# hTK-Legend Test Framework
# Functional component

## Whitepaper

Jindrich Rössler
Hendrik Ladner

# 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 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 range of services provided by henkel-TK GmbH covers 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. Currently supported protocols SMS over SIGTRAN, SMPP, UCP and MMS service related protocols allow mobile operators and software vendors to automatically test and examine system features and behavior during standard or very specific situation.

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 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. Test results of each call flow execution are stored in databases and are accessible via a web-based Result-Portal. Directly from there comprehensive reports can be sent to configurable lists of email recipients.

**Key features of the Functional component:**

- detailed message parameter configuration
- flexible and maintainable data administration via testdata sets
- sophisticated testcase composition with messages sent to or received from System Under Test
- scenarios composed of testcases representing call flows
- testsuites containing scenarios and other testsuites
- scheduled activites for scenarios and testsuites
- detailed results stored in SQLite database format
- results accessible via web-based Result-Portal
- reports of results provided via email
- support of different protocols running at the same time
- powerful and extendable parameter value evaluation
- stable API for easy and reliable integration into customer environment

# Supported protocols and protocol stacks

The hTK-Legend Test Framework supports several protocols and protocol stacks. Protocols and technologies currently implemented in Functional component are listed below, sorted alphabetically:

- MMS - Multimedia Messaging Service
- SMPP - Short Message Peer-to-Peer
- SMS - Short Message Service
- UCP - Universal Computer Protocol

# Connection example

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

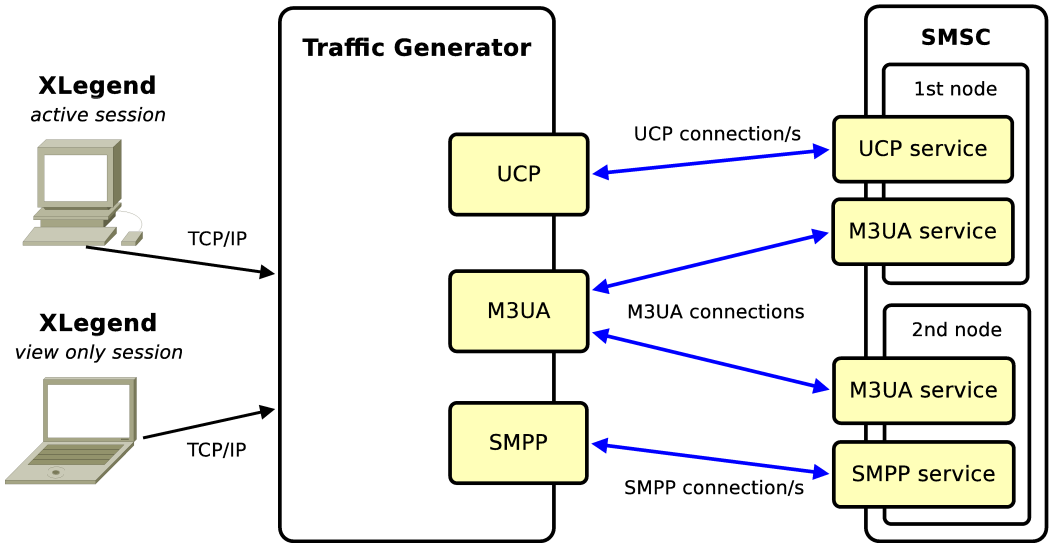
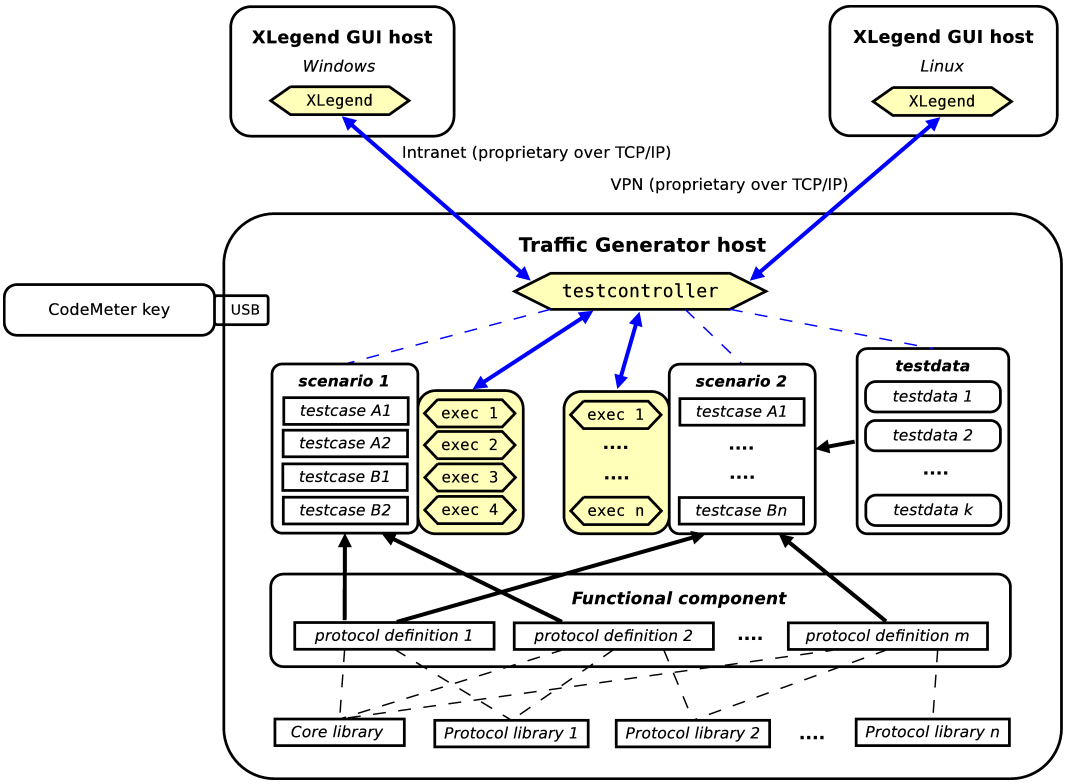



