TaRGeT Software Requirements Specification

Version: 01.10.01

Date: 21-Jul-2009

ABSTRACT: TaRGeT is a tool designed to automatically generate test from suitable input use case documents. This document describes the requirements of this tool.

KEYWORDS: Use Cases, Requirements, Automatic Test Generation

Contents

1. Introduction.. 3

1.1 Purpose 3

1.2 Scope 3

1.3 Target Audience 3

1.4 Abbreviations, Acronyms and Definitions 3

1.5 References 3

1.6 Overview 3

1.7 Problem Reporting Instructions 4

2. Functional Requirements... 5

2.1 Handling Projects 5

2.2 Handling Use Cases Documents 7

2.3 Preferences 9

2.4 On The Fly Generation 10

2.5 Consistency Management 15

2.6 Controlled Natural Language (CNL) 18

2.7 Use Case Editor 19

2.8 Others 21

3. Environment Requirements.. 22

3.1 Compatibility 22

3.2 Software Dependencies 22

4. Tool Workspace.. 23

4.1 Project Window 23

4.2 Menu Bar 23

4.3 Left Panel 24

4.4 Bottom Panel 24

4.5 Main Panel 24

5. Appendix A – Use Case Document Format............................... 25

6. Appendix B – Feature Model.. 29

1. Introduction

1.1 Purpose

The purpose of this document is to describe the requirements for TaRGeT tool.

1.2 Scope

The scope of the document is to cover the functional, non functional and interface requirements of TaRGeT.

1.3 Target Audience

The target audience is the stakeholders that are involved with TaRGeT development and use.

1.4 Abbreviations, Acronyms and Definitions

TaRGeT	Test and Requirements Generation Tool
OS	Operational System
JRE	Java Runtime Environment
JVM	Java Virtual Machine
UI	User Interface
RTM	Requirements Traceability Matrix

1.5 References

[1] Java JRE 6.0, http://java.sun.com/javase/downloads/index.jsp

[2] .NET Framework 3.0 Runtime, http://msdn.microsoft.com/windowsvista/downloads/products/getthebeta/default.aspx#runWinFXApps

[3] TaRGeT Use Case Document

1.6 Overview

TaRGeT is a working environment that is capable to visualize and navigate through input use case documents in order to generate test suites. The inputs and outputs of the tool are organized as a project.

The remainder of this document describes the functional (Section 2), non functional requirements (Section 3) and the workspace of the tool (Section 4). At last, the appendix sections are the future enhancements and the format definition of the use cases used as input for the tool.

The use cases for the requirements detailed in this document are described in [3].

1.7 Problem Reporting Instructions

Problems and corrections to this document should be reported to Michelle Silva (mcms@cin.ufpe.br) or Paulo Borba (phmb@cin.ufpe.br).

2. Functional Requirements

The purpose of this section is to document all the functional requirements of TaRGeT tool that can handle the follow artifacts:

- Use case documents – Word documents that describe the behavior of mobile features. It is the main input of the tool;

- Test suites documents – Excel sheets produced as the outputs of test generation.

2.1 Handling Projects

This subsection presents the requirements related to project handling.

2.1.1 Renaming Artifacts (FR TARGET 0001)

The user shall be able to rename the artifact's files from the tool's interface.

2.1.2 Deleting Artifacts (FR TARGET 0003)

The user shall be able to delete the artifact's files from the tool's interface. A confirmation dialog must be displayed before deleting the artifacts in order to confirm if this is the really intention of the user.

2.1.3 Opening and Editing Project Artifacts (FR TARGET 0005)

The tool shall allow the user to edit the project artifacts using the default editors.

2.1.4 Project folders (FR TARGET 0010)

Each project is placed in a folder (or directory) that contains:

- The project file – control information about the project;

- The use case's folder – stores the input: use case documents;

- The test suite's folder – stores the output: test suites.

The tool shall place the generated test suite file in the appropriate folder of the project.

2.1.5 Refreshing a Project (FR TARGET 0015)

The tool shall allow the refresh of the project. The refresh option shall restore the information displayed in the tool’s view. The user can choose between options “Refresh Automatically” and “Refresh Project”.

- Refresh Automatically: by selecting this option, the refresh shall be done automatically when a use case document is updated, imported or deleted from the project. Also, the project shall be updated when the test suite’s spreadsheets are removed/renamed from the project’s test suite folder.

- Refresh Project: consider the Refresh Automatically option is unselected, the refresh shall occur when, and only when, the user clicks in this option. This command can be also executed using a shortcut.

2.1.6 Creating a new project (FR TARGET 0020)

The user shall be able to create a new test generation project. Every new project is created empty (without use cases or test suites).

The default project’s folder is the folder where the tool is installed. The screen of the creation project is displayed with name and location folder default. The user shall be able to choose the name of the project and where it will be placed.

If a project is created in a directory that already contains information about a previous project, the tool shall ask the user if he/she wants to delete all directory content, before creating a new project.

