

User Manual

TestLink version 1.8

Version: 1.09
Status: Reviewed

© 2004 - 2008 TestLink Community

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. The license is available in ["GNU Free Documentation License" homepage](#).

Table of Contents

1	General information	3
1.1	Overall structure	3
1.2	Basic Terminology	3
1.3	Example of TestLink simple work-flow	4
2	Test Projects	6
2.1	Creating new Test Projects	6
2.2	Edit and delete Test Projects	6
3	Test Specification	7
3.1	Test Suites	7
3.2	Test Cases	8
3.3	Keywords	9
4	Requirement based testing	11
4.1	Availability	11
4.2	Requirements Specification Document	11
4.3	Requirements	12
5	Test Plans	14
5.1	Create and delete Test Plan	14
5.2	Builds	14
5.3	Populating Test Plan - adding Test Cases	14
5.4	Test execution assignment	16
5.5	Prioritize testing	16
5.6	Milestones	16
6	Test Execution	18
6.1	General	18
6.2	Navigation and settings	18
6.3	Test execution	19
7	Test Reports and Metrics	22
7.1	General Test Plan Metrics	22
7.2	Query Metrics	23
7.3	Blocked, Failed, and Not Run Test Case Reports	24
7.4	Test Report	24
7.5	Charts	24
7.6	Total Bugs For Each Test Case	25
7.7	Requirements based report	25
7.8	How to add a new report	25
8	Administration	27
8.1	User account settings	27
8.2	Role Permissions	27
8.3	Test Plan assignment to users	28
8.4	Custom Fields	28
9	Import and Export data	31
9.1	Import/Export Keywords	31
9.2	Export/Import Test Project	32
9.3	Import/Export Test suite	33
9.4	Just one Test Case	34
9.5	All Test Cases in test suite	35
9.6	Import/Export Software Requirements	35
9.7	Import Test Cases from Excel via XML	37

1 General information

TestLink is web based **Test Management** system. This manual should serve as source for users to understand processes, terms and organization of work with TestLink. See the **Installation manual** for more information about system requirements, installation steps and configuration. The latest documentation is available on www.teamst.org or testlink.sourceforge.net.

Please use our [forum](#) if you have questions that the manual doesn't answer.

1.1 Overall structure

There are three cornerstones: **Test Project**, **Test Plan** and **User**. All other data are relations or attributes of this base. First, we will define terms that are used throughout the documentation and testing world.

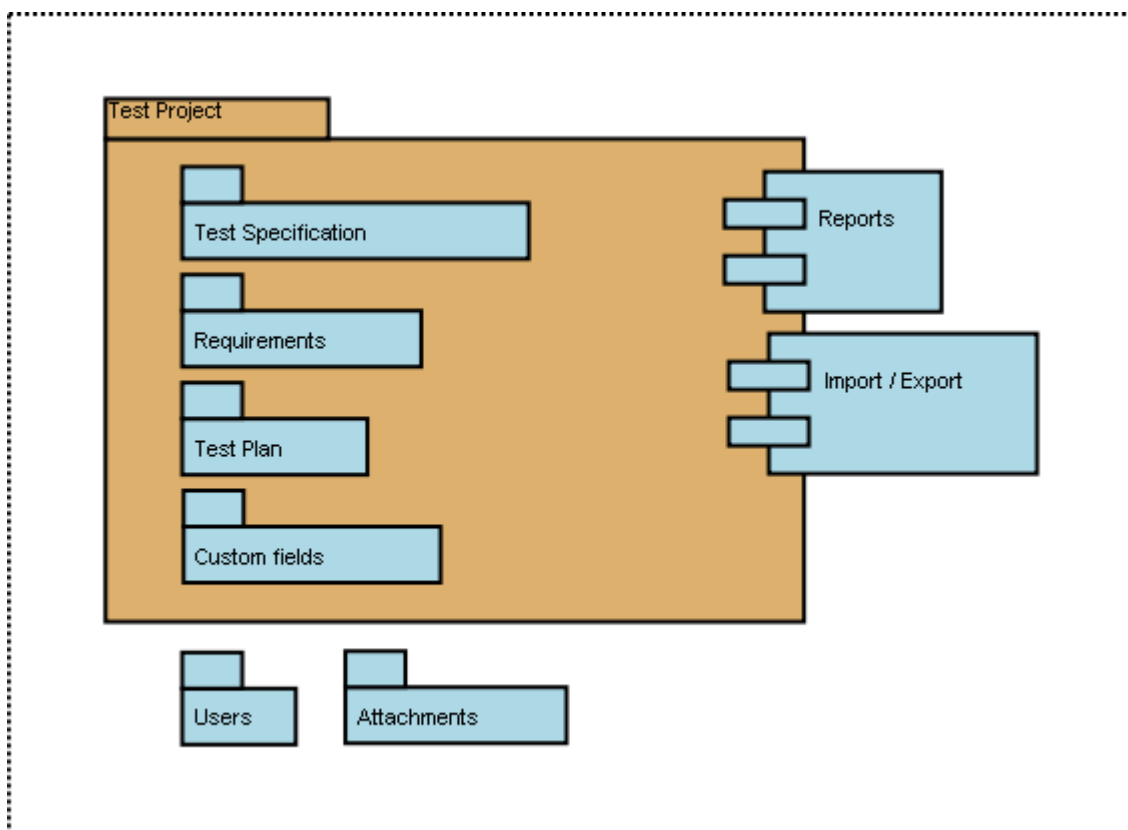


Illustration 1: Test Project is the basic component in TestLink

1.2 Basic Terminology

Test Case describes a testing task via steps (actions, scenario) and expected results. Test Cases are the fundamental piece of TestLink.

Test Suite (Test Case Suite) organizes Test Cases to units. It structures Test Specification into logical parts.

*Test Suite replaces the **components** and **categories** in TL 1.6 and earlier.*

Test Plan is created when you'd like to execute Test Cases. Test Plans can be made up of the Test Cases from one or many Test Projects. Test Plan includes Builds, Milestones, user assignment and Test Results.

Test Project is something that will exist forever in TestLink. Test Project will undergo many different versions throughout its lifetime. Test Project includes Test Specification with Test Cases, Requirements and Keywords. Users within the project have defined roles.

*Test Project was called **Product** in TL 1.6 and earlier.*

User: each TestLink user has a Role that defines available TestLink features. See more in chapter "User Administration". Illustration 2 shows common activities according to user roles.

1.3 Example of TestLink simple work-flow

1. Administrator creates a **Test Project** "Fast Food" and two users, Adam with rights "Leader" and Bela with rights "Senior Tester".
2. Leader Adam imports **Software Requirements** and for part of these requirements generates empty Test Cases. He reorganize them into two Test Suites: "Fish" and "Chips".
3. Tester Bela describes a test scenario (create a content of empty Test Cases) using these **Test Specification** that is organized into Test Suites.
4. Adam creates keyword "Regression testing" and assigns this keyword to ten of these Test Cases.
5. Adam creates a **Test Plan** "Fish & Chips 1", **Build** "Fish 0.1" and links all Test Cases in Test Suite "Fish" to this Test Plan. He assigns himself and Bela as resources to this Test Plan too.
6. Now developers produce a first build. Adam and Bela execute and record the testing with the result: 5 passed, 1 failed and 4 are blocked.
7. Developers make a new Build "Fish 0.2" and Bela tests the failed and blocked Test Cases only. This time these all blocked and failed Test Cases passed. They retest also all Test Cases with keywords "Regression testing" in addition.
8. A **manager** of this team would like to see the results. Administrator explains to him that he can create an account himself on the login page. Manager does it. He has default "Guest" rights and can see **Test Results** and Test Cases. He can see that everything passed in the overall report ;-) and problems in Build "Fish 0.1" in a report for that particular Build.¹
9. Later, developers finally add also "Chips" functionality. Adam creates a **Test Plan** "Fish & Chips 2". He can reuse the first Test Plan as template. All "Fish" Test Cases and roles will be automatically added. He create a new Build "Fish 1.1" and links all "Chips" Test Cases to this Test Plan too.
10. Now testing starts as usual.
11. Later, Admin creates a new **Test Project** for another product "Hot Dog". But this is another test team and a different story.

