|  |  |
| --- | --- |
|  |  |

EPTF CLL Improved Log, User Guide

Contents

[1 Introduction 2](#_Toc335038115)

[1.1 Revision history 2](#_Toc335038116)

[1.2 About this Document 2](#_Toc335038117)

[1.2.1 How to Read this Document 2](#_Toc335038118)

[1.2.2 References 2](#_Toc335038119)

[1.2.3 Abbreviations 3](#_Toc335038120)

[1.2.4 Terminology 3](#_Toc335038121)

[1.3 System Requirements 3](#_Toc335038122)

[2 ILog 3](#_Toc335038123)

[2.1 Overview 3](#_Toc335038124)

[2.2 Description of files in this feature 3](#_Toc335038125)

[2.3 Description of required files from other features 4](#_Toc335038126)

[2.4 Installation 4](#_Toc335038127)

[2.5 Configuration 5](#_Toc335038128)

[3 Usage 5](#_Toc335038129)

[3.1 ILogBase 5](#_Toc335038130)

[3.2 ILog 6](#_Toc335038131)

[3.3 ILogString 6](#_Toc335038132)

[4 Warning messages 7](#_Toc335038133)

[5 Error messages 7](#_Toc335038134)

[6 Examples 7](#_Toc335038135)

[6.1 Configuration file 8](#_Toc335038136)

[6.2 Demo Module 8](#_Toc335038137)

# Introduction

## Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Rev | Characteristics | Prepared |
| 2012-06-22 | PA1 | First draft version | <EILDVNC> |

## About this Document

### How to Read this Document

This is the User Guide for the ILog feature of the Ericsson Performance Test Framework (TitanSim), Core Load Library (CLL). TitanSim CLL is developed for the TTCN-3 [1] Toolset with TITAN [2]. This document should be read together with the Function Description of the ILog feature [6]. For more information on the TitanSim CLL please consult the Product Revision Information [3], the Function Specification [4] and the Users Guide [5] of the TitanSim.

### References

1. ETSI ES 201 873-1 v3.2.1 (2007-02)  
   The Testing and Test Control Notation version 3. Part 1: Core Language
2. 1/198 17-CRL 113 200 Uen  
   User Guide for the TITAN TTCN-3 Test Executor
3. 109 21-CNL 113 512-2 Uen   
   TitanSim CLL for TTCN-3 toolset with TITAN, Product Revision Information
4. 155 17-CNL 113 512 Uen   
   TitanSim CLL for TTCN-3 toolset with TITAN, Function Specification
5. 198 17-CNL 113 512 Uen  
   TitanSim CLL for TTCN-3 toolset with TITAN, User Guide
6. 35/155 16-CNL 113 512 Uen  
   EPTF CLL Improved Logging Function Description
7. TitanSim CLL for TTCN-3 toolset with TITAN, Reference Guide  
   <http://ttcn.ericsson.se/products/libraries.shtml>

### Abbreviations

CLL Core Load Library

EPTF Ericsson Load Test Framework, formerly TITAN Load Test Framework

TitanSim Ericsson Load Test Framework, formerly TITAN Load Test Framework

TTCN-3 Testing and Test Control Notation version 3 [1]

ILog Improved Log

### Terminology

*TitanSim Core (Load) Library(CLL)* is that part of the TitanSim software that is totally project independent. (I.e., which is not protocol-, or application-dependent). The TitanSim CLL is to be supplied and supported by the TCC organization. Any TitanSim CLL development is to be funded centrally by Ericsson

## System Requirements

In order to use the ILog feature the system requirements listed in TitanSim CLL User Guide [5] should be fulfilled.

# ILog

## Overview

Logging every traffic case not only can result in unnecessarily huge log files, but it even makes logs less useable thus harder to find the cause of the failure. The EPTF CLL ILog feature makes it possible not to log successful traffic cases, only the failed ones.