The user shall be able to create a new project when another project is already open. In this case, the current project will must be closed before creating a new project.

2.1.7 Project and Artifacts Names (FR TARGET 0025)

The name of project and artifacts shall not be empty and shall not contain any of the following characters: ‘/’, ‘\’, ‘:’, ‘*’, ‘?’, ‘“’, ‘<’, ‘>’ and ‘|’. All other characters are allowed.

2.1.8 Closing a project (FR TARGET 0035)

The user shall be able to close a project that is opened. A confirmation dialog must be displayed before closing the project in order to confirm if this is really the intention of the user.

2.1.9 Opening a project (FR TARGET 0040)

The user shall be able to open an existing project. It shall be not allowed to open two projects simultaneously. If the user wants to open an existing project P when another one is already opened, the current project must be closed before.

2.2 Handling Use Cases Documents

This subsection describes the requirements related to how the use case documentation is handled by the TaRGeT. An use case document may contain use cases, as specified in Appendix A – Use Case Document Format. When mentioning a *use case document*, it refers to documents that contain use cases.

2.2.1 Importing Documents (FR TARGET 0100)

Use case documents shall be a Microsoft Word 2003 or XML document (INPUT Feature – See Appendix B). The user shall be able to import one or more validated use case documents to an open project. The imported documents shall be stored in the appropriate folder of the project. Before importing, the use case documents shall be validated according to the rules described in the Appendix A – Use Case Document Format.

2.2.2 Progress Bar (FR TARGET 0110)

The user shall be aware about the progress of the following operations: import document, refresh project and create or open a new project.

2.2.3 Indicating Errors on Use Case Documentation (FR TARGET 0120)

The inadequacies found after a validation in imported documents shall be pointed by the tool according to the following classification:

- Empty mandatory fields (according to Appendix A – Use Case Document Format);
- Duplicated step IDs in the same use case;
- References to step IDs that do not exist;
- Invalid documents (see requirement FR_TARGET_0125).

The warnings indicating errors in imported documents shall be displayed in the specific error area in the tool where the user shall be able to visualize the description, the resource and the location of the errors.

2.2.4 Rejecting Invalid Use Case Documents (FR TARGET 0125)

The tool shall reject a document if it is crashed. A document becomes crashed if the user inserts one of the following errors:

- Missing mandatory field;
- Use case ID or feature ID have invalid value;
- Any field that is not according to the predefined order;
- Contains a field that is not specified;
- Whole structure of the document is empty;
- It does not contain any feature;
- It does not contain any use case in a feature;
- It contains duplicated use case IDs in a same feature;
- It contains duplicated feature IDs;

It means that the document shall be ignored and no information about the document will be considered. If a valid document that is already imported is updated and becomes crashed, it shall be rejected.

Features with the same ID but in different documents shall be allowed. However, even in different documents, the use cases IDs shall be different. For further information about document structure, see Appendix A – Use Case Document Format.

2.2.5 Use Case Preview (FR TARGET 0130)

The tool shall allow the user to preview a specific use case in HTML format within the TaRGeT UI in the main panel or in the browser screen without the TARGET UI.

If there are problems in the previewed use case, the errors shall be highlighted in the HTML preview.

2.2.6 Search Mechanism (FR TARGET 0135)

The tool shall allow the user to perform searches in use case documents. The user shall be able to visualize the search results.

2.3 Preferences

This subsection describes the requirements related to user preferences configuration around test suite content.

2.3.1 Preferences Configuration (FR TARGET 0231)

The tool shall allow the user to setup some preferences, as described below:

- The user shall be able to setup the test case ID format that is shown in the test case document.
- The user shall be able to setup the test case initial ID value. For example, if there is already another test case suite referring to the same feature and it has 10 test cases, the user can select the new test suit to begin from 11 test case id.
- The user shall be able to setup a default value for all Use Case fields that are empty. The default value shall appear in the generated Test Cases.
- The user shall be able to setup which label shall appear before the objective description.

2.3.2 Test cases descriptions configuration (FR TARGET 0232)

The user shall be able to choose how the Use Case description and Flow Description is composed in the test suite. This configuration can be according one of the following options:

- The user shall be able to set to Case Description and Objective fields in test suite to be empty. This choice can be done setting use case description and flow description to None.
- The user shall be able to set to test case descriptions be composed by all use case descriptions that belong to this test. The same option can be done to the test case objective that can be composed by all concatenated flow descriptions (related to flows that compose this test). For example, a test case is composed by steps from different use cases, then, the test case description shall be all use cases descriptions concatenated.
- The user shall be able to set to test case descriptions be composed by only last use case description that belong to this test. The same option can be done to the test case objective that can be composed by only last flow description (related to flows that compose this test). For example, a test case is composed by steps

from different use cases, then, the case description shall be just the last use case description.

2.3.3 Requirements options (FR TARGET 0233)

The tool shall provide an option to show or not the requirements in the respective step of the test case in addition to showing it in the requirements list of the test case.