1 He, however, can edit nothing. ;-)

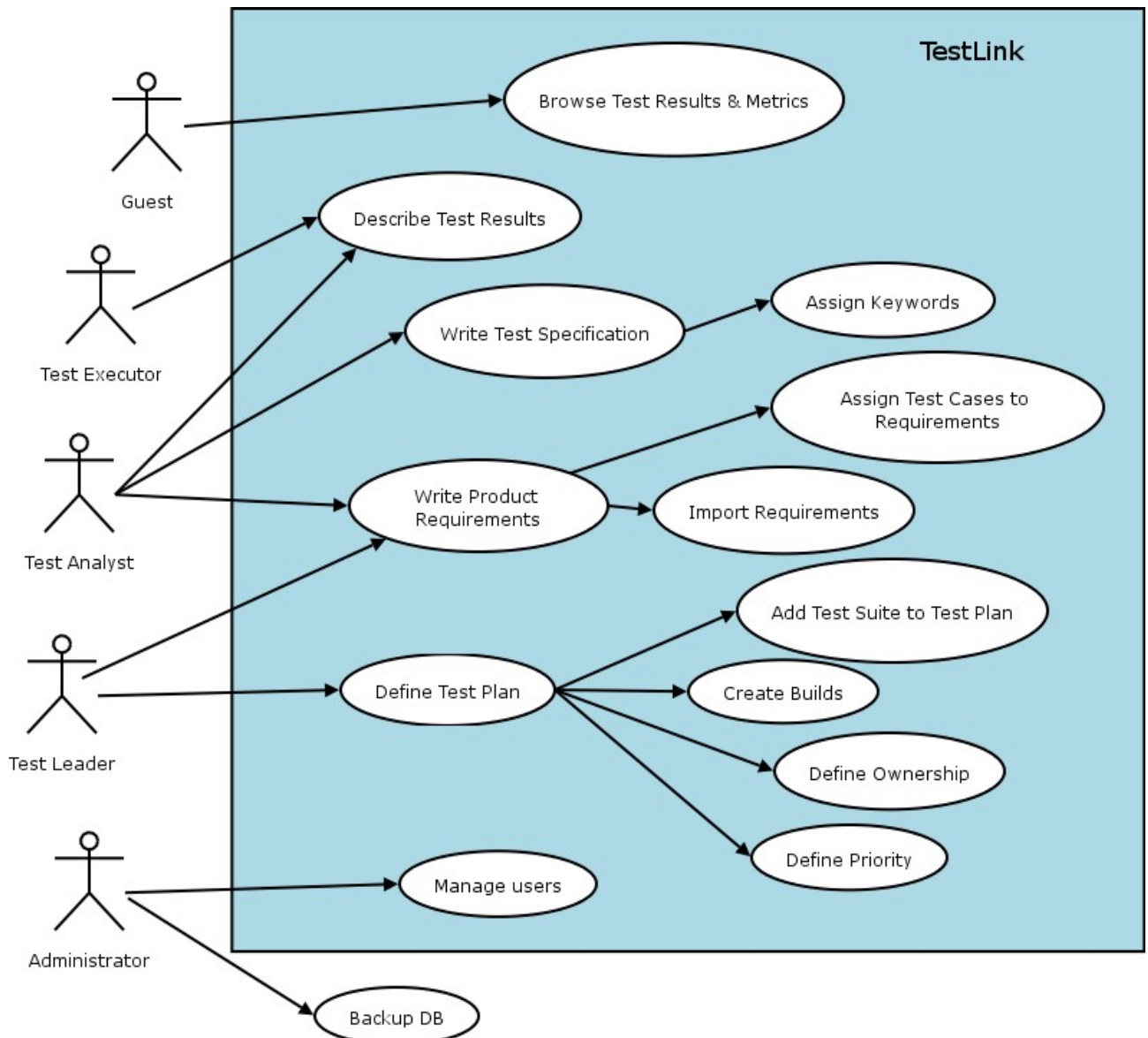


Illustration 2: Functionality overview

2 Test Projects

Test Projects are the basic organisational unit of TestLink. Test Projects could be products or solutions of your company that may change their features and functionality over time but for the most part remains the same. Test Project includes requirements documentation, Test Specification, Test Plans and specific user rights.

Test Projects are independent and do not share data. Consider using just one Test Project for one Test team and/or one product.

For example: there are two test teams for one product: system testing and integration testing. These teams will share some test cases. You should create one project for the product. The work of both teams will be structured in two or more trees in Test Specification and testing results will be collected to different Test Plans.

2.1 Creating new Test Projects

Creating a new Test Project requires "admin" rights. Each Test Project must have an unique name. There is text area to characterize the object of project. There are several features that are enabled/disabled on Test Project level:

1. Admin can enable requirements related functionality.
2. Background colours can be assigned to Test Project templates to visually distinguish them.

Note: The feature must be enabled in config at first.

3. Test prioritization could be enabled to select appropriate testing set in limited time.
4. Test automation support could be enabled as well.

Things to note when creating a new Test Project:

- Deleting Test Projects from the system is not recommended as this either orphans a large number of Test Cases or deletes the Test Cases from the system.
- Test Plans represent the testing of a Test Project at a certain point in time. Consequently, Test Plans are created from Test Project Test Cases. We do not recommend to create separate Test Projects for versions of one Product.
- TestLink supports importing XML or CSV data into a Test Project. This is explained further in the Import section, below.

2.2 Edit and delete Test Projects

Editing Test Projects requires "admin" rights. User can change Test Project name, colour of background and availability of requirements functionality, test prioritization and test automation.

User can **inactivate** the Test Project if it's obsolete. This will mean that the Test Project is not visible in the list within the top navigation bar for common users.

Admin will see this Test Project in the list marked by ''.*

User can also **delete** a Test Project. This action will also delete all related data from database. **This action is not reversible.** We strongly recommend using inactivate instead of delete.

3 Test Specification

TestLink breaks down the Test Specification structure into Test Suites and Test Cases. These levels are persisted throughout the application.

3.1 Test Suites

Users organize Test Cases into Test (Case) Suites. Each Test Suite consists of a title, formatted description Test Cases and possibly other Test Suites. TestLink uses tree structure for Test Suites. The common practise is that the description holds information valid for most of included data.

For example, the following could be specified: scope of included tests, default configuration, preconditions, links to related documentation, list of tools, infrastructure overview, etc.

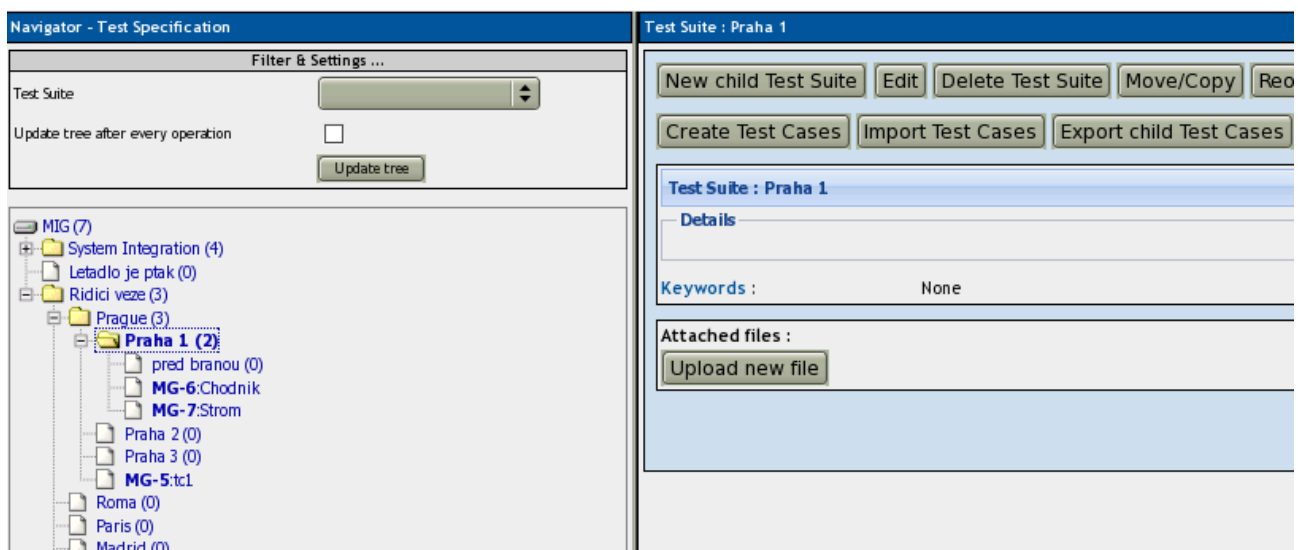


Illustration 3: Test Specification is structured by Test Suites

Creating one or more Test Suites is one of the first steps when creating your Test Project. Users (with edit right) can create, delete, copy, move, export and import both nested Test Suites and Test Cases. Title and description can be modified too.

You can reorder both Test cases and child Test Suites via Drag & Drop items on the navigation tree.²

Practice: Consider structure of your test specification. You could divide tests by functional/non functional testing, particular features, components. You can change the structure later of course – move test suites without lost of any information.

Practice: Later versions of your product could have some features obsolete. You can create a special Test suite 'Obsolete' or 'Your product 0.1' and move test cases there. Deleting of test case causes that earlier test results will be deleted too.

Attachments with external documents or pictures could be added into particular Test Suites.

Note: The functionality must be allowed via TestLink configuration.

More information about import and export are in appendix.

FAQ: Could I copy/move Test suites between projects? You cannot directly in this version. We designed Test projects fully independent each from other. Use import/export as workaround.

² It requires EXT-JS component (default).

3.2 Test Cases

Test Case is a set of inputs, execution preconditions, and expected results (outcomes) developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.

Test Cases have the following parts:

- **Title:** could include either short description or abbreviation (e.g. TL-USER-LOGIN)
- **Summary:** should be really short; just for overview.
- **Steps:** describe test scenario (input actions); can also include precondition and clean-up information here.
- **Expected results:** describe checkpoints and expected behaviour of a tested product or system.
- **The numeric ID** of a Test Case is assigned automatically by TestLink, and can not be changed by users. This ID is system wide, meaning that when a Test Case is created a global counter is used, independent of the Test Project in which the Test Case is created.
- **Attachments:** could be added if configuration allows it.

FAQ: Our test steps are pretty detailed are use a lot of formatting coming from Word. Use "Paste from msword" feature of FCKeditor. It cleans garbage. (Configure toolbar if not available.)

Test Case - Active Attribute

If several versions of a Test Case exist, it would be useful to have a new attribute, Active/Inactive, to use in this way:

- Every Test Case version is created ACTIVE
- An Inactive Test Case Version will not be available in "Add Test Cases to Test Plan". This can be useful for Test Designers. They can edit or change the Test Case Version and only when he/she decides it is completed, change Status to ACTIVE so it will be available to be used in a Test Plan.
- Once a Test Case Version has been linked to a Test Plan, and has results, it can't be turned INACTIVE.

