hTK-Legend Test Framework Functional component

Whitepaper

Table of Contents

Executive Summary

Increasing amount of mobile and network communication brings stability, reliability and efficiency of associated services on top of public and company interests. Events like New Year's Eve and voting on popular TV pop star singing contests cause steep traffic peaks and require powerful infrastructure and related systems.

Continuous innovation of technologies brings urgent need of introducing new functionalities faster and more reliable then ever before. These circumstances force service providers to optimize their network structure and resources, test and prepare systems for high load as well as for new features.

To be able to collect information about current system status and make strategic decisions to improve service quality and efficiency, necessity to simulate different traffic scenarios becomes much more important and tests of new features and system performance are crucial part of service deployment and maintenance. These tests require simulation of traffic scenarios close to real conditions with the ability to clearly identify and examine test results.

henkel-TK GmbH provides a traffic generator with unique possibilities to test performance, functionality, features and reliability of service provider systems with many different protocols and applications simulating real traffic conditions. The services provided by henkel-TK GmbH range from consultancy and support to training and fully customized on-site testing with presence of henkel-TK GmbH staff.

Introduction

The hTK-Legend Test Framework with its Functional component is a traffic generator system, which allows enhanced creation, management and execution of detailed message call flows involving one or more protocols at the same time. Support of various protocols allow service provides and software vendors to automatically test and examine system features and behavior during standard or specific simulated situations.

The application spectrum covers pre-release product tests at software vendors, acceptance tests during implementation of software updates and newly introduced features at service providers, service correctness tests during maintenance windows and regularly scheduled tests of network elements for their quality and reliability.

The Functional component is based on the concept of the widely used and proven hTK-Legend Test Framework Load & Stress component, which makes its network integration fast and easy. User access, test management and test execution are provided by the well known graphical user interface XLegend. Results of each test execution are stored in databases and are accessible via a Result-Portal with web-based user interface. Directly from there comprehensive reports can be sent to configurable lists of e-mail recipients.

Key features of the Functional component:

- architecture designed to easy add almost any protocol or stack
- in-house created protocol definitions allow using Functional component for various types of testing
- detailed configuration of parameters within protocol messages
- flexible and maintainable data administration via testdata sets
- sophisticated call flow composition with messages sent to and received from System Under Test
- scenarios composed of reusable testcases representing desired callflows
- testsuites contain scenarios and other testsuites
- testsuites handle conditional execution and provide execution watchdog
- scheduled execution of scenarios and testsuites with flexible execution interval options
- detailed results stored in SQLite database format
- results details accessible via Result-Portal with web-based user interface
- reports of results provided by Result-Portal via email
- support of different protocols running at the same time within one scenario
- powerful and extendable parameter value evaluation and result clarification
- expression evaluation language to evaluate received or manipulate sent parameter values
- stable interface for easy and reliable integration into existing environments