2.4 On The Fly Generation

This subsection describes the requirements related to the generation of test suites by the tool.

2.4.1 On The Fly Generation (FR TARGET 0242)

The tool shall generate test cases, if and only if there is at least one imported document, and all imported use case documents do not have errors. The imported documents shall also contain at least one feature specification.

The On The Fly Generation shall provide a pre-visualization of the test cases that will be generated. It also shall allow the user to include/exclude test cases that have been previously selected by the filters and visualize this change before the generation.

2.4.2 Import template (FR TARGET 0251)

It shall be possible to import a template before generate the test suite. This template is a spreadsheet in which the system will base to generate the new test suit and some information shall be preserved.

2.4.3 Test Selection (FR TARGET 0234)

The On the Fly Generation shall provide some filters that allow the user to select the test cases according to the requirements, use cases, test purpose and coverage. The test suite shall be the result of filters combination. If the user does not select any filter, the tool shall generate all possible test cases from current use case documents.

2.4.3.1 Test Generation For Specific Requirements (FR TARGET 0145)

Through the Requirements filter, the user shall be able to generate test cases that cover specific requirements. In this case, the tool shall allow the user to select a subset of requirements, all requirements, or none requirements that are referenced in the imported use case documents.

The generation for specific requirement shall be enabled, if and only if there is at least one reference to a requirement in the imported use cases according to use case document specifications.

2.4.3.2 Test Generation For All Scenarios (FR TARGET 0150)

The user shall be able to generate test cases for all scenarios specified in the imported use cases. In this case, the test generation algorithm will consider all existent flows described by the use cases and no filter is applied.

2.4.3.3 Test Selection With Test Purposes (FR TARGET 0151)

The user shall be able to select test cases using test purposes. The tool shall allow the user to define the test purposes.

A test purpose is an abstract description of a subset of the specification, allowing choosing behaviors to test and, consequently, allowing reducing the specification exploration. A test purpose is composed of a sequence of steps. The operator "*" also can be used and means any sequence of steps. For example, if the test purpose is defined as "*;4M", it means that the tool shall select all test cases that finish on 4M step.

2.4.3.4 Test Selection Based on Similarity (FR TARGET 0152)

In the Path Coverage filter, the user shall be able to select test cases based on similarity. The tool shall discard the most similar test cases based on the percentage informed by the user.

The similarity between two test cases is defined as:

N/(S1+S2)/2)

where N is the number of steps of test case TC1 that are contained in test case TC2, S1 is the number of steps of TC1, and S2 is the number of steps of TC2.

2.4.3.5 Test Selection Based on Use Cases (FR TARGET 0153)

In the Use Case filter, the user shall be able to add and/or remove one or more test cases based on the use cases. The tool shall select all test cases that contain at least one step related to selected use cases.

2.4.4 Test Cases Viewer

This subsection describes the requirements related to test cases visualization, filters selection, inclusion and exclusion of the test cases and test suite generation.

2.4.4.1 Test case visualization options (FR TARGET 0235)

The user shall be able to visualize all selected (FR_TARGET_0234) test cases according to the filters. It is possible to visualize these test cases by group separately: Common Test Cases, New Test Cases and Removed Test Cases.

The Common Test Cases shall represent all the test cases that are in common between actual filter (FR_TARGET_0236) and the last filter (FR_TARGET_0234) selected, and shall appear in black color.

The New Test Cases shall represent all the new test cases comparing the actual filter (FR_TARGET_0236) and the last filter (FR_TARGET_0234) selected, and it shall appear in green color.

The Removed Test Cases are the test cases that were removed comparing the actual filter (FR_TARGET_0236) and the last filter (FR_TARGET_0234) selected, and they shall appear in red color.

2.4.4.2 Filter selection (FR TARGET 0236)

The user shall be able to save the current filter selection to use afterwards. The saved filter shall appear in the list of filters. In order to use it, the user must select it from the list and load it. This list shall have the “All Test Cases” as default value and the other saved filters shall have the “Selection X” name, where X represents the order in which it was generated. After loading a saved filter, the respective test cases shall appear (FR_TARGET_0235) to the user.

2.4.4.3 Test case visualization (FR TARGET 0237)

The tool shall provide the visualization of all selected test cases. The user shall be able to select one of them and it shall appear in a table with all test case contents according to Test Suite Format (FR_TARGET_0175)

2.4.4.4 Test case inclusion/exclusion (FR TARGET 0238)

The user shall be able to include or exclude permanently a test case, by clicking the right button of the mouse. This option means that a test case will be always included or excluded of test suite independently of any filter selected.

2.4.4.5 Test suite generation (FR TARGET 0239)

The user shall be able to choose a filter or none filter in the On The Fly Generation editor. If no filters are selected, then all possible test cases shall be generated. After select the wanted test cases, the user shall be able to generate the Test Suite.

If there is at least one error in the imported use cases documents, the tool shall not generate the test suit.