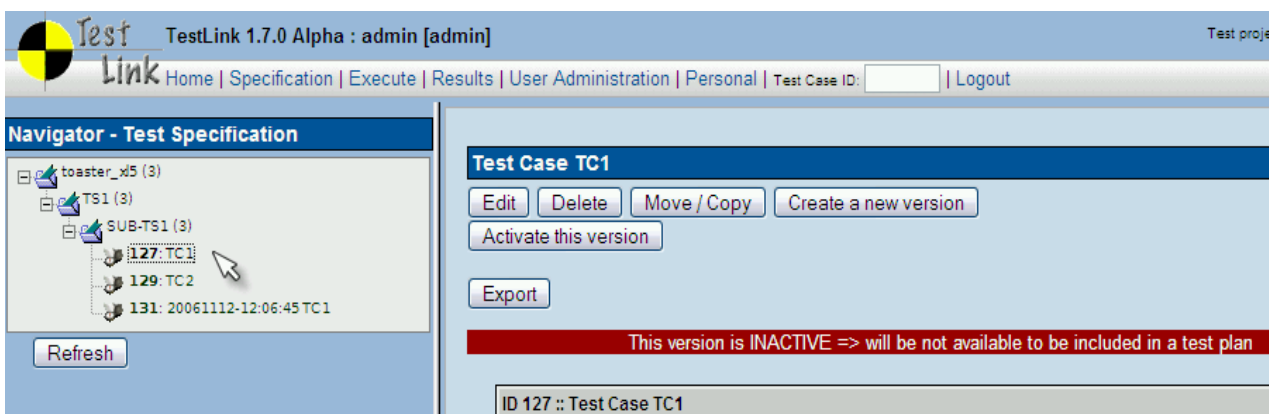


Illustration 4: What you see in Test Case specification

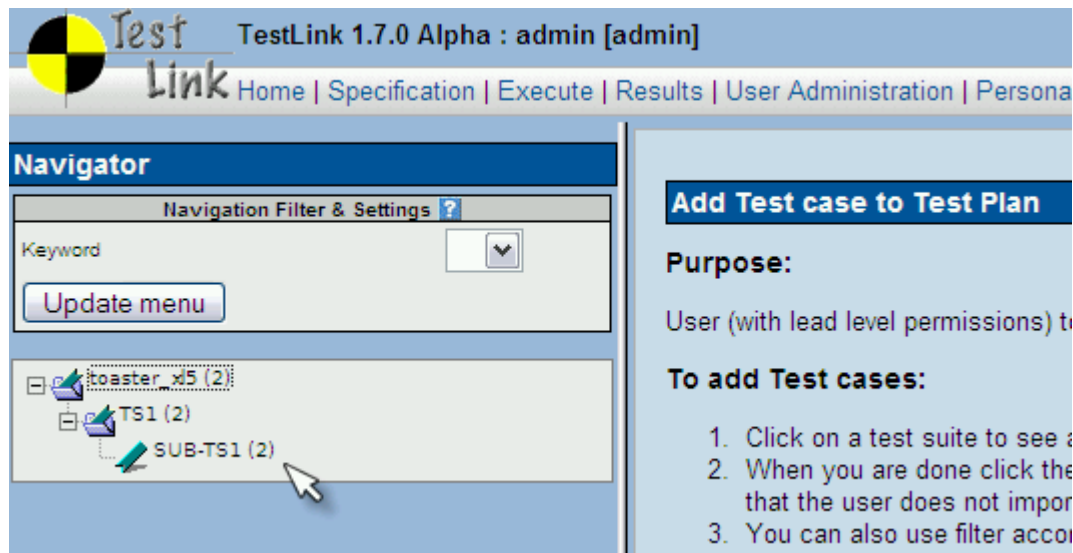


Illustration 5: What you see when trying to add Test Cases to a Test Plan

As you can note, the number near Test Project name (in this example: **toaster_xl5**) is 2, but the Test Project has 3 Test Cases. Test Case TC1 is not counted, because is inactive.

Removing Test Cases

Test Cases and Test Suites may be removed from a Test Plan by users with "lead" permissions. This operation may be useful when first creating a Test Plan since there are no results. However, removing Test Cases will cause the loss of all results associated with them. Therefore, extreme caution is recommended when using this functionality.

Requirements relation

Test Cases could be related to software/system requirements as n to n. The functionality must be enabled for a Test Project. User can assign Test Cases and Requirements via the Assign Requirements link in the main screen.

3.3 Keywords

Keywords were created to give users another level of depth when categorizing Test Cases. Keywords serve as a means of grouping Test Cases with some attribute within a Test Specification. For example, you can use it to define:

- Regression or Sanity set
- Reviewed Test Cases
- Set of Test Cases valid for one platform

Keyword Creation

At this time keywords can only be created by users with the *mgt_modify_key* rights. These rights are currently held only by Leaders. Once a keyword or grouping of keywords has been created users may assign them to Test Cases.

Assigning Keywords

Keywords may be assigned to Test Cases either from the assign keyword screen (in batch) or via the Test Case management (individually).

Filter by Keyword

Users have the ability to filter by Keywords to:

- Search Test Cases in Test Specification.
- Add groups of Test Cases in a Test Case Suite (Test Plan).
- Execute test screen.

4 Requirement based testing

To prove that a system is built as specified, testers use requirement based testing. For every requirement, they design one or more Test Cases. At the end of the test execution a test manager reports on the tests that are executed and the requirements that are covered. Based on this information the client and the various stakeholders decide whether a system can be transferred to the next test phase or can go live. Test managers use a combination of risk and requirement-based testing to ensure that a system is built as specified from the customer and stakeholders perspective. As a result, this complete testing delivers the following advantages:

- Linking risks and requirements will reveal vague or missing requirements. This is especially interesting for risks with a high priority.
- Testing can be focused on the most important parts of an information system first: covering the risks with the highest priority.
- Communicating in the same language as the client and the stakeholders. This makes it easier to report on the status of the Test Project. Then a better founded decision can be made whether to invest more in testing or take the risk.
- The risks and their priority make negotiating on the Test Project in times of pressure easier. What risks have to be covered within this Test Project and which ones can be postponed. Risk and requirement-based testing results in a better controlled Test Project. The communication with the client and the stakeholders is improved. The test manager begins testing with risks with the highest priority. The process is streamlined and the end result is higher quality.

4.1 Availability

The functionality is available at the Test Project level. I.e. Administrator should enable it for a specified Test Project (Edit Test Project link in Main window). Otherwise links are not shown.

There are two user levels for this feature. Most roles can view requirements, but not modify them. Refer to User section for more details.

4.2 Requirements Specification Document

Requirements are grouped to one or more System/Software/User Requirement Specifications.

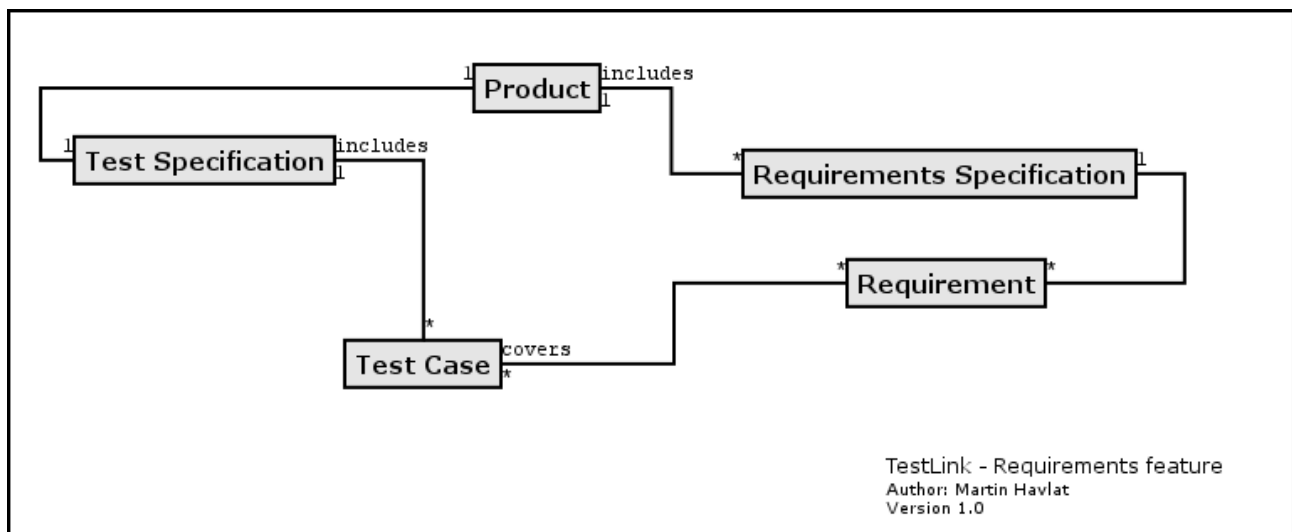


Illustration 6: Dependencies between requirement related objects

Create a document with Requirements:

1. Click Requirements Specification in Main window. The list of Requirement Specifications is shown.
2. Press Create button to create a document.
3. Adjust `Title`, `Scope` and eventually `Count of Test Cases`. The last parameter is used for statistics. Use only if you have a valid Requirement document but not all requirements are available at the moment in TestLink. Default value '0' means that the current count of requirements in a specification is used.
4. Press Create button to add data to database. You can see the title of your new created document in the table of list of Requirement Specifications.
5. Click the title of document for next work. The Requirement Specification window is shown.

Each Requirement Specification has own statistics and report related to included data.

All Specifications can be printed using the "Print" button in the "Requirement Specification" window. Administrator can define company, copyright and confident text via configuration files.

4.3 Requirements

Each requirement has `Title`, `Scope` (html format) and `Status`. `Title` must be unique and has max. 100 characters. `Scope` parameter is text in HTML format. `Status` can have the values VALID or NOT_TESTABLE. A NOT_TESTABLE requirement is not counted in metrics.

Requirements could be created/modified or deleted manually via TestLink interface or imported as CSV file.

Import requirements

TestLink supports two types of CSV. The first 'simple' is composed from title and scope in each row. The second 'Export from Doors' tries to detect the header and choose the correct fields. Import compares titles and tries to resolve conflicts. There are three ways to do this: update, create requirements with same title and skip adding the conflicted ones.

Requirements to Test Case relation

Test Cases are related with software/system requirements as * to *. I.e. you can assign one or more Test Cases to one Requirement and one or more requirements could be covered by one Test Case. User can assign Requirements to Test Cases via the Assign Requirements link in the Main window.

The coverage of the Test Specification could be viewed via pressing the Analyse button in the Requirement Specification window.

Requirement based Report

Navigate to Reports and Metrics menu. There is a Requirements based Report link. Requirements in the current Requirement Specification and Test Plan are analysed for this report. The latest results of Test Cases (available in Test Plan) are processed for each requirement. The result with the highest priority is applied for the requirement. Priorities, from the highest to lowest, are: Failed, Blocked, Not Run and Passed.

Example of requirement coverage

A requirement is covered by three Test Cases. Two of them are included in the current Test Suite. One passed and one was not tested for the Build 1. Now Requirement has overall result of Not Run. Second Test Case was tested with Build 2 and passed. So

Requirement passed too.

5 Test Plans

A record of the test planning process detailing the degree of tester involvement, the test environment, the Test Case design techniques and test measurement techniques to be used, and the rationale for their choice.

Test Plans are the basis for Test Case execution. Test Plan contains name, description, collection of chosen Test Cases, Builds, Test Results, milestones, tester assignment and priority definition.

5.1 Create and delete Test Plan

Test Plans may be created from the "Test Plan management" page by users with lead privileges for the current Test Project. Press button "Create" and enter data.