Figure 1, “SS7/SIGTRAN network simulation for SMSC with hTK-Legend Test Framework” shows connection between Traffic Generator and 2-node SMSC using 3 different protocols (M3UA, UCP, SMPP). hTK-Legend Traffic Generator generates and accepts SMS messages to and from SMSC on all protocols and simulates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

# Architecture

An abstract overview of the hTK-Legend Test Framework architecture with Functional component 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 a lot of different possibilities for usage and expansion. 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 completely separated and shall be hosted by different machines. Both components 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 plus 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 possible other testsuites for sequential execution.

Results produced by running execs 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.

To operate the hTK-Legend Test Framework it is required to have a CodeMeter USB stick with valid license connected to the Traffic Generator host.

# 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 base 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
- rich 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

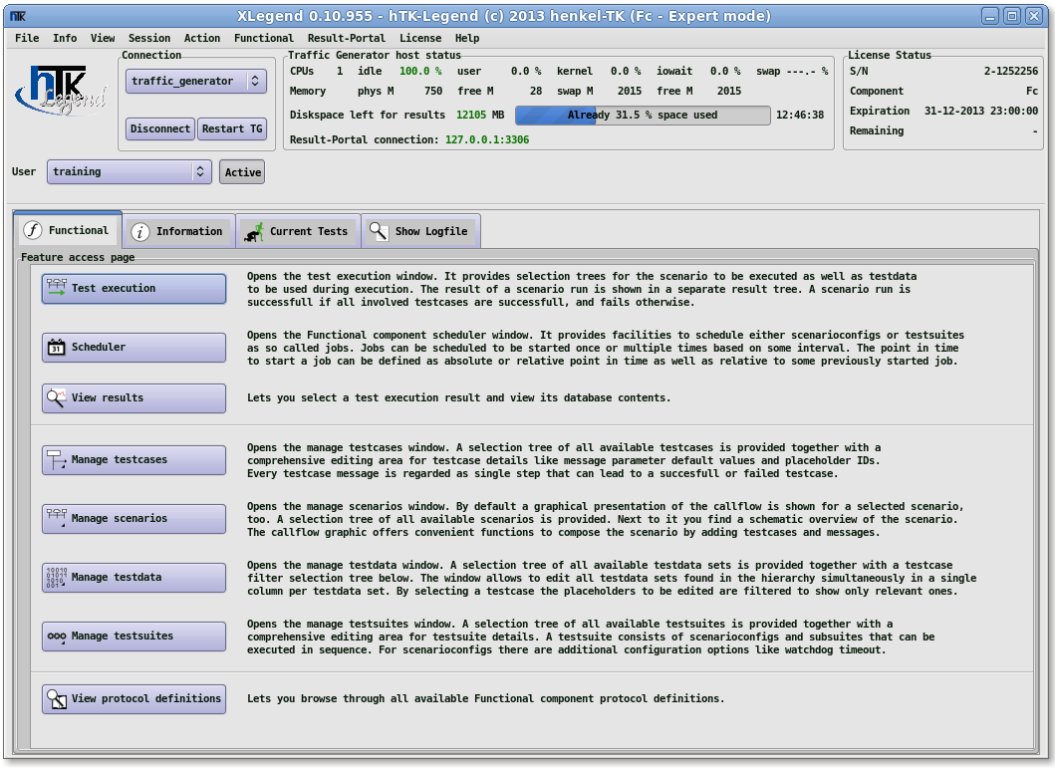
The Result-Portal is deployed together with the Functional component and is optionally available for the Load & Stress component. Dedicated hardware for the Result-Portal is recommended. Key features are listed below:

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

# Screenshots of XLegend windows

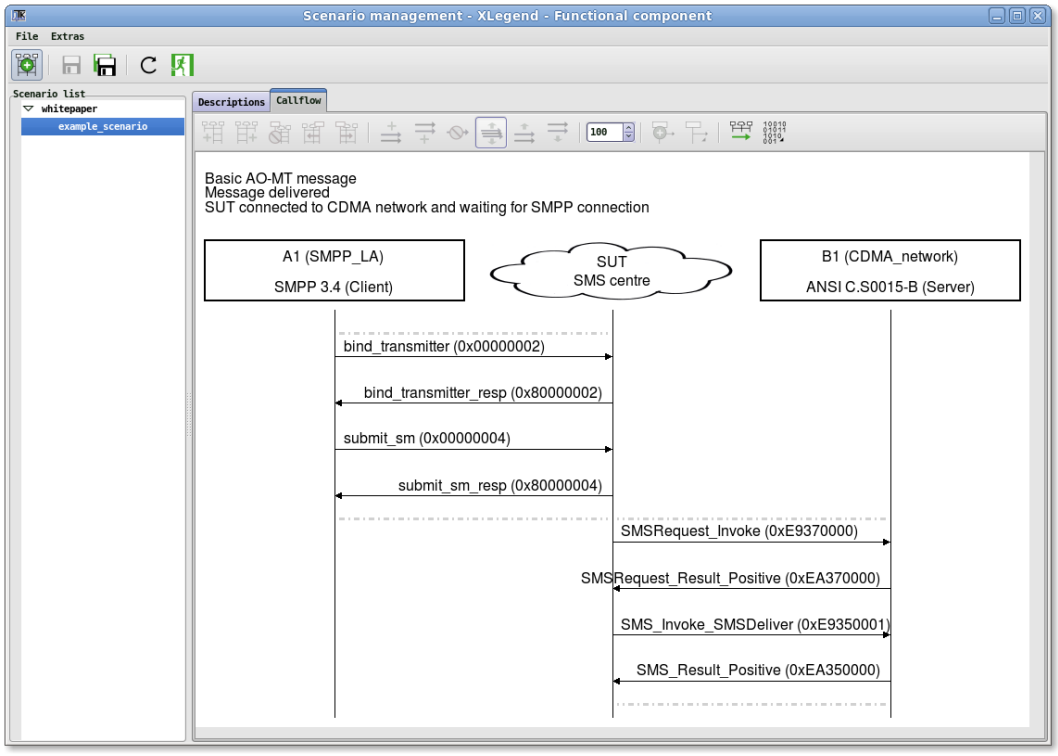
The following screenshots display different windows of XLegend used for functional test configuration and execution.