Connection example

Figure 1. SS7/SIGTRAN network simulation for SMSC with hTK-Legend Test Framework

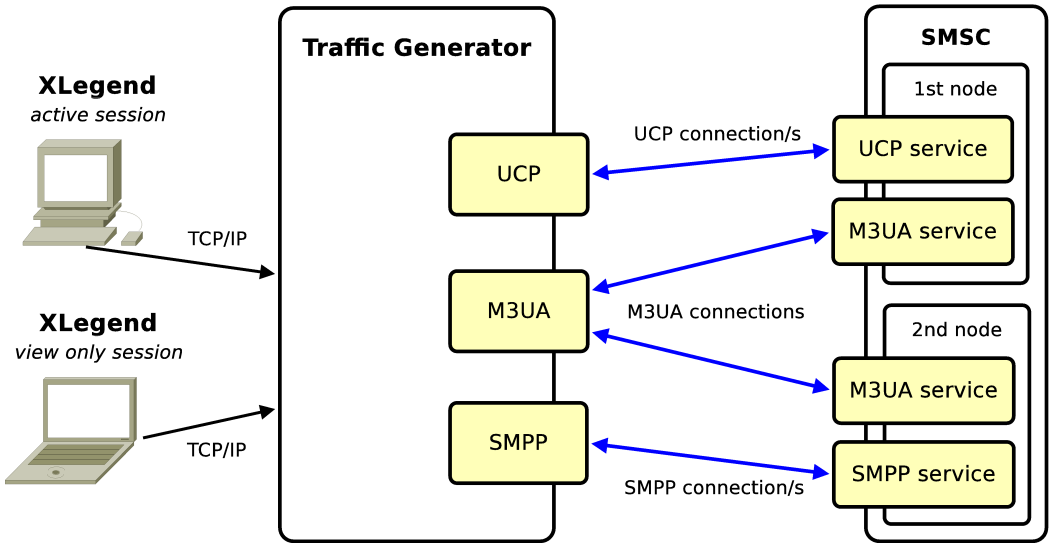


Figure 1, “SS7/SIGTRAN network simulation for SMSC with hTK-Legend Test Framework” shows connection between Traffic Generator and 2-node SMSC using 3 different protocols (M3UA, UCP, SMPP). hTK-Legend Traffic Generator generates and accepts SMS messages to and from SMSC on all protocols and simulates HLR function on M3UA connections.

Field application examples

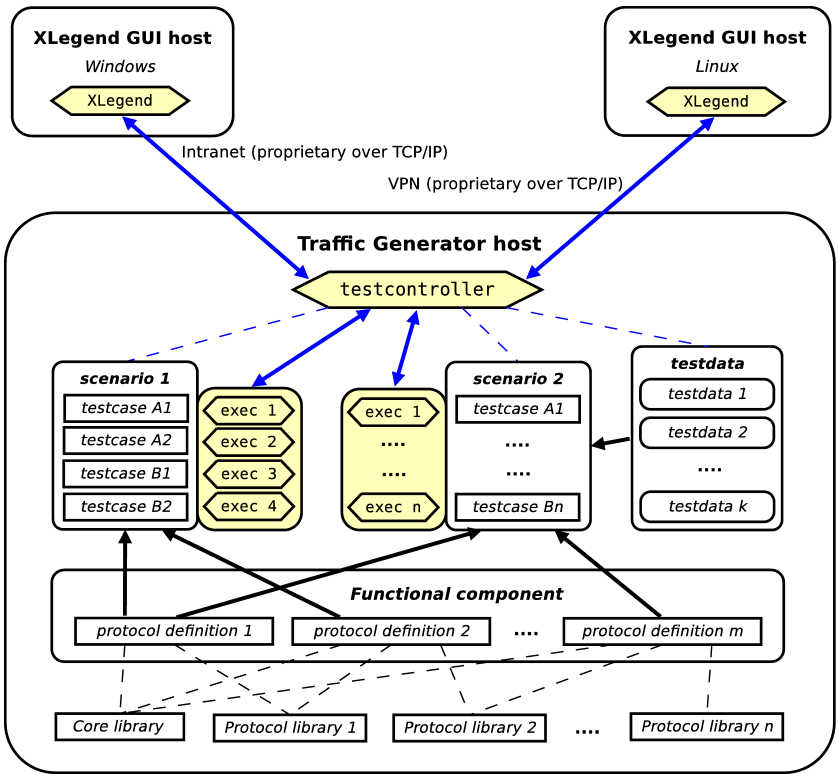
Thanks to broad features support, the Functional component in combination with Result-Portal can be used by organizations, companies and departments to improve service and software quality and increase effectivity of internal testing and monitoring procedures. The list below provides some examples:

- at *software vendor company* engineers of development and support departments can use full configurability of the message parameters to test newly implemented features and bugfix or simulate customer reported situations on test system
- at *software vendor company* engineers of test department can use regularly scheduled testsuites, stable interface for integration into current test environment and results reporting via e-mail to automatically and continuously test new software releases; this is known as regression testing, and tests are enhanced and extended based on new failure reports or requirements
- at *service provider company* engineers at test, monitoring and service integration departments can use configurability, automation features, testsuites with conditional execution and reporting to test new software releases, monitor and report status of various systems in both test and live environments together with automated issue identification and various service disruption diagnosis

Architecture

An abstract overview of the Functional component architecture is shown below in Figure 2, “hTK-Legend Test Framework architecture”.