Test Plans are made up of Test Cases imported from a Test Specification at a specific point of time. Test Plans may be created from other Test Plans. This allows users to create Test Plans from Test Cases that exist at a desired point in time. This may be necessary when creating a Test Plan for a patch. In order for a user to see a Test Plan they must have the proper rights. Rights may be assigned (by leads) in the define User/Project Rights section. This is an important thing to remember when users tell you they can't see the project they are working on.

Test Plans may be deleted by users with lead privileges. **Deleting Test Plans permanently deletes both the Test Plan and all of its corresponding data, including Test Cases** (not in Test Specification), **results, etc.** This should be reserved only for special cases. Alternatively, Test Plans may be deactivated on the same page, which suppresses display on selection menus in the "main" page.

5.2 Builds

An user with lead privileges could follow the link "**Build management**" in the main page.

Builds are a specific release of software. Each project in a company is most likely made up of many different Builds. In TestLink, execution is made up of both Builds and Test Cases. If there are no Builds created for a project the execution screen will not allow you to execute. The metrics screen will also be completely blank.

Each Build is identified via title. It includes description (html format) and two states:

- Active / Inactive – defines whether the Build is available for TestLink functionality. Inactive Build is not listed in either execution or reports pages.
- Opened / Closed – defines if *Test Results* can be modified for the Build.

Builds can be edited (via link under a Build title) and deleted (by click on the appropriate "bin" icon) in the table of existing Builds.

5.3 Populating Test Plan - adding Test Cases

Data from multiple Test Projects can be added into one Test Plan. Test Specification data can be filtered by keywords (adjusted in navigation pane).

Once data has been linked into a Test Plan it will be marked with check-mark. If a Test Case has already been imported it will be ignored if it is imported again.

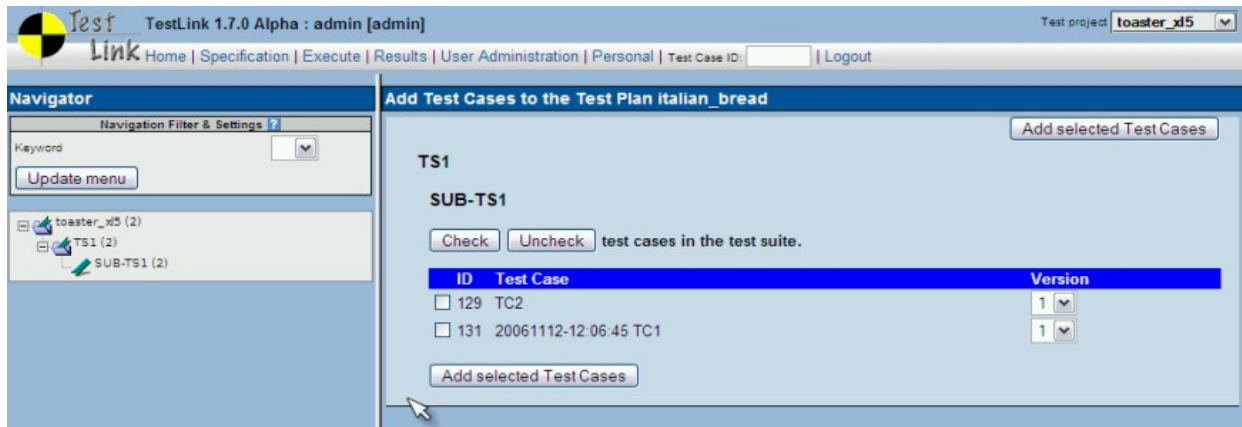


Illustration 7: Frame for Adding Test Cases into Test Plan

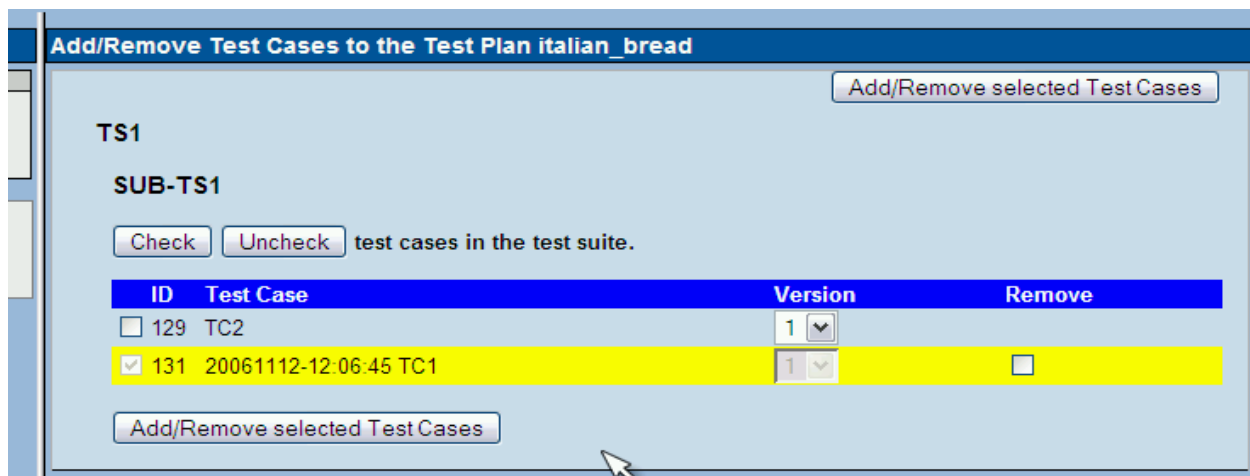
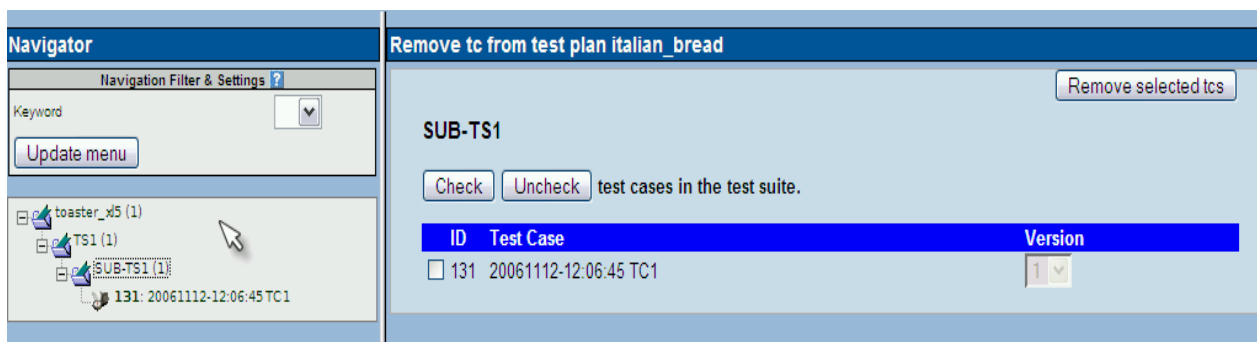


Illustration 8: Frame for modifying content of test cases within Test Plan

Removing Test Cases from Test Plan

Test Cases and Test Suites can be removed from a Test Plan by users with lead permissions from the "Remove Test Cases" page. Removing data may be useful when first creating a Test Plan since there are no results. However, removing Test Cases will cause the loss of all results associated with them. Therefore, extreme caution is recommended when using this functionality.



Tree in left pane, shows only the Test Cases present in Test Plan.

5.4 Test execution assignment

Test execution assignment affects both the execution and metrics screens. In the execution screen users have the ability to sort the executable Test Cases to see the ones they have been assigned. In the main metrics screen there is a table that shows the remaining Test Cases by tester. If there are no Test Case tester assigned it defaults to none.

Assigning Test Case execution tasks for test plan TP1			
TS1			
Bulk user assignment <input type="text"/> <input type="button" value="Do"/>			
<input checked="" type="checkbox"/> ID	Test Case	Version	User
<input type="checkbox"/> 3	TC1	1	havlatm
<input type="checkbox"/> 5	TC2	1	admin

A Tester can also see the metrics of his/her own executed tests in main page if these metrics are enabled (see Installation manual).

5.5 Prioritize testing

Assigning priority are optional feature. Generally, some features are new or heavy development was done. It should receive more attention. On the other side some features was nearly untouched in development phase and only test with the highest importance make sense to execute.

Configuration: This feature must be allowed on Test Project level by administrator ("Test project management" link on main page).

TestLink gives users the ability to assign **Importance** to Test Cases at the design time. There are three levels: Low, Medium (default) and High.

Urgency is another parameter that affects resulted **Test Priority**. Urgency is defined for a certain Test Plan. Follow the link "Set Urgency" at the right side of main page. There are also three levels: Low, Medium (default) and High.

Urgency is not copied if you copy a Test Plan. A different set of urgency is expected.

TestLink combines both these attributes into **Priority** levels High, Medium and Low. Priority should help to choose right set of execution to test at first. Test report "Test Plan metrics" includes prioritized results then. Milestones allows to define goals for these three levels as well.

5.6 Milestones

Test leader can define a milestone for certain date with a goal of expected percentage of finished tests. This goal could be defined for three levels of priority in the case that test prioritization is allowed.

See "Test Plan metrics" report to check a satisfaction of these goals.