### Figure 3. XLegend main window



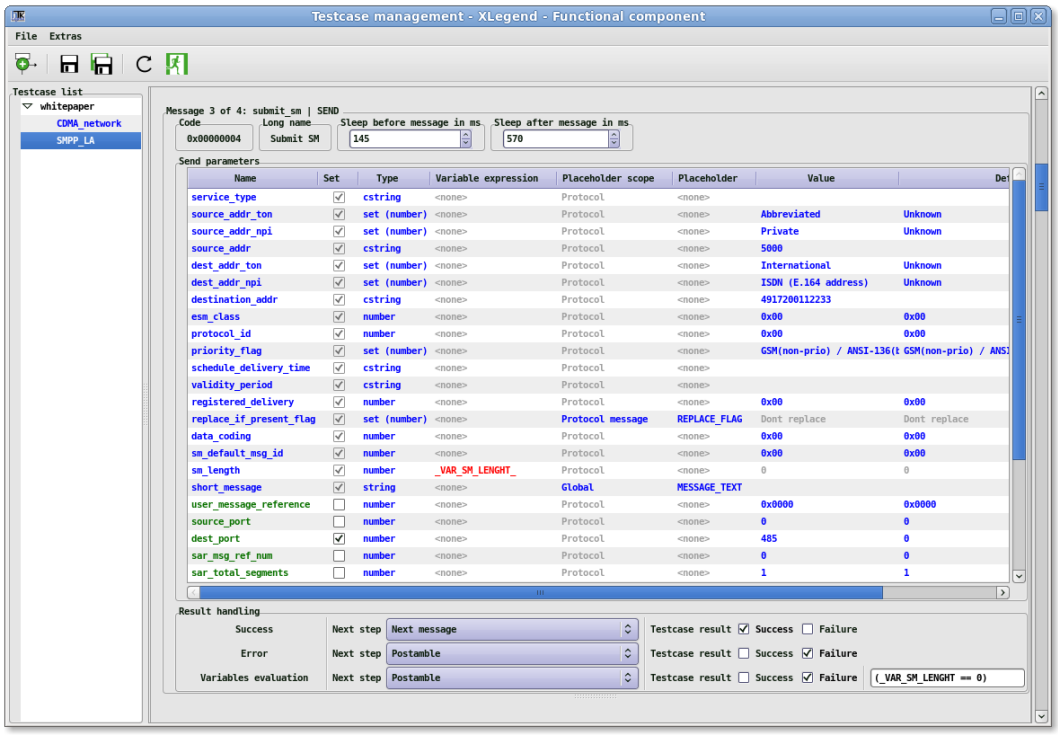

The above figure displays main XLegend window providing access to all features of Funtional component. In this example the user *whitepaper* does not have the privilege to restart the Traffic Generator, therefore the *Restart TG* button is inactive.

**Figure 4. XLegend scenario management window**




The above figure displays the scenario management window with an example of basic SMS delivery callflow.

