, a bar chart is displayed instead.
Test Type	A group of test activities based on specific test objectives with the purpose of testing a component or system for specific properties (Functional, Non-Functional, Structural, Regression, Retest).
Trac	Trac is an open source application for defect management and issue tracking in software development. Klaros Test Management integrates with Trac.
V	
Variety	The expected outcome of the test: positive or negative.
Verdict	A rating is the result of the executed test case, test step or test suite. Possible ratings are "Passed", "Failed", "Error", "Inconclusive", "Skipped".
W	
Web Browser	Klaros-Testmanagement is a web application and runs within a web browser. Supported browsers are Apple Safari, Google Chrome, Mi- crosoft Edge and Mozilla Firefox.
Work Log	The work log shows the chronological progression and duration of the activities for processing a job. The time spent for the entire test execution results from the sum of the execution times. the sum of the execution times.
Y	
YouTrack	YouTrack is a popular application by the company JetBrains for defect management, troubleshooting, and project management.

Index

A

Access Permissions, 287 API, 285 Attachments, 130 Referencing, 73 Upload, 65 Authentication Single Sign-on, 4 Automated Installation Script, 31 Automated Test Cases, 7

В

Bookmarks, 60 Bulk Actions, 58

С

CAS, 256 Categories, 56 Change History, 67 Chart, creating, 266 Classes KlarosAttachment, 361 KlarosCategoryNode, 362 KlarosCategoryTree, 363 KlarosConfiguration, 365 KlarosContext, 405 KlarosEnumValue, 366 KlarosExternalServer, 367 Klaroslssue, 368 KlarosIssueManagement, 370 KlarosIteration, 371 KlarosJob, 373 KlarosJobDependency, 374 KlarosJobTimeBlock, 375 KlarosJobUpdateAction, 376 KlarosLabeledObject, 377 KlarosNamedEntity, 378 KlarosProperty, 379 KlarosRepositoryEntity, 380 KlarosRequirement, 381 KlarosRequirementsManagement, 383 KlarosResult, 384 KlarosRevision, 385 KlarosSUTImplementation, 387 KlarosTestCase, 388 KlarosTestCaseResult, 390

KlarosTestCaseStep, 392 KlarosTestCaseStepResult, 393 KlarosTestEnvironment, 394 KlarosTestExecutable, 395 KlarosTestRun, 396 KlarosTestSegment, 397 KlarosTestStepContainer, 399 KlarosTestSuite, 400 KlarosTestSuiteResult, 401 KlarosUser, 402 Parameter, 412 Configure, 223 E-Mail, 251 Languages, 240 Notification Scheme, 234 System, 232 User Interface, 239 Users, 226 Create, 229 Delete, 229

D

Dashboard, 177 Default, 178 Default URL, 33 Docker, 4 Description, 14

Ε

E-Mail, 251 Enumeration Values, 62 Execute, 154 Jobs, 154 Review, 156 Step-by-Step View, 158 Tabular Step View, 160 Test Case, 156 Test Suite, 163 Export Excel, 283

F

Filtering, 54

Η

Help Menu, 53

I

Icons, 435

Actions, 435 Arranging Actions, 437 Execution Actions, 436 **Test Execution Actions**, 436 Document Formats, 438 Information, 437 Properties Priorities, 438 Results, 438 Sections, 435 Table Operations, 437 IKlarosAttachment, 288 IKlarosCategoryNode, 289 IKlarosCategoryTree, 290 IKlarosConfiguration, 292 IKlarosContext, 407 IKlarosEnumValue, 296 IKlarosExternalLink, 297 IKlarosExternalServer, 298 **IKlarosIssue**, 299 IKlarosIssueManagement, 303 IKlarosIteration, 304 IKlarosJob, 308 IKlarosJobDependency, 312 IKlarosJobTimeBlock, 313 IKlarosJobUpdateAction, 315 IKlarosLabeledObject, 316 IKlarosNamedEntity, 318 IKlarosPersistentObject, 319 IKlarosProperty, 319 IKlarosQueryFactory, 403 IKlarosRepositoryEntity, 320 IKlarosRequirement, 322 IKlarosRequirementsManagement, 325 IKlarosResult, 326 IKlarosRevision, 330 IKlarosSUTImplementation, 331 IKlarosTestCase, 334 IKlarosTestCaseResult, 340 IKlarosTestCaseStep, 342 IKlarosTestCaseStepResult, 343 IKlarosTestEnvironment, 346 IKlarosTestExecutable, 347 IKlarosTestRun, 348 IKlarosTestSegment, 352 IKlarosTestStepContainer, 355 IKlarosTestSuite, 356 IKlarosTestSuiteResult, 359 IKlarosUser, 360