## Description of files in this feature

The EPTF CLL Improved Logging API includes the following files:

* ILogBase
  + EPTF\_CLL\_ILogBase\_Definitions.ttcn: This TTCN-3 module contains common type definitions that should be used in all ILog components.
  + EPTF\_CLL\_ILogBase\_Functions.ttcn: This TTCN-3 module contains the implementation of ILog functions.
* ILogString:
  + EPTF\_CLL\_ILogString\_Definitions.ttcn: This TTCN-3 module contains the type definitions of the ILogString feature.
  + EPTF\_CLL\_ILogString\_Functions.ttcn: This TTCN-3 module contains the function implementations of the ILogString feature.
* ILog:
  + EPTF\_CLL\_ILog\_Definitions.ttcn: This TTCN-3 module contains the type definitions of the ILog feature.
  + EPTF\_CLL\_ILog\_Functions.ttcn: This TTCN-3 module contains the function implementations of the ILog feature.

## Description of required files from other features

The ILog feature is part of the TitanSim EPTF Core Load Library (CLL). It relies on several features of the CLL. The user has to obtain the products/files to be found under the respective feature names:

* Base
* Common
* FreeBusyQueue
* HashMap
* LGenBase
* NQueue
* TCCUsefulFunctions

## Installation

Since EPTF CLL ILog is used as a part of the TTCN-3 test environment this requires TTCN-3 Test Executor to be installed before any operation of these functions. For more details on the installation of TTCN-3 Test Executor see the relevant section of [2].

If not otherwise noted in the respective sections, the following are needed to use EPTF\_CLL\_ILog:

* Copy the files listed in section [2.2, 2.3] to the directory of the test suite or create symbolic links to them.
* Import the Improved Logging demo or write your own application using ILog.
* Create Makefile or modify the existing one. For more details see the relevant section of [2].
* Edit the config file according to your needs. See the following section [2.5].

## Configuration

The executable test program behaviour is determined via the run-time configuration file. This is a simple text file, which contains various sections. The usual suffix of configuration files is .cfg. For further information on the configuration file see [2].

The ILog feature defines TTCN-3 module parameters as defined in [2] clause 4. Actual values of these parameters – when there is no default value or a different from the default actual value to be used – shall be given in the [MODULE\_PARAMETERS] section of the configuration file.

This feature does not contain any ILog–specific module parameters.

# Usage

The ILog feature consists of three modules; each of them can be used independently.

## ILogBase

The ILogBase is the common layer. It is provides the base functionality and can be used by the user or by other modules as well.

In order to use the EPTF CLL ILogBase feature the user component should extend the EPTF\_ILogBase\_CT component.

This component should be initialized by the function f\_EPTF\_ILogBase\_init\_CT before other functions of ILogBase can be used.

In order to use this feature, the user has to create one or more LogHeader and a LogItem functions, which create the log based on the chains that contain the log items.

There has to be at least one user side database, which contains the log items. The record type, used for this database, depends on the user. The database can be registered in the ILogBase component by using the f\_EPTF\_ILogBase\_register\_DB function. This function returns with an integer value, which is the Id of that database. This Id has to be used, when a new element is added to a chain.

The chain can be created by using the f\_EPTF\_ILogBase\_newChain function. This function also returns an integer value that is the Id of that chain. When a new log item is joined to a chain with the f\_EPTF\_ILogBase\_addToChains function, the Ids returned by the previous functions have to be used. This function returns with the element index of the new log item. This index should be use to store the item in the user side database.

The chains can be logged out or deleted at the end of the operation.

## ILog

In the first step, shall be implemented the ILog feature in application libraries handling session-based protocols.

A supported possible use case can be the following:

* The application writes a detailed log, and a short summary. In case of successful message flow, the short summary is written down. In case of failed message flow, the short summary is written down (like a header), and then the detailed log.
* The test steps (provided by the application and the application libraries) and the transport layer always log into the detailed log.
* The test steps mentioned above can log into the short summary too. This logging can be turned on/off by test steps. This way the users can manage from the FSM, which steps to write into the short description.