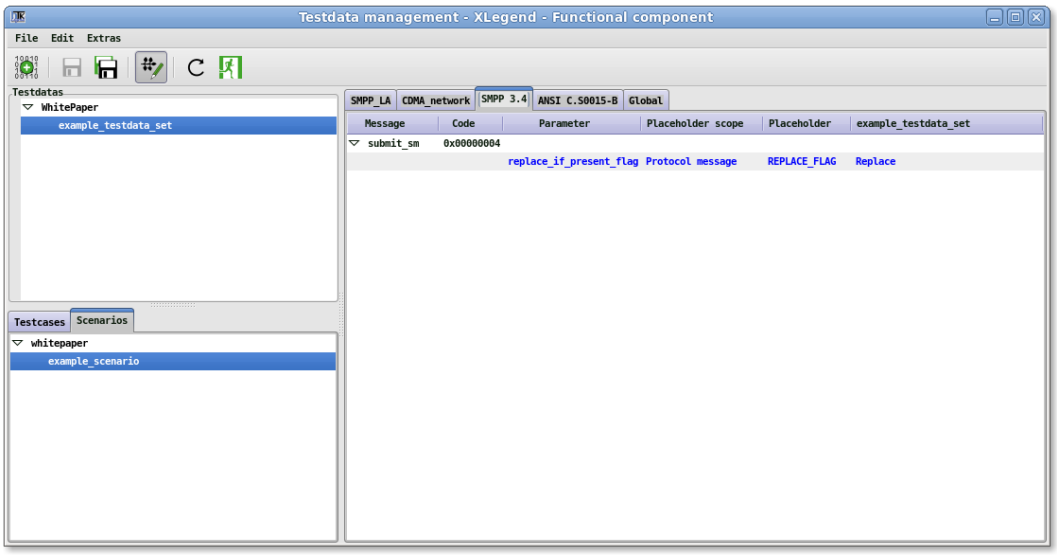
**Figure 5. XLegend testcase management window**




The above figure displays an example of parameters configuration of Submit SM message of SMPP protocol. The testcase will wait 145 ms before and 570 ms after sending this message. Fixed values and

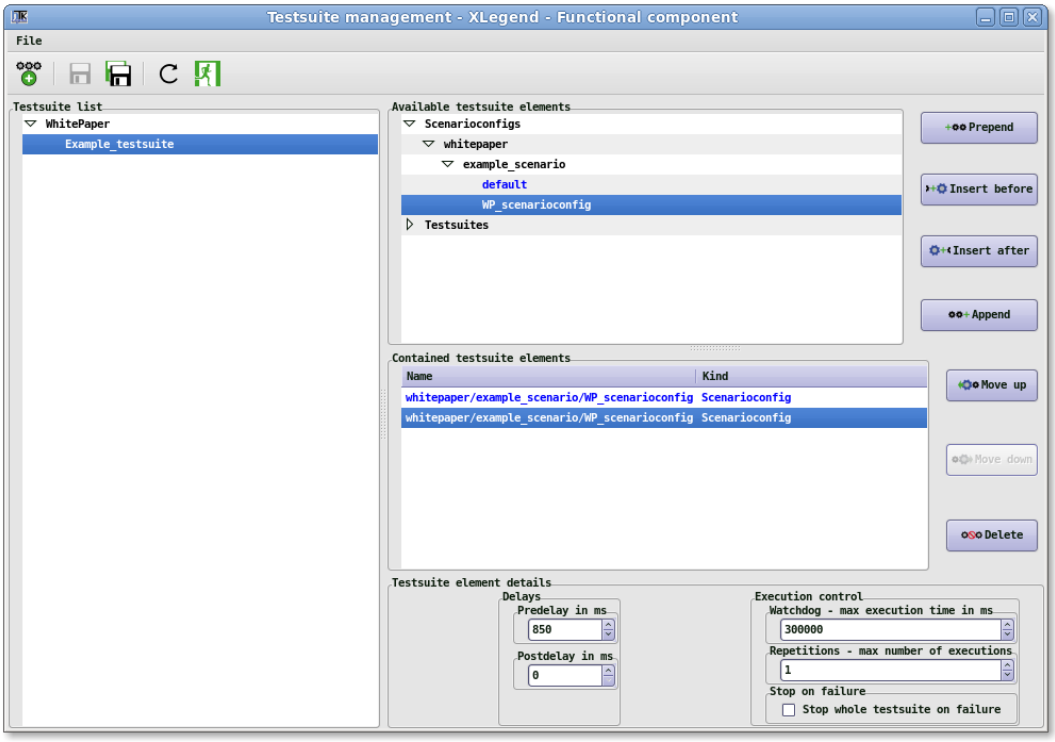
placeholders are used for the parameter values, the variable of parameter *sm_lenght* is being evaluated and message execution is considered as successful if message is sent out.