Milestones for Test Plan TP1

Milestone was successfully added

New Milestone

Milestones must be created at today's date or greater

Name:

Second circle

Target Date:

10

09

2007

% A Priority:

90

% B priority:

50

% C priority:

0

Create

Existing Milestones

Name	Target Date	% A Priority	% B prior
One circle more	13/09/2007	50	50

Illustration 9: Test leader can define one or more milestones for a Test Plan

6 Test Execution

6.1 General

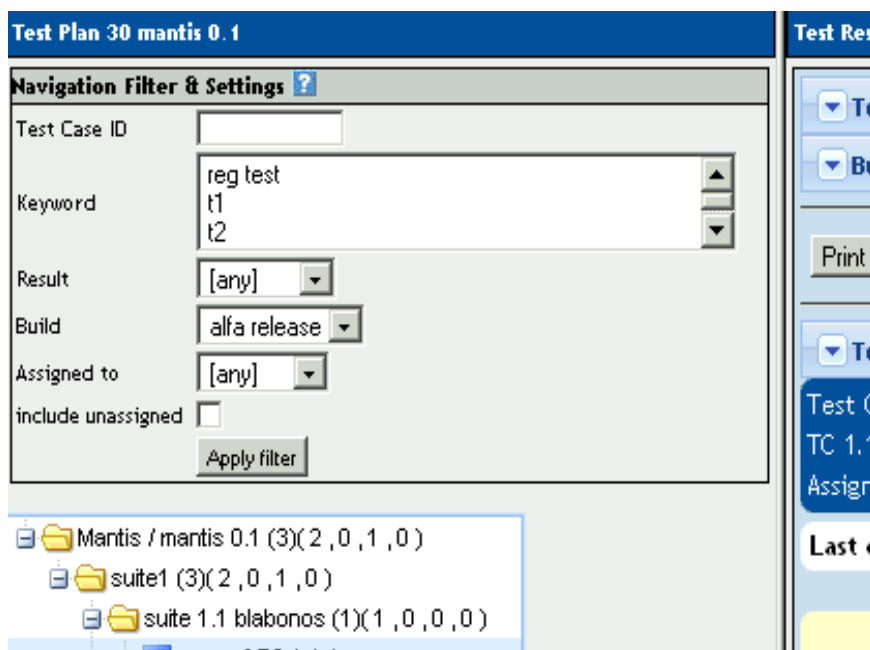
Test execution is available after:

1. A Test Specification is written.
2. A Test Plan is created.
3. Test Cases are added into Test Plan.
4. At least one Build is created.
5. Testers have appropriate rights for execution to work with the this Test Plan.

Select the "Execute" link in top menu or "Execute Tests" link in the *Main page* to navigate to the "Test Execution" window. The left pane allows navigation in Test Cases via a tree menu, and set-up settings and filters. The execution window shows relevant information and allow to add test results.

6.2 Navigation and settings

The navigation pane consists of a 'Filter & Settings' box and a tree menu with Test Case Suite.



Define a tested Build

Users should specify one from all active Builds to add results. The latest build is set by default. Build label specified exact package of product under test for tracking purpose. Each Test Case may be run more times per a Build. However it's common practise that just one testing round is executed against a Build for a test case.

Builds can be created by Test Leader using the Create New Build page.

Search a Test Case

Users can specify exact the Test Case identifier to faster navigation.

Filtering Test Cases

This table allows the user to filter Test Cases for smart navigation before they are executed. You must hit the "Apply filter" button to propagate a new filter settings.

- **Tester:** Users can filter Test Cases by their tester. There is also check-box to see both the chosen tester and unassigned Test Cases.
- **Keyword:** Users can filter Test Cases by keyword. See [3.3 Keywords](#).
- **Result:** Users can filter Test Cases by results. Results are what happened to that Test Case during a particular Build. Test Cases can pass, fail, be blocked, or not be run.
- **Test priority:** Users can prioritize Test Cases.

Tree menu

The tree menu in navigation pane shows the chosen list of Test Cases in the Test Plan. It allows to open appropriate Test Case(s) for test execution in the right frame. Test Suites in the menu are enhanced by short test status overview behind a title (count of test cases, passed, failed, blocked and not-executed counter) coloured by results.

The tree menu could be coloured for some types of used menu component only. By default the tree will be sorted by the results for [the defined Build](#) that is chosen from the drop-down box.

Example TC coloured according to the Build:

User selects Build 2 from the drop-down box and doesn't check the "most current" check box. All Test Cases will be shown with their status from Build 2. So, if Test Case 1 passed in Build 2 it will be coloured green.

A second possibility is that the menu is coloured according to the latest Test Result.

Example TC coloured according to the latest result

User selects Build 2 from the drop-down box and this time checks the "most current" check box. All Test Cases will be shown with most current status. So, if Test Case 1 passed in Build 3, even though the user has also selected Build 2, it will be coloured green.

6.3 Test execution

Test Status

Execution is the process of assigning a result (pass, fail, blocked) to a Test Case for a specific Build. A 'Blocked' Test Case is not possible to test for some reason (e.g. a problem in configuration disallows to run a tested functionality).

Insert Test Results

Test Results screen is shown via click on an appropriate Test Suite or Test Case in navigation pane. The title shows the current Build and owner. The coloured bar indicate status of the Test Case. Yellow box includes test scenario of the Test Case.

Test Results on Build BU1

Test plan notes

Build's notes

Bulk TC status management

Print

Show only last execution

Testsuite SUB-TS1

Test Case ID 131 :: Version: 1

20061112-12:06:45 TC1

Execution history

Date	Tested by	Status	Notes	attachments
12/11/2006 19:45:21	Testlink Administrator	Blocked		
12/11/2006 19:45:17	Testlink Administrator	Failed		
12/11/2006 19:45:13	Testlink Administrator	Pass		
12/11/2006 19:45:04	Testlink Administrator	Failed		

Illustration 10: Frame with several results of one Test Case

Test plan notes

Build's notes

Bulk TC status management

Print

Show only last execution

Testsuite SUB-TS1

Illustration 11: User can select to print only the last result

Print

Show full execution history

Testsuite SUB-TS1

Test Case ID 131 :: Version: 1

20061112-12:06:45 TC1

Last execution for this build

Date	Tested by	Status	Notes	attachments
12/11/2006 19:45:21	Testlink Administrator	Blocked		

Illustration 12: The last result could be printed only

The indication that the Test Case was updated or deleted in test Specification is not supported after 1.5 version.

Updated Test Cases: TL 1.0.4 version has indication by flag, that is missing from 1.6 version. If users have the proper rights they can go to the "Update modified test case" page through the link on main page. It is not necessary for users to update Test Cases if there has been a change (newer version or deleted).

7 Test Reports and Metrics

Test Reports and Metrics are accessed by clicking the "Results" or "Test reports and Metrics" links on the main page. Reports and Metrics are based on the selected Test Plan (from combo box menu). The page that is displayed to the user includes:

The **right pane** will be populated with instructions on how to use the controls and how each report is produced.

The **left pane** is used for navigating to each report and operating controls which effect how reports behave and are displayed. The button "Print" initializes printing of the right pane (no navigation will be printed).

All test reports (except charts) can be generated in 1 of 3 **format** :

1. Normal - report is displayed in web page (html)
2. MS Excel – report exported to Microsoft Excel
3. HTML Email – report is emailed to user's email address

There are currently 9 separate reports to choose from, their purpose and function are explained below. Currently, there are no reports which compile results across multiple Test Plans.

7.1 General Test Plan Metrics

This page shows you only the most current status of a Test Plan by test suite, owner, milestone, priority and keyword. In addition there are also basic metrics for all enabled builds.

The most "current status" is determined by the most recent Build Test Cases were executed on. For instance, if a Test Case was executed over multiple Builds, only the latest result is taken into account. **"Last Test Result"** is a concept used in many reports, and is determined as follows :

- 1) The order in which Builds are added to a Test Plan determines which Build is most recent. The results from the most recent Build will take precedence over older Builds. For example, if you mark a test as "fail" in Build 1, and mark it as "pass" in Build 2, it's latest result will be "pass".
- 2) If a Test Case is executed multiple times on the same Build, the most recent execution will take precedence. For example, if Build 3 is released to your team and tester 1 marks it as "pass" at 2PM, and tester 2 marks it as "fail" at 3PM – it will appear as "fail".
- 3) Test Cases listed as "not run" against a Build are not taken into account. For example, if you mark a case as "pass" in Build 1, and don't execute it in Build 2, it's last result will be considered as "pass".

The following tables are displayed :

Results by top level Test Suites

Lists the results of each top level suite. Total cases with status are listed: passed, failed, blocked, not run, and percent completed. A "completed" Test Case is one that has been marked pass, fail, or block. Results for top level suites include all children suites.

Results by Build

Lists the execution results for every Build. For each Build, the total Test Cases, total pass, % pass, total fail, % fail, blocked, % blocked, not run, and %not run are displayed. If a Test Case has been executed twice on the same Build, the most recent execution will be taken into account.

Results By Keyword

Lists all keywords that are assigned to cases in the current Test Plan, and the results associated with them.

Example :

Keyword	Total	Passed	Failed	Blocked	Not run	Completed [%]
P3	1128	346	47	55	680	39.72
P2	585	372	25	31	157	73.16
P1	328	257	6	51	14	95.73

Results by owner

Lists each owner that has Test Cases assigned to them in the current Test Plan. Test cases which are not assigned are tallied under the "unassigned" heading.

Example:

Tester	Total	Passed	Failed	Blocked	Not run	Completed [%]
Dominika	579	217	34	47	281	51.47
Mohammad	246	82	9	2	153	37.80
unassigned	35	19	0	1	15	57.14
Ken	289	110	1	21	157	45.67
Mallik	430	269	5	18	138	67.91
Ali	227	123	28	13	63	72.25
Mike	24	22	0	0	2	91.67
Alex	272	155	1	36	80	70.59

7.2 Query Metrics

This report consists of a query form page, and a query results page which contains the queried data.

Query Form Page:

User is presented with a query page with 4 controls. Each control is set to a default which maximizes the number of Test Cases and Builds the query should be performed against. Altering the controls allows the user to filter the results and generate specific reports for specific owner, keyword, suite, and Build combinations.

keyword – 0->1 keywords can be selected. By default – no keyword is selected. If a keyword is not selected, then all Test Cases will be considered regardless of keyword assignments. Keywords are assigned in the Test Specification or Keyword Management pages. Keywords

assigned to Test Cases span all Test Plans, and span across all versions of a Test Case. If you are interested in the results for a specific keyword you would alter this control.

owner – 0->1 owners can be selected. By default – no owner is selected. If an owner is not selected, then all Test Cases will be considered regardless of owner assignment. Currently there is no functionality to search for “unassigned” Test Cases. Ownership is assigned through the “Assign Test Case execution” page, and is done on a per Test Plan basis. If you are interested in the work done by a specific tester you would alter this control.

top level suite - 0-> n top level suites can be selected. By default – all suites are selected. Only suites that are selected will be queried for result metrics. If you are only interested in the results for a specific suite you would alter this control.

Builds – 1->n Builds can be selected. By default – all Builds are selected. Only executions performed on Builds you select will be taken into account when producing metrics. For example – if you wanted to see how many Test Cases were executed on the last 3 Builds – you would alter this control.