Several test steps can be used to start a traffic flow in a scenario, and in another scenario they are inside a traffic flow. Only the business logic implemented in the Application level can decide when to start a new chain. Therefore the chains can be started by an FSM test step and obviously logging out the chains can be initiated by an FSM test step too.

The user can log out both the detailed and the short chain in case of failed traffic with using the *f\_EPTF\_ILog\_step\_logSuccess* function, and only the short chain in case of successful traffic by using the *f\_EPTF\_ILog\_step\_logFailed* function. It is also possible to delete both chains, when the traffic is successfully ended; this functionality is implemented in the *f\_EPTF\_ILog\_step\_deleteLog* function.

This layer of the feature can be used on the application and the AppLib levels. All the functions are accessible that are provided by the ILogBase layer, which means that the user can register new databases, create new detailed and short chain pairs, add elements to the log chains and log out and delete the log chains by using only this upper layer.

## ILogString

The ILogString feature provides charstring logging functionality. It has a built-in database that has records, which consist of a charstring and a float value that contains a timestamp.

To use this feature, it has to be initialized with the f\_EPTF\_ILogString\_init\_CT function. The initialization creates the built-in database as well. The functionalities of this feature are the same as those ones provided by the ILogBase. The difference between these two layers is that in case of ILogString, the user has to store only the Ids of the chains that were created; everything else is administered by the feature. The log item and the log header functions are also provided by the ILogString layer, which means that these functions are not customizable by the user.

# Warning messages

Please note, that besides the below described warning messages, warning messages shown in [2] or those of other used features or product may also appear.

The ILog, ILogBase and ILogString feature may log one or more of the following warning messages:

* “f\_EPTF\_ILog\_newChains: There is unprocessed chain data! Please log or delete it previously!”
* “f\_EPTF\_ILog\_addToChains: No valid chain was given in the list, which is currently in use!”
* “f\_EPTF\_ILog\_step\_logSuccess: No valid chain was given in the list, which is currently in use!”
* “f\_EPTF\_ILog\_step\_logFailed: No valid chain was given in the list, which is currently in use!”
* “f\_EPTF\_ILog\_step\_deleteLog: No valid chain was given in the list, which is currently in use!”
* “f\_EPTF\_ILogBase\_newChain: Chain id inconsistency!”
* “f\_EPTF\_ILogBase\_addToChains: Invalid database id: “<pl\_dbId>””
* “f\_EPTF\_ILogBase\_addToChains: No valid chain was given in the list, which is currently in use!”
* “f\_EPTF\_ILogBase\_addToChains: There is no detailed chain in use!”
* “f\_EPTF\_ILogBase\_getName: Invalid chain, id: “<pl\_chainId>””
* “f\_EPTF\_ILogBase\_deleteChain: Invalid chain, id: “<pl\_chainId>””
* “f\_EPTF\_ILogBase\_logChain: Invalid chain, id: “<pl\_chainId>””
* “f\_EPTF\_ILogBase\_logAll: No chain to log!”
* “f\_EPTF\_ILogString\_addToChains: Could not add the log item to any of the chains in the chain list! Save of the log item is failed!”

# Error messages

Please note, that besides the below described error messages, error messages shown in [2] or those of other used features or product may also appear.

There are no error messages for ILog.

# Examples

The “demo” directory of the deliverable contains the following examples:

* EPTF\_ILog\_demo.cfg
* EPTF\_ILog\_demo.ttcn

## Configuration file

The used configuration file (EPTF\_ILog\_demo.cfg) is for the ILog example, it is placed in the demo directory.

## Demo Module

The demo module EPTF\_ILog\_demo.ttcn illustrates a typical usage of the ILog feature. It is placed in the demo directory of the released feature.