2.4.5 Traceability Matrix (FR TARGET 0240)

The user shall be able to visualize all selected test cases, according to the Requirements x Test Cases relation and the Use Case x Test Cases relation.

The FR_TARGET_0235, FR_TARGET_0236, FR_TARGET_0237 and FR_TARGET_0238 are also related to this requirement. When a test case is selected, it shall be showed in the spreadsheet format according to Test Suite Format (FR_TARGET_0175). The user shall be able to visualize only one test case per time.

2.4.6 Included or Excluded Test Cases (FR TARGET 0241)

The user shall be able to visualize all test cases included and/or excluded permanently from the test suite. For example, besides the filters mentioned in FR_TARGET_0234, the user should be able to include or exclude some test cases permanently, independently from filters selection.

2.4.7 Generation Summary (FR TARGET 0160)

The tool shall display a summary of the generation right after it has finished. This summary shall include the follow information:

- Total time required to complete the process;
- The total number of generated test cases.

2.4.8 Test Suite Name (FR TARGET 0170)

The name of the test suite file shall be automatically generated by the tool.

If the IMPORT TEMPLATE TEST CENTRAL 4 Feature (See Appendix B) is present in the TaRGeT product, the user shall be able to import a template (FR_TARGET_0175) and the test suite file name shall have the same name from template.

If the IMPORT TEMPLATE TEST CENTRAL 4 Feature (See Appendix B) is not present in the TaRGeT product, the user shall not be able to import a template and the test suite file name shall has the format "NewTestSuite000X", where X is the number of test suits already generated in crescent order. This name shall be different of any test suite file that is already stored on the current project.

2.4.9 Test Suite Format (FR TARGET 0175)

The generated test suite can be formatted in xml, html or Excel formats depending on the enabled feature on the TaRGeT product (OUTPUT Feature – See Appendix B). Each generated test case shall have the following information:

- Case – A label that uniquely identifies the test case inside the test suite. The user shall be able to customize the test case identification field in the Preferences window (FR_TARGET 0231);
- Execution type – The default content generated by TaRGeT;
- Case Description: The test case's description originated from use case description. The user shall be able to customize the use case description in the Preferences window (FR_TARGET 0232);
- Use cases – The use case's id related to the test case;
- Objective – The test case's objective originated from flow description of the use case. The user shall be able to customize the flow description in the Preferences window (FR_TARGET 0232);
- Requirements – the ids of the requirements that the test case covers;
- Setup - a draft of the test case's setup;
- Initial conditions – a draft of test case's initial conditions;
- Test Procedure (Step Number)- The test case steps number;
- Procedures – the actions of the test case;
- Expected Results: The responses of the test case;
- Notes - default content generated by TaRGeT.

The other fields of the test case, such as regression level, final conditions and cleanup are left empty. The user shall be able to customize the empty field in the Preferences window (FR_TARGET 0231);

Moreover, the test suite shall contain information about the Traceability Matrix (FR_TARGET_0180).

2.4.10 **Requirements Traceability Matrix (RTM) Generation (FR TARGET 0180)**

The tool shall automatically generate the RTM for each generated test suite. The RTM shall contain the following relations: Requirements X Use Cases, Requirements x Test Cases, and Use Cases x Test Cases.

2.4.11 **Revision History Generation (FR TARGET 243)**

In spreadsheets formats, it shall be able to generate a table related to revision history providing the user to control the test suits versions. The fields available are:

- Date Last Updated
- Modified By
- Comments

These fields are empty and the user may add the information manually.

2.5 **Consistency Management**

This subsection describes the requirements related to the management of test cases consistency (CONSISTENCY MANAGEMENT Feature – See Appendix B)

2.5.1 **Consistency Management (FR TARGET 244)**

Consistency management is a feature to control the consistency of already generated test cases if the user makes any modifications in the use case document. The feature shall demonstrate the percentage of similarity between a new test case and an old one and the fields that are different between them.

The user shall be able to associate a new test case with and old one based on the level of similarity between them. The tool shall perform a merge keeping the following fields from the new test case:

- Steps
- Requirements

- Initial Conditions
- Use Case References
- Setup
- Feature Id

and the following fields from the old test case:

- Id
- Status
- Description
- Objective
- Note
- Execution Type
- Regression Level
- Final Conditions
- Cleanup

The user shall be able to attribute a new id to one test case. In this case, all fields of the new test case shall be kept and only the id shall be modified.

Consistency manager will only work if there is at least one test suite created.

2.5.2 Selecting Test Suite (FR TARGET 245)

The user shall be able to select an already created test suit from the project to compare it with the new test cases generated by the On The Fly functionality.

2.5.3 Associating Test Cases (FR TARGET 246)

The user shall be able to associate new test cases generated by the On The Fly functionality with old test cases from the selected test suite based on the level of similarity between them.

The tool shall show the percentage of similarity of a new test case with the old ones. Test cases that have a similarity level of 100% shall be automatic associated.

The tool shall indicate when a new test case does not have a similarity level of 100% with any one other (it may probably be a new test case created by the user or a modified one).