Keyword, owner, and top level suite selections will dictate the number of Test Cases from your Test Plan that are used to compute per suite and per Test Plan metrics. For example, if you select owner=“Greg”, Keyword=“Priority 1”, and all available test suites – only Priority 1 Test Cases assigned to Greg will be considered. The “# of Test Cases” totals you will see on the report will be influenced by these 3 controls.

Build selections will influence whether a case is considered “pass”, “fail”, “blocked”, or “not run”. Please refer to “**Last Test Result**” rules as they appear above.

Press the “submit” button to proceed with the query and display the output page.

Query Report Page:

The report page will display:

1. the query parameters used to create the report
2. totals for the entire Test Plan
3. a per suite breakdown of totals (sum / pass / fail / blocked / not run) and **all** executions performed on that suite. If a Test Case has been executed more than once on multiple Builds – all executions will be displayed that were recorded against the selected Builds. However, the summary for that suite will only include the “Last Test Result” for the selected Builds.

7.3 Blocked, Failed, and Not Run Test Case Reports

These reports show all of the currently blocked, failing, or not run Test Cases. “**Last Test Result**” logic (which is described above under General Test Plan Metrics) is again employed to determine if a Test Case should be considered blocked, failed, or not run. Blocked and failed Test Case reports will display the associated bugs if the user is using an integrated bug tracking system.

7.4 Test Report

View status of every Test Case on every Build. If a Test Case was executed multiple times on the same Build – the most recent execution result will be used. It is recommend to export this report to Excel format for easier browsing if a large data set is being used.

7.5 Charts

This report page requires your browser have a flash plug-in. “Last Test Result” logic is used for all 4 charts that you will see. The graphs are animated to help the user visualize the metrics from the current Test Plan.

The four charts provide are :

1. Pie chart of overall pass / fail / blocked / and not run Test Cases
2. Bar chart of Results by Keyword
3. Bar chart of Results By Owner
4. Bar chart of Results By Top Level Suite

The bars in the bar charts are coloured such that the user can identify the approximate number of pass, fail, blocked, and not run cases.

It uses flash technology provided by <http://www.maani.us> to display results in a graphical format.

7.6 Total Bugs For Each Test Case

This report shows each Test Case with all of the bugs filed against it for the entire project. This report is only available if a Bug Tracking System is connected.

7.7 Requirements based report

This report is available if requirements are allowed for the current Test Project. Report is generated against one Requirement Specification document chosen from combo box menu.

There are two sections: metrics and results overview.

The following metrics are available:

- Total number of requirements
- Requirements within TestLink
- Requirements covered by Test Cases
- Requirements not covered by Test Cases
- Requirements not covered or not tested
- Requirements not tested

Requirements are divided into four sections. Each requirement is listed together with all related Test Cases (coloured according to Test Case result):

- Passed Requirements
- Failed Requirements
- Blocked Requirements
- Not-executed Requirements

7.8 How to add a new report

Copy one of the current reports and modify it according to your need. Don't forget that we use templates for rendering (`<testlink_root>/gui/templates/<report_name>.tpl`) and logic (`<testlink_root>/lib/results/<report_name>.php`). We recommend reusing existing functions to collect data for report instead of creating new ones.

Edit `<testlink_root>/lib/results/resultsNavigator.php` to add a link to your new report. There is an array, that could be easily enhanced. You must add a new URL and "name keyword"³ of report.

You can modify CSS style for a report. We suggest creating new classes instead of modifying

³ The string must be defined in locale/en_gb/string.txt

current ones (to avoid unwanted changes in other pages).

If you contribute your new report(s) via our tracker, you can find it also in the next versions ... otherwise you risk that it will not work for the next main release.

8 Administration

8.1 User account settings

Every guest can create own account on login page.

The auto-create feature can be disabled by configuration. There is also configurable default role parameter ('guest' by default).

Every user on the system will be able to edit their own information via the Account settings window ('**Personal**' link in menu bar). He can modify own Name, email address, password and generate API key (if enabled).

TestLink allows users with administrator rights to create, edit, and delete users within the system. However, TestLink does not allow administrators to view or edit user's passwords. If users forget their passwords there is link on the login screen, that will mail the user their password based upon their user name and the email address they entered.

8.2 Role Permissions

User can see the current role in top line of TestLink window. Each users has 'generic role' that applies on all projects. Administrator can assign specific project role for a particular project. Users can have modified role for particular Test Plan as well.

For example Jack could be guest by default (can see all test cases and reports) and Test leader on project 'XYZ' to lead this project testing.

TestLink is built with 6 different default permission levels built in. These permission levels are as follows:

- **Guest:** A guest only has permission to view Test Cases, reports and metrics. He cannot modify anything.
- **Test Executor:** A tester has permissions to see and run tests allocated to them.
- **Test Designer:** A user can fully work (view and modify) with Test Specification and Requirements.
- **Test Analyst:** A tester can view, create, edit, and delete Test Cases as well as execute them. Testers lack the permissions to manage Test Plans, manage Test Projects, create milestones, or assign rights. (initially Senior tester).
- **Test Leader:** A lead has all of the same permissions as a Tester but also gains the ability to manage Test Plans, assign rights, create milestones, and manage keywords.
- **Administrator:** An admin has all possible permissions (leader plus the ability to manage Test Projects and users).

Changing of these rights is handled by the user administration link which is accessible by administrators. Admin can add a new custom roles and modify existing ones via GUI.

See Developer's Guide to understand more.

If you view the table you will see rows for each of the permissions levels (guest ,tester, senior tester, leader, admin). The second column holds all of the different rights levels which will be defined below. These levels have been determined as standard for use but they can be edited to define new roles (for experienced administrator). The user table contains a foreign key that points to the appropriate permission level in the rights table.

Note: Test Plan related features needs also assign a Test Plan to be available. See [Test](#).

Plan Assignment.

Case study – restrict access by default

Situation

In our organization we would like to restrict access to projects unless it is specifically granted. We currently have about 150 users with just over 90 different project. Many of the users are not QA, but are business analysts, and in some cases end users doing UAT. From what I can tell, all rights are inherited based on how the user was set-up.

But we don't want a Business Analyst who is working on only one project have access to all 90.

Solution

Set these users with the global role *<No rights>* and granted an appropriate role on a Test project or Test plan base. In `custom_config.inc.php` you can also set the default role id to *<No rights>*:

```
$tlCfg->default_roleid = TL_ROLES_NO_RIGHTS;
```

You can also would like to rename a role "No rights" to something more polite.

8.3 Test Plan assignment to users

Users can execute only assigned Test Plans. In order to gain Test Plan permissions a user with lead or admin status must give them rights through the "Define user/project rights" link under "Test Plan Management".

All users in the system will by default not have permissions to view newly created Test Plans (except for the Test Plan creators who can give themselves permissions at creation). Zero Test Plan permissions means that users will not see any Test Plans in the Test Plan drop-down box on main screen.

However he can browse reports.

8.4 Custom Fields

Users can extend the most important objects (Test cases, suites, plans) in TestLink with customized fields. After you have created a Custom Field, you need to assign it to the Test Project in which do you want to use it. So each project have own set of additional fields.

Common practice: Test case CF could be: review notes or status, estimated or real execution time, platform (OS, HW, ...), test script name or owner, etc.

Navigate to "Custom field management" link on main page. This page is the base point for managing custom fields pool. It lists the custom fields defined in the system. Custom field definitions are system wide, i.e., you can not define two custom fields with same field ID.

The "Create" button take you to a page where you can define the details of a custom field. These include it's name, type, value and availability/display information. On the edit page, the following information is defined to control the custom field:

- **name**
- **type** - possible values are listed below.
- **Value constraints** (Possible values, default value, regular expression, minimum length, maximum length).
- **Availability/Display control** (where the field will show up and must be filled in

All fields are compared in length to be greater than or equal to the minimum length, and less than or equal to the maximum length, unless these values are 0. If the values are 0, the check is skipped. All fields are also compared against the regular expression. If the value matches

the expression, then the value is stored. For example, the expression `"/^-[0-9]]*$/"` can be used to constrain an integer.

There is data format verification for types numeric, float and email. On submit (when created a Test case or while executing) CF of this type are controlled. Empty value OK, but if value NoT empty but do not conform a regular expression (NOT USER DEFINED but coded) submit will be blocked with message on screen.

The table below describes the field types and the value constraints.

Type	Field Contents	Value Constraints
String	text string up to 255 characters	
Numeric	an integer	
Float	a floating point number	
Email	an email address string up to 255 characters	When displayed, the value will also be encapsulated in a mailto: reference.
Checkbox	zero or more of a list of text strings	Enter the list of text strings separated by " " (pipe character) in the Possible Values field. The Default value should match one of these strings as well. This will be displayed as a list of text strings with a checkbox beside them.
List	one of a list of text strings	Enter the list of text strings separated by " " (pipe character) in the Possible Values field. The Default value should match one of these strings as well. This will be displayed as a multi-line dropdown menu.
Multiselection List	zero or more of a list of text strings	Enter the list of text strings separated by " " (pipe character) in the Possible Values field. The Default value should match one of these strings as well. This will be displayed as a multi-line dropdown menu.
Date	text string defining a date	This is displayed as a set of dropdown menus for day, month, and year. Defaults should be defined in yyyy-mm-dd format.
Text Area	Rich text	

Table 1: Custom field types

The display entries are used as follows:

Entry	Meaning
Display on test specification	If checked, the field will be shown on Test Specification tree
Enable on test specification	If checked, the field will be editable (assign/change) on Test Specification
Display on test execution	If checked, the field will be shown on Test Execution
Enable on test execution	If checked, the field will be editable on Test Execution
Available for	Test case, Test plan or Test suite

Table 2: Custom fields: Availability/Display Attributes

Example 1.

Custom Field : Additional Notes
Type : string

*applicable to test suites, to be edited ONLY during
Test Case specification, but useful to be seen during test execution.*

show on design = YES
enable on design = YES
show on execution = YES
enable on execution = NO

Example 2.

Custom Field : Operating System
Type : list

*applicable to Test Cases, to be edited ONLY during
Test Case EXECUTION, unused during Test Case DESIGN.*

show on design = NO
enable on design = NO
show on execution = YES
enable on execution = NO

*Custom field feature has been implemented using a mix functionality from Mantis
(<http://www.mantisbt.org/>) and dotproject (<http://www.dotproject.net/>) models.*

Estimated and real time execution of Test case

Testers could specify a duration of test execution. Test plan report will display overall testing duration then. Create custom field with ID = CF_EXEC_TIME and assign to test cases. Configure it to be displayed and used on execution.