**Figure 6. XLegend testdata management window**



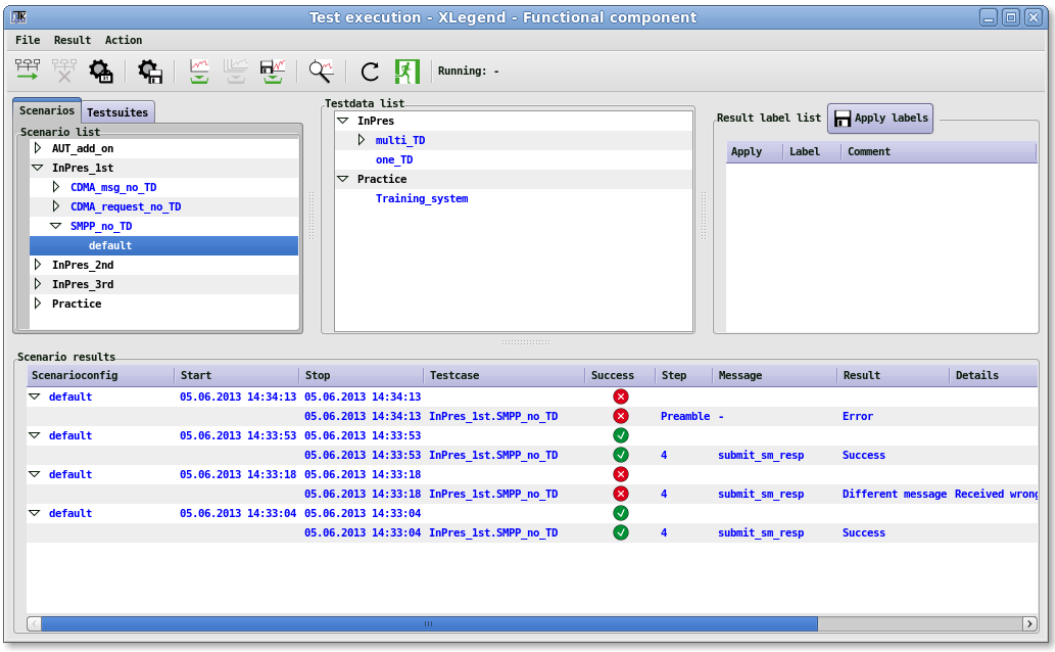

The above figure displays an example of testdata definition for particular scenario. The actual parameter value will be used during scenario execution.