Figure 2. hTK-Legend Test Framework architecture



The hTK-Legend Test Framework is built using a modular design architecture with lots of possibilities for its usage and extension. Functional component main development goal is to provide rich configuration possibilities for supported protocols. The control GUI software module called XLegend and the Traffic Generator processes are separated and shall be hosted by different machines. Both modules communicate via a proprietary protocol on top of TCP/IP. This communication is fully encrypted and connection is initiated from XLegend towards Traffic Generator host.

The Traffic Generator consists of a control engine called *testcontroller*, a core library and multiple protocol libraries. The Functional component consists of protocol definitions developed for functional tests with simulated traffic call flows. The definitions reference API functions offered by core and protocol libraries. Every protocol definition provides a set of messages with parameters configurable during creation and modification of testcases.

A *testcase* is associated with exactly one protocol definition of Functional component but every protocol definition can be referenced by several testcases. Testcases are composed to *scenarios* representing complex call flows. Configuration data is maintained in *testdata* sets that are applied to scenarios during test execution. *Testsuites* group scenarios and possibly other testsuites for sequential and possibly conditional execution.

Results produced by executables are communicated to connected XLegends via the testcontroller process and are also written to a file system mounted on the Traffic Generator host.

Control and management of the Traffic Generator itself and all its constituents are performed using the XLegend - the graphical user interface part of hTK-Legend Test Framework.

The hTK-Legend Test Framework is protected by Wibu CodeMeter software protection solution.

Testcontroller

The testcontroller is the central process running on the Traffic Generator host, see Figure 2, “hTK-Legend Test Framework architecture”. Its most important tasks and features are listed below:

- controls and monitors test execution
- executes scheduled scenarios and testsuites
- supports multi user environment
- manages connections and data transfer to and from one or more XLegends

Functional component

The Functional component consists of protocol definitions. These provide the foundation for testcases, which generate and receive traffic between Traffic Generator host and the System Under Test. Key features are listed below:

- developed and implemented using XML markup and C programming languages
- easily extendable with new messages, parameters and protocols
- fully flexible configuration of complex call flows involving different protocols
- detailed test results in SQLite database format
- detailed result configuration including parameter evaluation and manipulation

XLegend

The XLegend controls, manages and monitors the Traffic Generator. The following list provides few examples of its rich functionality:

- provides interface to execute and schedule tests
- supports editing of complex test configurations including drawing of call flows
- presents messages and parameters of implemented protocols
- allows users to access features according to their respective privileges