It is possible to summarize estimated execution time. Create custom field with ID = CF_ESTIMATED_EXEC_TIME to define Expected test execution time. It should be used in Test Specification and possibly displayed in execution.

Localizations: Use " . " (dot) as decimal separator or sum will be wrong.

9 Import and Export data

TestLink supports several ways to share data. See the next table for overview. In addition you can consider to use TestLink API to deliver supported data.

Troubleshooting: No answer for import action? Check size of imported file. There are limits in TestLink configuration and web server settings. Check if DOM module is loaded for your web server.

Item	File format	What you get
Keyword	CSV XML	All Test Project's keywords
Test Project	XML	All Test Suites and Test Cases. You can choose whether to also export assigned keywords.
Test Suite	XML	Test Suite details, All Test Cases and child test suites and Test Cases. You can choose whether to also export assigned keywords.
Test Case	XML	Two types of exports can be done: - Just one Test Case - All Test Cases in Test Suite. You can choose whether to also export assigned keywords.
Requirement	CSV CSV DOORS (*) XML	(*) Only import is supported for this format.

Table 3: Items that can be exported/imported

Limitation: Attached files and custom fields are not exported.

9.1 Import/Export Keywords

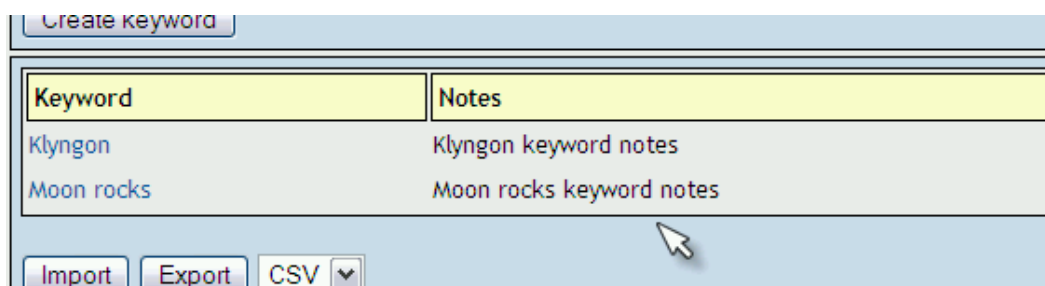


Illustration 13: Keywords frame includes buttons for import and export

Example of CSV "Keyword;Notes":

```
Klyngon;Klyngon keyword notes
Moon rocks;Moon rocks keyword notes
```

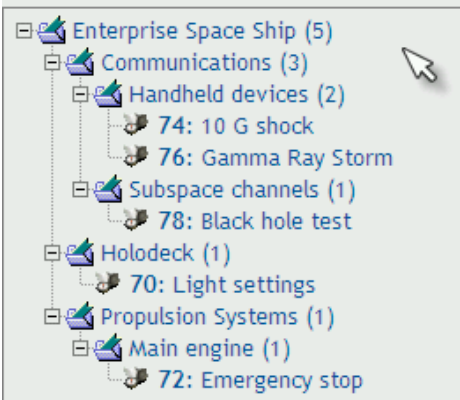


Example of XML with keywords:

```
<?xml version="1.0" encoding="UTF-8"?>
<keywords>
  <keyword name="Klyngon">
    <notes>
      <![CDATA[Klyngon keyword notes]]>
    </notes>
  </keyword>
  <keyword name="Moon rocks">
    <notes>
      <![CDATA[Moon rocks keyword notes]]>
    </notes>
  </keyword>
</keywords>
```

9.2 Export/Import Test Project

User can import or export Test Project including Description of the project, Test Specification and Keywords. The next two pictures show tree menu with data and the same data as XML file.



```
<?xml version="1.0" encoding="UTF-8"?>
<testsuite name="">
  <details><![CDATA[]]></details>
  <testsuite name="Communications">
    <details><![CDATA[<p>Communication Systems of all types</p>]]></details>
    <testsuite name="Hand-held devices">
      <details><![CDATA[]]></details>
      <testcase name="10 G shock">
        <summary><![CDATA[]]></summary>
        <steps><![CDATA[]]></steps>
        <expectedresults><![CDATA[]]></expectedresults>
      </testcase>
      <testcase name="Gamma Ray Storm">
        <summary><![CDATA[]]></summary>
        <steps><![CDATA[]]></steps>
        <expectedresults><![CDATA[]]></expectedresults>
      </testcase>
    </testsuite>
    <testsuite name="Subspace channels">
      <details><![CDATA[<p>Only basic subspace features</p>]]></details>
      <testcase name="Black hole test">
        <summary><![CDATA[]]></summary>
        <steps><![CDATA[]]></steps>
        <expectedresults><![CDATA[]]></expectedresults>
      </testcase>
    </testsuite>
  </testsuite>
```



```

</testsuite>
</testsuite>
<testsuite name="Holodeck">
  <details><![CDATA[]]></details>
  <testcase name="Light settings">
    <summary><![CDATA[]]></summary>
    <steps><![CDATA[]]></steps>
    <expectedresults><![CDATA[]]></expectedresults>
  </testcase>
</testsuite>
<testsuite name="Propulsion Systems">
  <details><![CDATA[]]></details>
  <testsuite name="Main engine">
    <details><![CDATA[]]></details>
    <testcase name="Emergency stop">
      <summary><![CDATA[]]></summary>
      <steps><![CDATA[]]></steps>
      <expectedresults><![CDATA[]]></expectedresults>
    </testcase>
  </testsuite>
</testsuite>
</testsuite>

```

9.3 Import/Export Test suite



XML Example – Test Suite without keywords

```

<?xml version="1.0" encoding="UTF-8"?>
<testsuite name="Hand-held devices">
  <details><![CDATA[]]></details>
  <testcase name="10 G shock">
    <summary><![CDATA[]]></summary>
    <steps><![CDATA[]]></steps>
    <expectedresults><![CDATA[]]></expectedresults>
  </testcase>
  <testcase name="Gamma Ray Storm">
    <summary><![CDATA[]]></summary>
    <steps><![CDATA[]]></steps>
    <expectedresults><![CDATA[]]></expectedresults>
  </testcase>
</testsuite>

```

XML Example – Test Suite with keywords

```

<?xml version="1.0" encoding="UTF-8"?>
<testsuite name="Hand-held devices">
  <details><![CDATA[]]></details>
  <testcase name="10 G shock">
    <summary><![CDATA[]]></summary>
    <steps><![CDATA[]]></steps>
    <expectedresults><![CDATA[]]></expectedresults>
    <keywords>
      <keyword name="Klyngon">
        <notes><![CDATA[Klyngon keyword notes]]></notes>
      </keyword>
    </keywords>
  </testcase>
  <testcase name="Gamma Ray Storm">

```

```

<summary><![CDATA[]]></summary>
<steps><![CDATA[]]></steps>
<expectedresults><![CDATA[]]></expectedresults>
<keywords>
  <keyword name="Klyngon">
    <notes><![CDATA[Klyngon keyword notes]]></notes>
  </keyword>
  <keyword name="Moon rocks">
    <notes><![CDATA[Moon rocks keyword notes]]></notes>
  </keyword>
</keywords>
</testcase>
</testsuite>

```

9.4 Just one Test Case

ID 78 :: Test Case Black hole test

Version 1

Summary

This procedure must be done once a week, with this safety device disabled:

- X45HH
- YY89-000-JI

Steps

Preset bias to 0

Enable long range communications control

Simulate black hole interference

Expected Results

Main Results	
Spin value	9.9
Opposite Angle	18 rad

Keywords: Moon rocks

Created on 27/07/2007 15:16:52 by admin

Last modified on 27/07/2007 16:16:33 by admin

Example of XML file:

```

<?xml version="1.0" encoding="UTF-8"?>
<testcases>
  <testcase name="Black hole test">
    <summary>
      <![CDATA[<p>This procedure must be done once a week, with this safety device disabled:</p>
<ol><li>X45HH</li><li>YY89-000-JI</li></ol>]]>
    </summary>
    <steps><![CDATA[
      <p>Preset bias to 0</p>
      <p>Enable <strong>long range</strong> communications control</p>
      <p>Simulate black hole interference</p>]]> </steps>
    <expectedresults><![CDATA[
      <table width="200" cellspacing="1" cellpadding="1" border="1">
        <caption>Main Results</caption>
        <tbody>
          <tr><td>Spin value</td><td>9.9</td></tr>
          <tr><td>Opposite Angle</td><td>18 rad</td></tr>
          <tr><td>&nbsp;</td><td>&nbsp;</td></tr>
        </tbody>
      </table>]]>
    </expectedresults>
    <keywords>
      <keyword name="Moon rocks">

```

```

    <notes><![CDATA[Moon rocks keyword notes]]></notes>
  </keyword>
</keywords>
</testcase>
</testcases>

```

9.5 All Test Cases in test suite

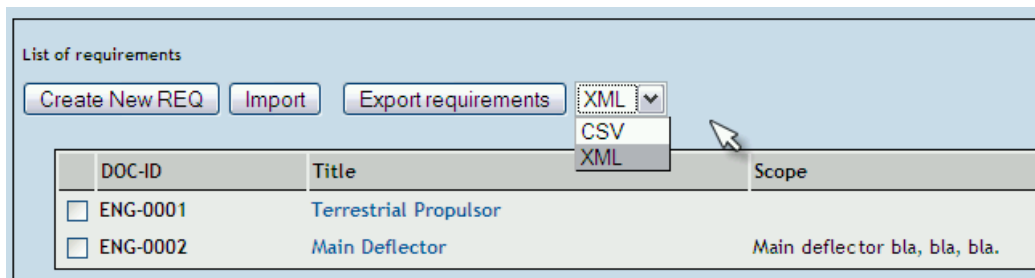


```

<?xml version="1.0" encoding="UTF-8"?>
<testcases>
  <testcase name="10 G shock">
    <summary><![CDATA[]]></summary>
    <steps><![CDATA[]]></steps>
    <expectedresults><![CDATA[]]></expectedresults>
  </testcase>
  <testcase name="Gamma Ray Storm">
    <summary><![CDATA[]]></summary>
    <steps><![CDATA[]]></steps>
    <expectedresults><![CDATA[]]></expectedresults>
  </testcase>
</testcases>

```

9.6 Import/Export Software Requirements



CSV file includes "Identifier of document, title, description".