The user shall be able to attribute a new id to a new test case or remove it from the test suite that will be generated.

2.5.4 Visualizing Test Case differences (FR TARGET 247)

The user shall be able to visualize the differences between a new test case generated by the On The Fly functionality and an old test case from the selected test suite.

The full description of the selected test case shall be displayed to the user on a table format. The tool shall indicate which fields of the test cases are different and if one test has extra steps compared to the other.

2.5.5 Setting up Next New ID (FR TARGET 248)

The user shall be able to define the new id that will be attributed to test cases marked as new ones. If there is more than one test case marked as new, the tool will increment the id starting from the value passed by the user.

2.5.6 Setting Up Automatic Association (FR TARGET 249)

The user shall be able to set up the values of test case similarity for the tool to perform an automatic association of new and old test cases according to these values. These values are in a range from 0 to 100%. The first value is related to the level of similarity of the most similar old test case and the second value is related to the level of similarity of the second most similar old test case.

For example, the user shall be able to define levels of similarity according to the behavior: "*if the first most similar test case to the new one has a similarity level greater than x% and the second most similar test case to the new one has a similarity level less than y%, the tool will associate automatically the new test case to the most similar old one*." the x and y values shall be configured by the user.

2.5.7 Changing test suite to compare (FR TARGET 253)

The user shall be able to change the test suite file that is being compared to test suite generated by On The Fly generator. In this case, another Consistency Management window is opened with this new suite comparison information.

2.5.8 Generating Test Suite (FR TARGET 250)

After performing the Consistency Management, the user shall be able to generate a new test suite. This test suite must contain all the modifications that the user has made during the consistency management activity.

This test suit will be the result of a merge between the old test suite selected to be compared and the new test cases generated by the On The Fly functionality. The new generated suite shall also consider the new ids that the user has attributed to the test case

2.6 Controlled Natural Language (CNL)

This subsection describes the requirements related to the controlled natural language feature.

2.6.1 Controlled Natural Language (FR TARGET 255)

A Controlled Natural Language (CNL) is a subset of an existing natural language, like English, French, or Portuguese. Basically, a CNL defines some writing rules and a restricted vocabulary in order to avoid authors to introduce ambiguities into their texts. It is designed to address the communicative needs of a particular area, or a particular company.

A CNL may aim three major purposes (listed below). Actually, these purposes complement each other, in a way that to achieve the purpose below, the above one must be achieved.

- Text Standardization – the most basic purpose of a CNL is to define a standard to be followed throughout an organization. For that, a set of writing constructions (grammar) and a specific vocabulary is needed;

- Deterministic Mapping to an Intermediate Representation – since there is a grammar and a vocabulary, it is possible to define rules that map the CNL sentences into a structured representation, in order to prepare it for further processing.

- Semantic Analysis – the goal of semantic analysis is understand the relation between the user utterance and the concept he/she wants to convey.

Currently, the CNL only address the above item (1). However, it is intended to continue the CNL development in order to achieve the purposes (2) and (3).

2.6.2 Searching, adding, editing and removing terms in lexicon vocabulary (FR TARGET 256)

The user shall be able to add, edit or remove adjectives, adverbs, conjunctions, determiners, nouns, prepositions and verbs in the lexicon vocabulary. For each category, specific information should be filled in. The CNL should not allow the insertion of pronouns and modal verbs.

2.6.3 Integration with a dictionary (FR TARGET 257)

The CNL shall be integrated with an external dictionary that will validate the term the user tries to add in the lexicon. The dictionary should analyze if the term is correctly spelled and if the category selected by the user is suitable. The dictionary will be also responsible to suggest synonyms for unknown term.

2.6.4 Visualizing syntax errors details (FR TARGET 258)

The user shall be able to visualize the syntax error details and to see the possible corrections for the mistakes found in the sentences.

2.6.5 Sorting CNL errors (FR TARGET 259)

The user shall be able to sort the CNL errors displayed in the user interface by the error description, the sentence where the error was found, the step id where the error was found or the use case document that contains the error.

2.6.6 Filtering CNL errors (FR TARGET 260)

The user shall be able to filter the CNL errors using as criteria the type of the sentences where they were found (user action, system condition or system response).

2.6.7 Reloading CNL (FR TARGET 261)

The user shall be able to reload the CNL configuration files and reprocess the imported use case documents. The CNL error table shall be updated on the moment of the refresh.

2.7 Use Case Editor

This subsection describes the requirements related to the Use Case Editor feature.

2.7.1 Opening an empty use case editor (FR TARGET 262)

The user shall be able to open an empty use case editor to create a new use case document or adding an use case to an existent one.

2.7.2 Opening and editing an existing use case using editor (FR TARGET 263)

The user shall be able to select an existing use case, open and edit its information about feature, use case, requirements, main flows and alternative flows.

2.7.3 Saving an use case (FR TARGET 270)

The user shall be able to create a new use case document or updating an old one. Interface components that contain reference errors should be highlighted after saving action. It is not allowed to create or update an use case document with empty mandatory fields.