Result-Portal

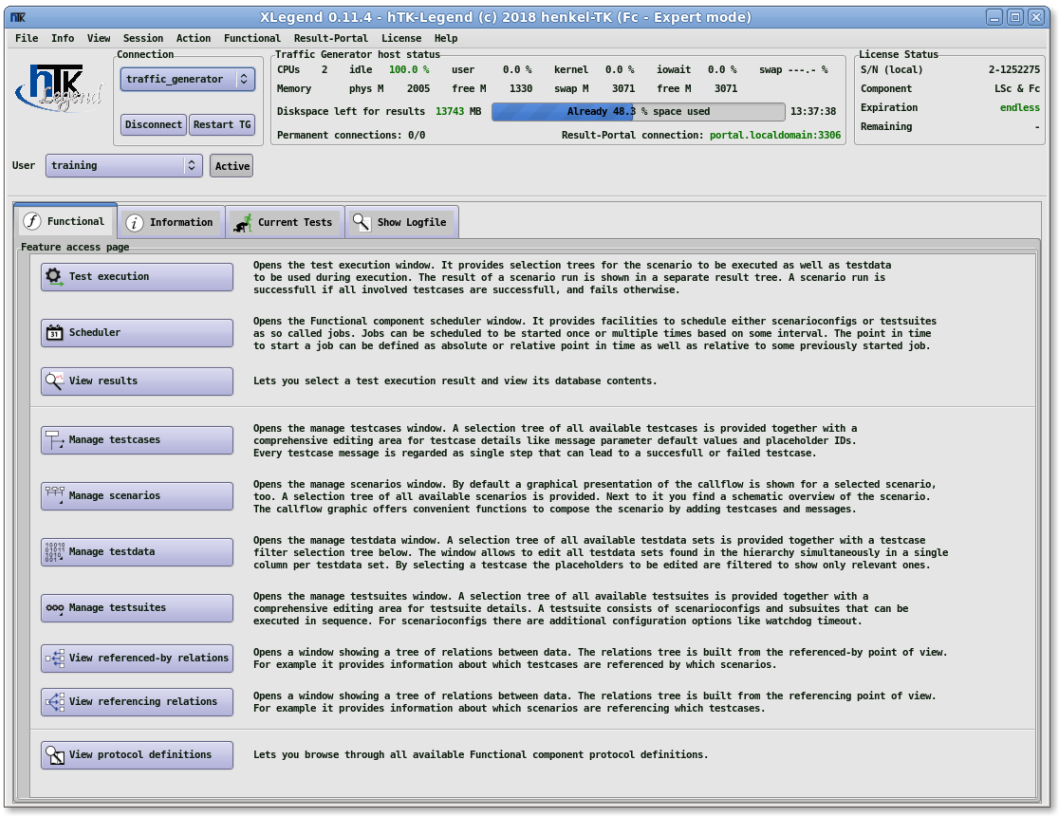
The Result-Portal is deployed together with the Functional component. A dedicated host system for the Result-Portal is highly recommended. Key features are listed below:

- web-based front-end with user access based on privileges
- uses filters and labels for easy navigation in results and archive
- provides enhanced view of all data involved with test execution, for example scenario callflows, testdata, traffic traces, etc.
- configurable result reports sent via e-mail
- protected by Wibu CodeMeter software protection solution

Test creation, management and execution

The main window of graphical user interface *XLegend* provides access to create, manage and execute tests. Its functionalities are available based on user priviliges configured on Traffic Generator. The menu at the very top contains items with all available functions. The top part contains areas with information about connected Traffic Generator and logged in user, information about Traffic Generator host status and license data. The main part contains *Functional* tab with buttons which open windows for tests management.

Figure 3. XLegend main window



Screenshots of main windows

Screenshots below display example of management windows needed to create, manage nad execute tests.

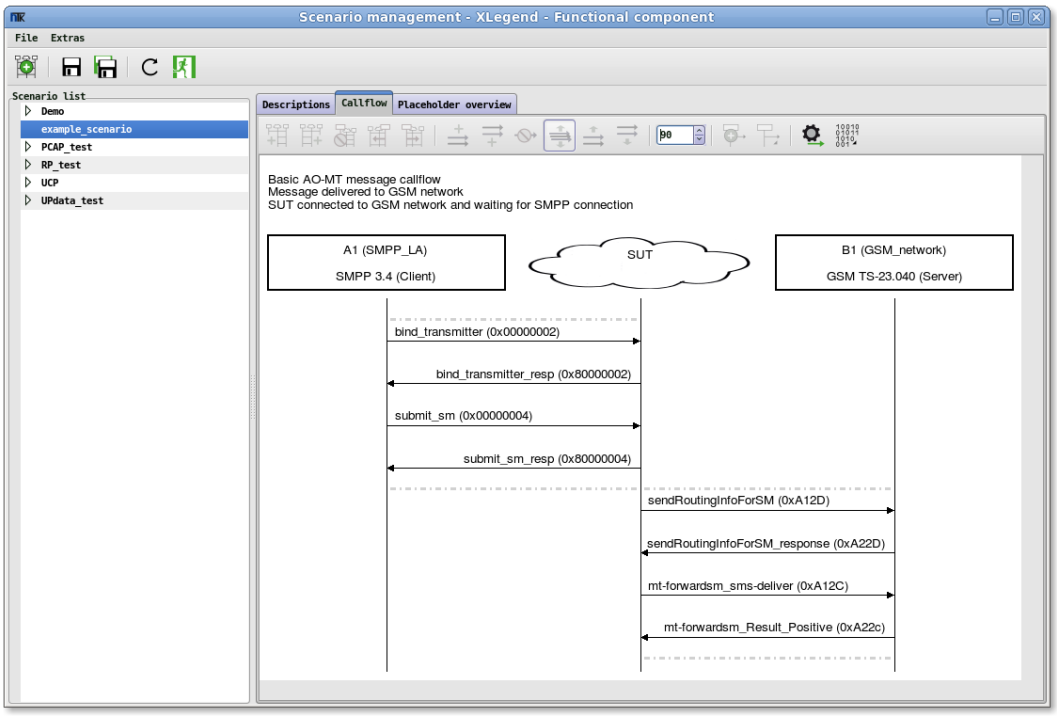
In the *Scenario management* window the scenario is created with testcases and callflow of messages between SUT and testcases. In *Testcase management* window parameters of messages are configured with either absolute values or placeholders.