**Figure 7. XLegend testsuite management window**



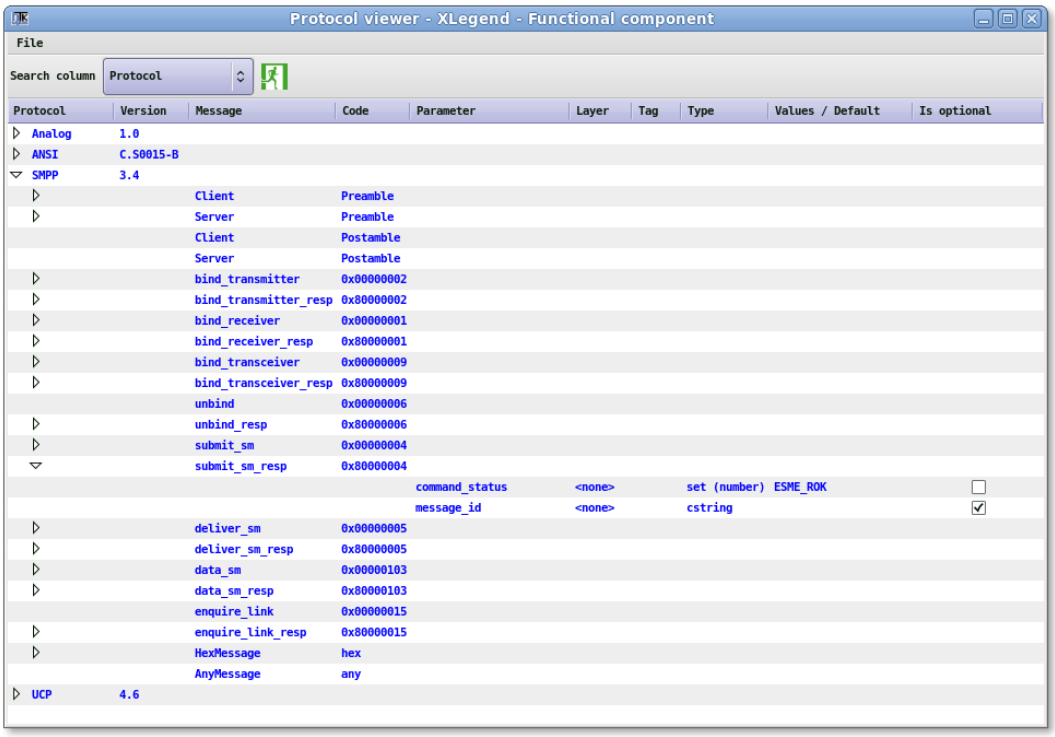

The above figure displays an example of testsuite creation where one scenarioconfig will be executed twice; each time with different Predelay and Postdelay settings.

**Figure 8. XLegend test execution window**




The above figure displays the result information of two executions of a scenario; each time with different result due to different reaction of the System Under Test.

**Figure 9. XLegend protocol definition window**




The above figure displays the set of parameters in *submit_sm_resp* message of the SMPP protocol in the Protocol viewer window.

# Prerequisites

hTK-Legend Test Framework is installed on two different systems, the Traffic Generator host and the graphical user interface XLegend host. Following sections contain information about their hardware and software requirements.

## Hardware requirements

The host system for Traffic Generator must be powerful enough to run modern Linux or Sun Solaris operating systems and must have fully working USB port. This USB port is needed for WIBU Systems' CodeMeter USB stick with valid license which must be present during all testing (note: connection through USB hub is not supported).

The only hardware requirement for XLegend is a screen resolution of at least 1024x768 pixel. A higher resolution is strongly recommended.

## Software requirements

The Traffic Generator supports two operating system families and requires following software:

– Linux OS: kernel version 2.6.32 or higher, lksctp-tools version 1.0.2 or higher, CodeMeter Runtime version 4.30 or higher, modules usb-mass-storage and scsi-generic-device, libsqlite3 version 3.6.17 or higher, libxml2 version 2.7.3 or higher

– Sun Solaris OS: version 5.10 or higher, CodeMeter Runtime version 4.40 or higher, SUNWusb* packages

The XLegend supports Linux, Windows (XP, Vista and 7) and Solaris operating systems and requires GTK+ library version 2.12.x or higher

## Successfully tested hardware and software

The Traffic Generator with Functional component has been deployed and used successfully, amongst others, on the following platforms:

- Server HP®, 2x Intel® Xeon® E5520 2.27GHz, 36GB RAM, Red Hat® EL 6.2 64bit
- Server HP®, 1x Intel® Xeon® L5220 2.27GHz, 4GB RAM, Red Hat® EL 6.7 64bit
- Server Sun® Fire T2000, UltraSparc® T1 32 Cores 1.0GHz, 32GB RAM, Solaris 10
- Desktop Intel® Core® i7-2600K 3.4GHz, 8GB RAM, HDD WD® 500GB, CentOS® 7.1 64bit
- Notebook Apple® MacBook® 13", Intel® Core® 2 T7400 2.16GHz, 2GB RAM, SSD 128GB, CentOS® 6.7 64bit