2.7.4 Adding alternative flows (FR TARGET 264)

The user shall be able to add alternative flows into a new or existing use case.

2.7.5 Removing alternative flows (FR TARGET 265)

The user shall be able to remove alternative flows from a new or existing use case. The associations of the removed flow should be checked before removing it and the user should be asked for confirmation.

2.7.6 Adding new steps (FR TARGET 266)

The user shall be able to add steps into an existing flow.

2.7.7 Removing steps (FR TARGET 267)

The user shall be able to remove steps from an existing flow.

2.7.8 Searching terms (FR TARGET 268)

The user shall be able to search terms in use case editor. The interface components where the term (or another term that contain it)is found should be highlighted.

2.7.9 Exporting use cases (FR TARGET 269)

The user shall be able to export use case document with no empty mandatory fields in PDF format.

2.8 Others

This section includes the requirements related to other features.

2.8.1 Internationalization (FR TARGET 252)

It can be available in two different languages: English or Portuguese (IDIOM Feature – See Appendix B)

2.8.2 Help (FR TARGET 0205)

It shall be possible to user to access the user manual from its UI.

The option "Automatically detect settings" in the Internet Options shall be disabled (Browser menu -> Tools -> Internet Options -> Connections -> Local Area Network (LAN) Settings) to access the user manual from TaRGeT UI.

2.8.3 About (FR TARGET 0210)

It shall be possible to user to view the details about the version and the developer. It also displays the TaRGeT and CIn/BTC logos.

2.8.4 Splash Screen (FR TARGET 0220)

While the tool is being launched, it shall display a splash screen, showing the TaRGeT and CIn/BTC logos.

2.8.5 Background Image (FR TARGET 0225)

The tool shall display a background image, showing the TaRGeT and CIn/BTC logos, when there is no project opened.

2.8.6 Environment Error (FR TARGET 0230)

When, the tool is being initialized, it shall check and notify the existence of any problem regarding to the environment (see Section 3). The tool shall only open if no problem is found.

3. Environment Requirements

This section describes non functional requirements related to environment compatibility and external software dependences.

3.1 Compatibility

This subsection describes the requirements regarding the tool’s environment compatibility.

3.1.1 Desktop Platform (ER TARGET 0010)

The “TaRGeT” application shall work on Windows 2000, Windows XP, Windows Vista and MacOS, depending on the TaRGeT product version.

3.2 Software Dependencies

This subsection describes the requirements regarding the tool’s external software dependencies.

3.2.1 For Microsoft Office Input (Just available for versions for Windows)

3.2.1.1 .NET Runtime Environment (ER TARGET 0025)

TaRGeT application only works properly in machines with .NET Framework 2.0 Runtime Components installed. If the machine does not have it installed, a warning message is presented.

3.2.1.2 Microsoft Office (ER TARGET 0015)

TaRGeT application works properly with Microsoft Office 2003/2007 Professional installed. Only documents created using Microsoft Office 2003 can be imported in the TaRGeT projects. However they can be edited in both Office versions.

3.2.2 JVM (ER TARGET 0020)

TaRGeT application only works properly in machines with JVM compatible with JRE Version 6.0 installed [1]. If the machine does not have the JVM installed, a warning message is presented.

4. Tool Workspace

This section aims to describe the TaRGeT workspace. The workspace is a standard visual structure of the tool composed by menus and panels.

4.1 Project Window

Figure 1 shows the TaRGeT work area when there is an opened project. It is composed by a menu bar and three panels: Left Panel, Main Panel and Bottom Panel.