In *Testdata management* window these placeholders are then configured with values for either scenario or testcase. The *Test execution* window allows management and execution of the default scenarioconfig or creation of new one with desired combination of scenario, testdata and labels.

The *Testsuite management* window provides possibility to serialize execution of scenarioconfigs and testsuites with option to define conditional execution of other scenarioconfigs and testsuites. The *Scheduler* window provides interface to manage internal Traffic Generator scheduler, schedule execution of scenarioconfigs and testsuites, define their repeat interval and delete already scheduled actions.

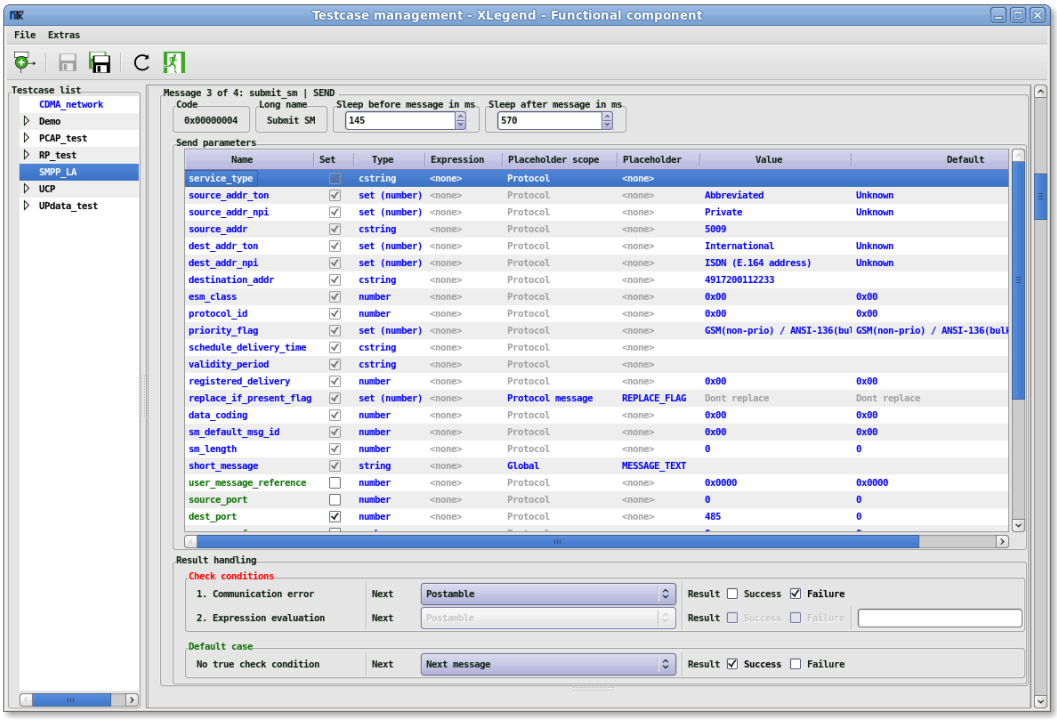
The figure below displays *Scenario management* window with an example of simple AO-MT SMS callflow.