Import Excel, 269 Format, Requirements, 273 Format, Test Cases, 269 **QF-Test**, 281 XML, 272 XML, Requirements, 275 Installation, 17 Integration, 82, 240 Issue Management, 82 Requirements Management, 83, 249, 249 Test Automation, 277 Interface, 239 Interfaces IKlarosAttachment, 288 IKlarosCategoryNode, 289 IKlarosCategoryTree, 290 IKlarosConfiguration, 292 IKlarosContext, 407 IKlarosEnumValue, 296 IKlarosExternalLink, 297 IKlarosExternalServer, 298 IKlarosIssue, 299 IKlarosIssueManagement, 303 IKlarosIteration, 304 IKlarosJob, 308 IKlarosJobDependency, 312 IKlarosJobTimeBlock, 313 IKlarosJobUpdateAction, 315 IKlarosLabeledObject, 316 IKlarosNamedEntity, 318 IKlarosPersistentObject, 319 IKlarosProperty, 319 IKlarosQueryFactory, 404 IKlarosRepositoryEntity, 320 IKlarosRequirement, 322 IKlarosRequirementsManagement, 325 IKlarosResult, 326 IKlarosRevision, 330 **IKlarosSUTImplementation**, 331 IKlarosTestCase, 334 IKlarosTestCaseResult, 340 IKlarosTestCaseStep, 342 IKlarosTestCaseStepResult, 343 IKlarosTestEnvironment, 346 IKlarosTestExecutable, 347 IKlarosTestRun, 348 IKlarosTestSegment, 352 IKlarosTestStepContainer, 355

IKlarosTestSuite, 356 IKlarosTestSuiteResult, 359 IKlarosUser, 360 Issues, 12 Iteration, 10 Iterations, 87

J

Job, 10, 139 Arranging, 141 by User, 152 from Test Cases, 149 from Test Suites, 150

К

KlarosAttachment, 361 KlarosCategoryNode, 362 KlarosCategoryTree, 363 KlarosConfiguration, 365 KlarosContext, 405 KlarosEnumValue, 366 KlarosExternalServer, 367 Klaroslssue, 368 KlarosIssueManagement, 370 KlarosIteration, 371 KlarosJob, 372 KlarosJobDependency, 374 KlarosJobTimeBlock, 375 KlarosJobUpdateAction, 376 KlarosLabeledObject, 377 KlarosNamedEntity, 378 KlarosProperty, 379 KlarosRepositoryEntity, 380 KlarosRequirement, 381 KlarosRequirementsManagement, 383 KlarosResult, 383 KlarosRevision, 385 KlarosSUTImplementation, 387 KlarosTestCase, 388 KlarosTestCaseResult, 390 KlarosTestCaseStep, 392 KlarosTestCaseStepResult, 393 KlarosTestEnvironment, 394 KlarosTestExecutable, 395 KlarosTestRun, 396 KlarosTestSegment, 397 KlarosTestStepContainer, 398 KlarosTestSuite, 400 KlarosTestSuiteResult, 401

KlarosUser, 402

L

Languages, 47, 240 LDAP, 253 License Model, 14 Login, 49

Μ

monitoring, 41

Ν

Notification, 236 Notification Scheme, 234

0

Objects, 12 Assigning, 58

Ρ

Parameter, 412 Parameter Types, 263 Performance Tests supported tools, 3 Plan, 139 Postcondition, 159 Precondition, 159 Print Views, 59 Project, 5 Projects, 75

Q

Quick-Select, 51 Quotes, 47

R

Remote API, 285 Reports Example, 264 Issues, 207 Latest Success Rate, 180 Parameters, 262 predefined, 3 Project Overview, 180 System under Test - Overview, 182 Templates, 223, 260 Test Activity, 181 Test Environment - Overview, 183 Test History, 185
Test Progress, 184 Types, 179 User Defined, 259 Requirement, 9 Requirements, 94 Requirements Management, 249 REST, 285 Backup, 284 Results Single Test Case, 198 Test Case, 195 Test Run, 192 Revisions, 63 Rights and Roles, 287 Roles and Rights, 287

S

Search, 51 Full Text Search, 51 Quick-Navigation, 52 Session Timeout, 233 Single sign-on Authentication, 256 Sorting, 56 SSL support, 44 Synchronization, 275 System, 232 System Prerequisites, 14 System under Test, 8 Systems under Test, 107

Т

Table Filtering, 54 Test Automation supported tools, 2 Test Case, 5 Test Case Result, 7 Test Cases, 120 Test Environment, 8 Test Environments, 101 Test Run, 9 Test Segment, 7 Test Segments, 115 Test Step, 6 Test Suite, 8 Test Suite, 8 Test Suite Result, 205 Test Suites, 132

U

Uninstall, 45

User, 226 User Defined Properties, 61 user roles, 11