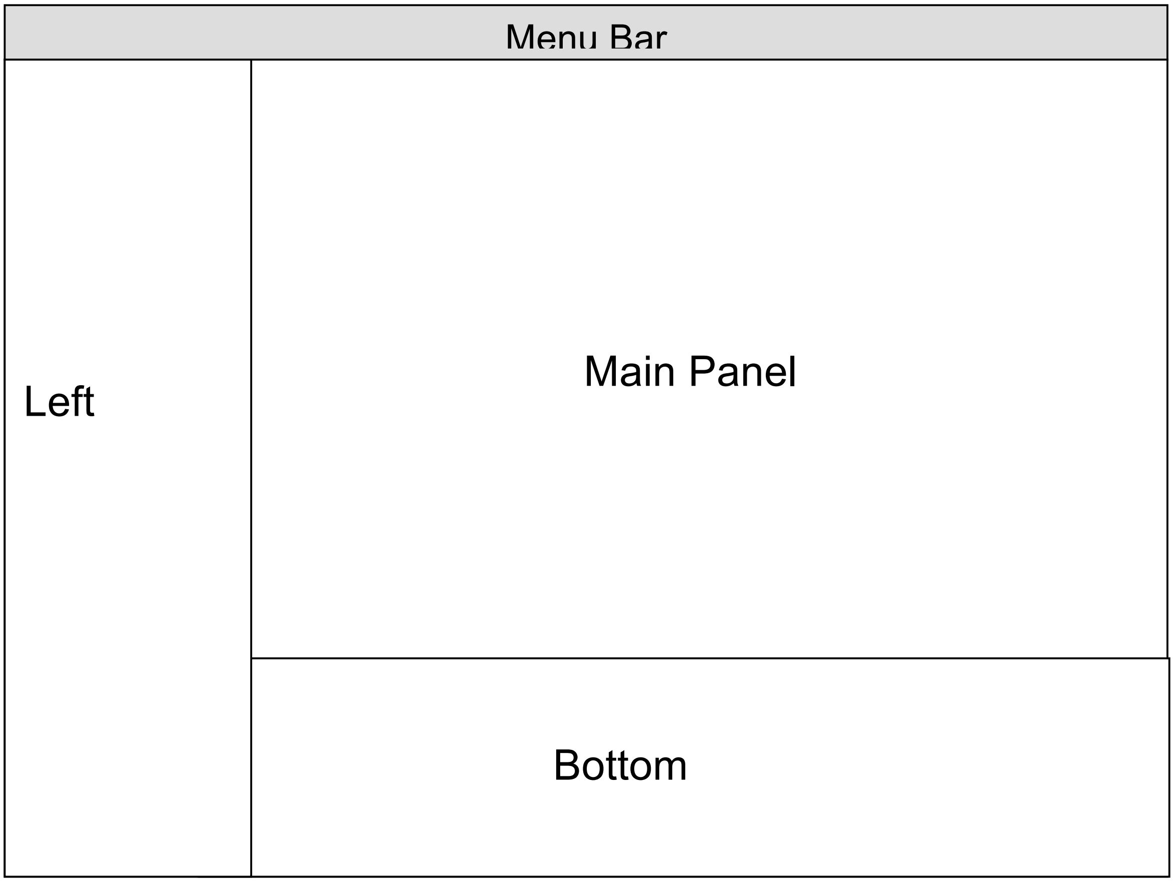


Figure 1 – Structure of TaRGeT's Interface.

4.2 Menu Bar

The tool GUI shall have the menu tree described below (consider the order):

- File

    - New Project
    - Open Project
    - Close Project
    - Exit

- Artifacts
    - Import Use Case Documents

- Project
    - Refresh Automatically
    - Refresh Project

- Tools
    - Search
    - Preferences
    - On the Fly Generation

- Help
    - Help Contents
    - About TaRGeT

4.3 Left Panel

This panel contains the Use Cases view, in which the use cases are outlined. It is shown as tree structure, grouping the use cases according to its feature.

4.4 Bottom Panel

This panel allows three views:

- Artifacts View – it outlines the project artifacts. It groups, in two different folders, the use case documents and the test suite documents.

- Error View – it lists all current errors that appear in the use case documents.

- Search Results View – contains the results of any search on use cases

4.5 Main Panel

This panel is used to display the use case document's content and On the Fly Generation editor. It shall allow the use case preview and test cases preview that should be generated according to the user choose.

5. Appendix A – Use Case Document Format

Use case documents shall be Word or XML documents according to the structure and rules described below.

Each use case document shall have some fields associated with a XML structure that stores the relevant information to be processed by the tool. Only the relevant fields (associated with XML) are used by the tool, any other information inside the document is ignored. The relevant fields are:

- Feature
  - Use Case
    - Flow
      - Step

Each document shall have at least one feature. Each feature shall have at least one use case. Each use case shall have at least one flow. Each flow shall have at least one step.

The upper structure of the document is made of features. A feature has an id, a name and a set of use cases; and, a use case has an id, name, description, setup information and flows. The details about the upper structure can be seen as follows.

The tool allows the user to import more than one document related to the same feature. In this case, the features shall have the same feature id. So, the tool merges the information and shows in the "Use Cases" view.

- **Feature**
  - *Feature id* – a free textual value and non empty text. It shall be unique inside a document.

    The tool should show an error message and reject the Use case document in the following cases:

    - *Feature Id* is empty;
    - The *Feature Id* is not unique in the same Use Case document;
  - *Feature name* – empty field or any other value;

- **Use Case**
  - *Use Case Id* – a free textual value and non empty text. It shall be unique inside a specific feature, i.e., it is possible to exist two use cases with same id in different features. The tool

should show an error message and reject the Use case document in the following cases:

  - The *Use Case Id* is empty;
  - The *Use Case Id* is not unique in the same feature

  In the other cases, the tool shall accept the Use case document and does not show the error message.

- o *Use Case Name* – empty field or any other value;
- o *Use Case Description* – empty field or any other value;
- o *Use Case Setup* – empty field or any other value.

The *Use Case Descriptions*, and use case setup are optional. All other fields are mandatory;

- **Flows**

The lower structure of the document is composed by flows. A flow has a description, a list of source flows (*FROM Steps*), a list of destination flows (*TO Steps*) and a sequence of steps. Each step has an id, an action, a condition, setup information (optional) and a response. The details about the lower structure can be seen as follow.