Figure 4. Scenario management window



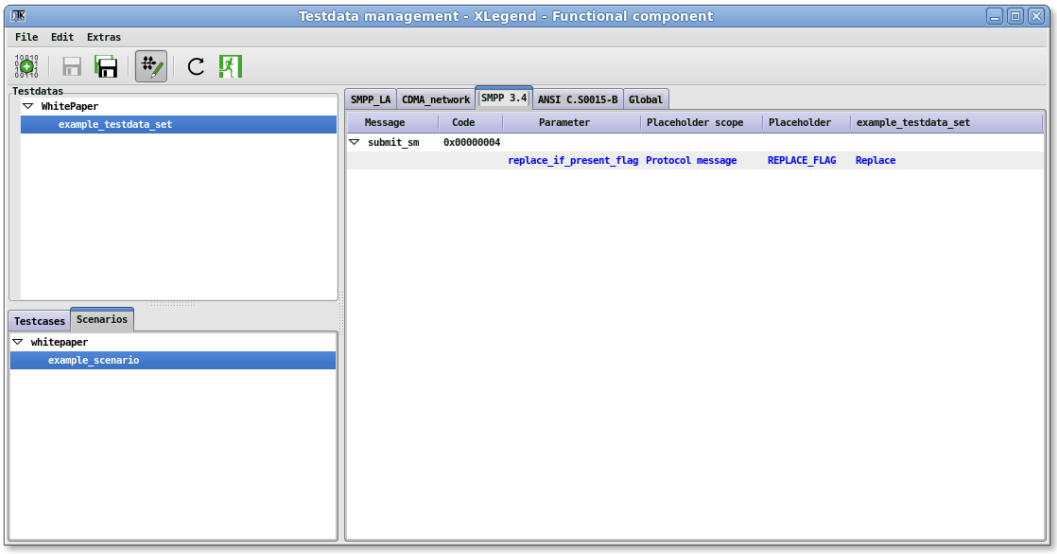
The figure below displays the *Testcase management* window with parameters configuration example of SMPP submit_sm message. The testcase will wait 145 ms before and 570 ms after sending this message. Fixed values and placeholders are used for the parameter values and message execution is considered as successful if message is sent out.

Figure 5. Testcase management window



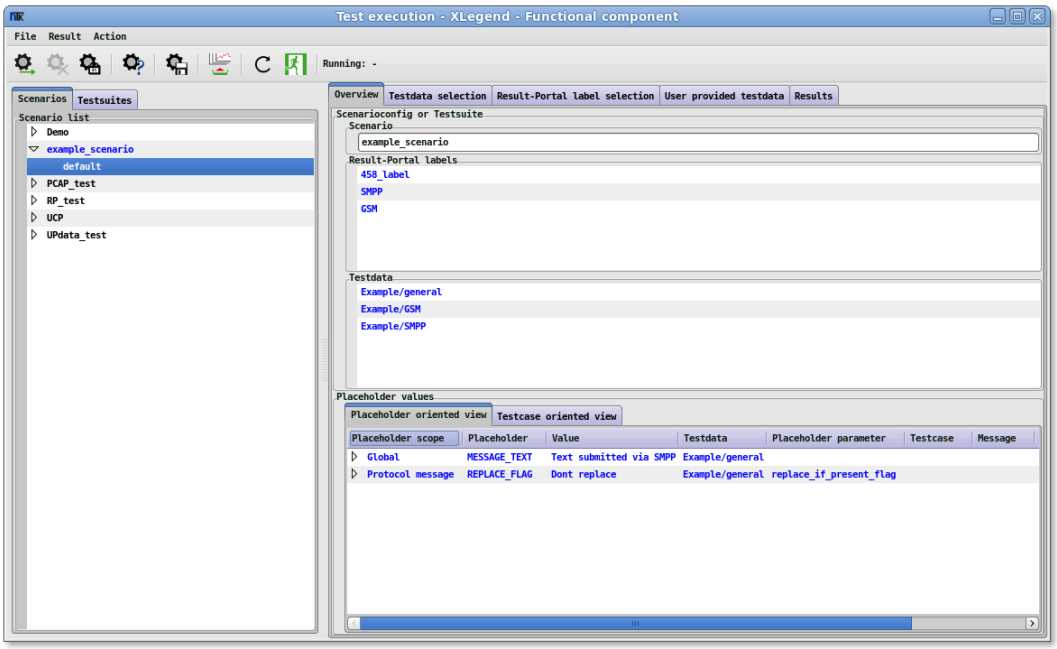
The figure below displays the *Testdata management* window with an example of testdata definition for particular scenario. The configured parameter value will be used during scenario execution.