Example of CSV file:

```

ENG-0001, Terrestrial Propulsor,
ENG-0002, Main Deflector, "<p>Main deflector bla, bla, bla.</p>"

```

Example of XML file:

```

<?xml version="1.0" encoding="UTF-8"?>
<requirements>
  <requirement>
    <docid><![CDATA[ENG-0001]]></docid>
    <title><![CDATA[Terrestrial Propulsor]]></title>
    <description><![CDATA[]]></description>
  </requirement>

```

```

<requirement>
  <docid><![CDATA[ENG-0002]]></docid>
  <title><![CDATA[Main Deflector]]></title>
  <description><![CDATA[<p>Maindeflector bla, bla, bla.</p>]]></description>
</requirement>
</requirements>

```

Import rich text format requirements via DocBook

There is limited support of import for documents in such formats as is MSWord or openoffice. You can export original document as DocBook (tested with openoffice2 and 3). Choose import button for your SRS in TestLink. Select type "DocBook".

The exported file is XML. Basic element for default settings could be the next:

```

...
<sect3>
  <title>Title</title>
  ...
  <para>Description</para>
  ...
  <orderedlist>
    <listitem>Item</listitem>
    ...
  </orderedlist>
  ...
  <informaltable>
    <tgroup>
      <thead>
        <row> ... <entry></entry> ... </row>
      </thead>
      <tbody>
        <row> ... <entry></entry> ... </row>
      </tbody>
    </tgroup>
  </informaltable>
  ...
</sect3>

```

TestLink uses such element as data for just one requirement. This element is defined via constant `DOCBOOK_REQUIREMENT` (check the code). i.e. `<sect3>` is default but could be modified.

Each requirement content is maintain the following way:

title – receive text in tag `<title>`

req_doc_id – parse title for the first two words and add counter. You can modify regular expression directly in code. Default is `"[a-zA-Z_]*[0-9]*"`.

description – parse following elements after title (`<para>`, `<orderedlist>`, `<informaltable>`, etc.). DocBook elements are modified to HTML tags. Unknown ones are omitted.

*Warning: the original code could be modified to fit your structure of DocBook. Check `requirement.inc.php` (**function** `importReqDataFromDocBook($fileName)`) and related constants.*

Warning: generated `REQ_DOC_ID` is danger for the case of update. Because it's generated from file content without relation to existing testlink data.

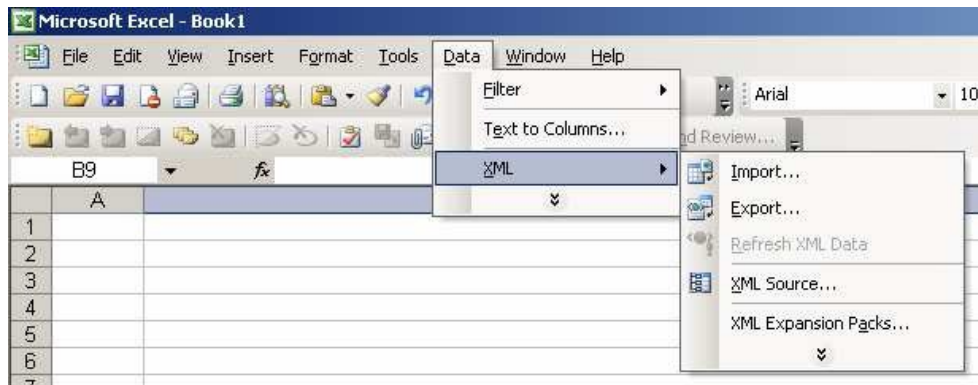
9.7 Import Test Cases from Excel via XML

Creating XML file to import in TestLink

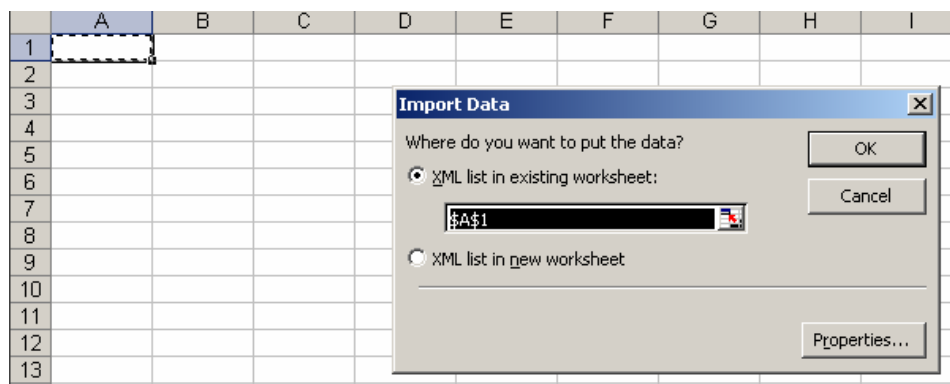
Step 1. Export one or more dummy Test Cases from TestLink into a XML file.

Step 2. Open new blank spread sheet document file.

Step 3. Navigate through menu bar Data > XML > Import & select the sample XML file. It creates appropriate structure in Excel.



Step 4. Then we will get dialog box asking "Where do you want to put the data?"



Step 5. Choose option one "XML list in existing worksheet" with first cell \$A\$1

Step 6. You will be able to see following columns : name, summary, steps & expected results

	A	B	C	D
1	name	summary	steps	expectedresults
2			user has logged in the application. Navigate to 'Accounts' screen by clicking on the Accounts screen	On navigating to 'Accounts' screen, the following view should be displayed. 'My Accounts List' view (By default this view should be displayed). 'All Accounts List' view.
3	ACC 1.1	Whether the user can view / tab.		be loaded.
4	ACC 1.2	Whether the required applet	i) Click on the 'Accounts' screen. ii) Check for applets	It should display all the following applets:

Step 7. Copy your data into this file accordingly & save the file in XML Data (*.xml) format

Step 8. Check your XML file for correctness by opening with the help of internet explorer.

```

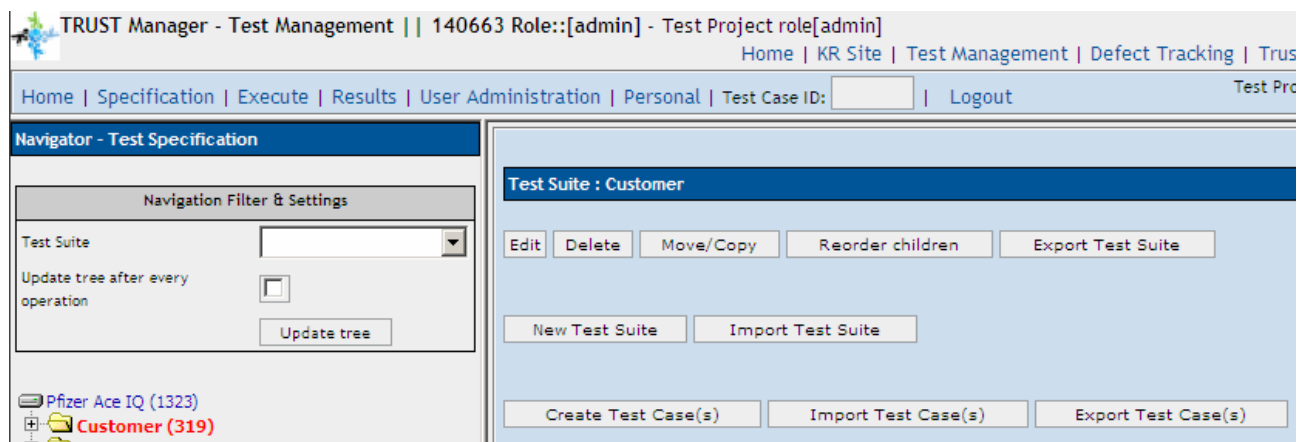
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <testcases>
- <testcase name="ACC 1.1">
  <summary>Whether the user can view Accounts List View as
  a default view under Accounts screen.</summary>
  <steps>Pre-condition: A valid user has logged in the
  application. Navigate to 'Accounts' screen by clicking on
  the Accounts screen tab.</steps>
  <expectedresults>On navigating to 'Accounts' screen, the
  following view should be displayed. 'My Accounts List'
  view (By default this view should be displayed). 'All
  Accounts List' view.</expectedresults>
</testcase>
- <testcase name="ACC 1.2">
  <summary>Whether the user can view Accounts List View as
  a default view under Accounts screen.</summary>
  <steps>Pre-condition: A valid user has logged in the
  application. Navigate to 'Accounts' screen by clicking on
  the Accounts screen tab.</steps>
  <expectedresults>On navigating to 'Accounts' screen, the
  following view should be displayed. 'My Accounts List'
  view (By default this view should be displayed). 'All
  Accounts List' view.</expectedresults>
</testcase>
</testcases>

```

Importing XML file into TestLink

Step 1. Login in to TestLink > Select your project in dropdown list.

Step 2. Click on Specification > Create New Suite > Select Suite > Click on Import Test Cases



Step 3. Browse for the XML file, submit it and you are done with the Importing.

Revision History:

#	Description	Date	Author
0.x	Documents for TL 1.5 and update for TL 1.6	2005	M. Havlát A. Morsing F. Mancardi
1.0	Converted to OO2 format;	2005/03/12	M. Havlát
1.1	Minor update; FIX 372, 352	2006/02/14	M. Havlát
1.2	Updated as draft for TL 1.7	2006/11/17	M. Havlát
1.3	Removed TL 1.6 terms Added initial information about Custom Fields	2007/03/01	F. Mancardi
1.4	Added content and updated Francisco's "jumpstart_manual" and tl_file_format. General style clean-up and update.	2007/09/06	M. Havlát
1.5	General update and restructuring; added Test Suite chapter; requirements report	2007/12/17	M. Havlát
1.6	Overall language review	2008/01/24	W. Pollans
1.7	Minor update; Added section Import Test Cases from Excel via XML (prepared by Prem)	2008/02/02	M. Havlat
1.8	Update to TL 1.8 draft	2008/04/16	M. Havlat
1.9	Update some new features	2008/07/21 2008/08/18	M. Havlát
1.10	Update for 1.8 RC3	01/15/09	M. Havlát