- o *Flow Description – empty field or any other value;*
- o *FROM Steps – Non empty textual value.*
  In the following situations the tool shows a warning before load the document, accept the Use Case document and show an error in a specific area into of the tool:

  - *FROM Steps* with repeated step ids the same field;
  - *FROM Steps* with step ids which are not separated by commas;
  - *FROM Steps* with step ids which contains symbols, numbers or letters;
  - *FROM Steps* with step ids which do not exist.

  In the case where the *FROM Steps* is empty, the tool rejects the use case document.

- o *TO Steps – same rule of FROM Steps;*

- **Steps**

A sequence of steps. Each step is composed by:

- o *Step Id* – a free textual value and non empty text. The tool shall show a warning before loading the document, accept the Use Case document and show an error in the following cases:

    - The textual value is empty
    - Duplicated Step Id in the same use case or same flow.

  In the cases that the rule above is not applicable, i.e., letter followed by number or number followed by minuscule letters or only numbers or only letters or symbols, the tool will show an error message and reject the Use Case document.

- *Step Action* – The tool will accept any value that does not empty. If field value is empty the tool will show a warning before load the document, accept the Use Case document and indicate an error in the error tab;

- *Step Condition* - The tool accept an empty field or any other value;

- *Step Response* - The tool will accept any value that does not empty. If field value is empty the tool will show a warning before load the document, accept the Use Case document and indicate an error in the error tab.

All the *Step Ids* shall be unique in the use case. The above fields, except the *Step Condition* and *Setup* are mandatory.

The *FROM Steps* and *TO Steps* is a list of references to steps from other flows. The former indicates that the flow behavior is a possible continuation of the referenced *Steps Ids*. The last indicates that the flow continues to the steps which ids are referenced. If the *FROM* and *TO* fields has more than one listed Step Id, they shall be separated by ',' (comma) character.

To indicate that the scenario may start at the flow, the user shall insert the 'START' id in the *FROM Steps* field. On the other hand, to indicate that the scenario may end at the flow, the user shall insert the 'END' id in the *TO Steps* field.

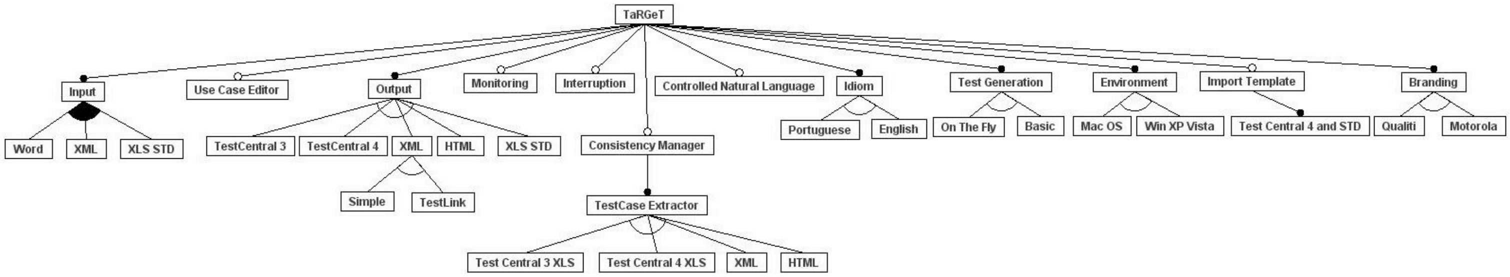
When it is desired to refer to a step whose flow is in the same use case, the user shall only provide the id of the referenced step.

When it is desired to refer to a step whose flow is in the same feature but in a different use case, the user shall provide 'UC_ID#STEP_ID' reference. Where UC_ID is the Use Case Id of the referenced step and STEP_ID is the Step Id.

When it is desired to refer to a step whose flow is in a different feature, the user shall provide 'FEATURE_ID#UC_ID#STEP_ID' reference. It means that FEATURE_ID is the Feature Id of the referenced step, UC_ID is the Use Case Id and STEP_ID is the Step Id.

The *Requirement Id* is an optional field that is written at the end of step action, condition and/or response fields. The id shall appear between '[' (open brace) and ']' (close brace) characters. If there is more than one requirement in the same step field, they shall be separated by ‘,’ (comma) character (e.g., [REQID1, REQID2]).

6. Appendix B – Feature Model


TaRGeT
Input
Use Case Editor
Output
Monitoring
Interruption
Controlled Natural Language
Idiom
Test Generation
Environment
Import Template
Branding
Word
XML
XLS STD
TestCentral 3
TestCentral 4
XML
HTML
XLS STD
Consistency Manager
Portuguese
English
On The Fly
Basic
Mac OS
Win XP Vista
Test Central 4 and STD
Qualiti
Motorola
Simple
TestLink
TestCase Extractor
Test Central 3 XLS
Test Central 4 XLS
XML
HTML