Figure 6. Testdata management window



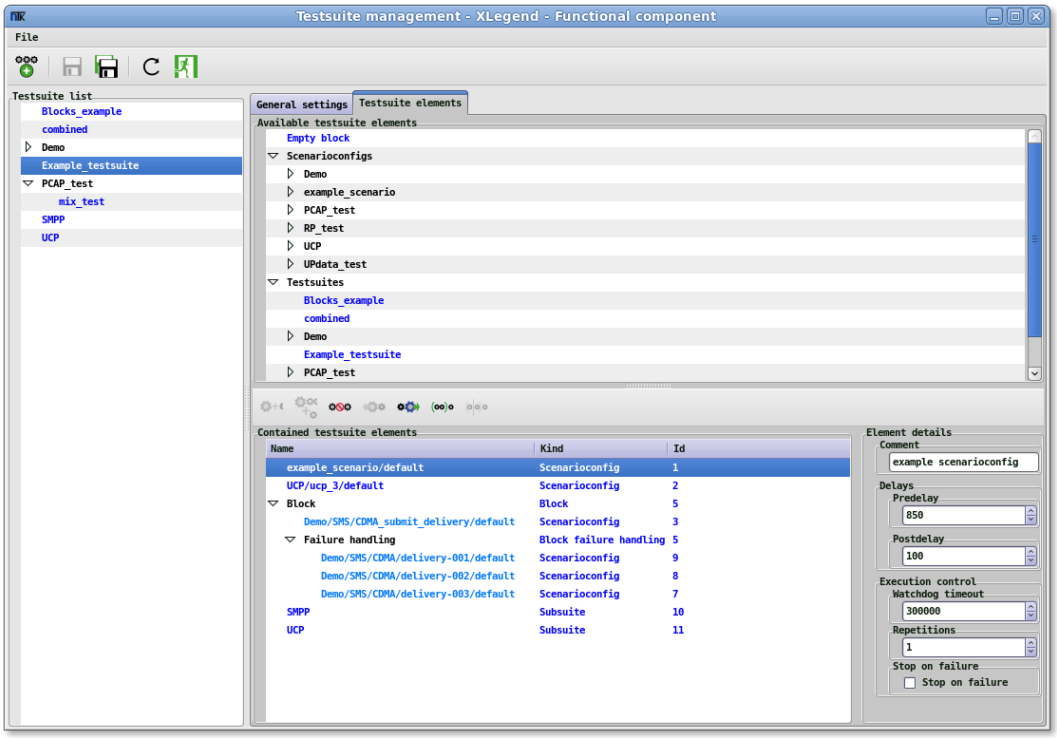
The figure below displays the *Test execution* window with overview of selected scenarioconfig. The different tabs provide possibility to select testdata sets, attach Result-Portal labels, provide testdata by user valid only for next execution and see results history of the selected scenarioconfig.

Figure 7. Test execution window



The figure below displays the *Testsuite management* window with an example of testsuite creation using scenarioconfigs, testsuites and a block defining failure handling.

Figure 8. Testsuite management window



The figure below displays the *Scheduler* window which allows to schedule selected scenarioconfig or testsuite with absolute or relative time definition, both time and entry relative, and repeat interval.

Figure 9. Scheduler window