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

EPTF CLL Central Scheduling, User Guide

Contents

[1 Introduction 2](#_Toc301275479)

[1.1 Revision history 2](#_Toc301275480)

[1.2 About this Document 2](#_Toc301275481)

[1.2.1 How to Read this Document 2](#_Toc301275482)

[1.2.2 References 2](#_Toc301275483)

[1.2.3 Abbreviations 3](#_Toc301275484)

[1.2.4 Terminology 3](#_Toc301275485)

[1.3 System Requirements 3](#_Toc301275486)

[2 Central Scheduling 3](#_Toc301275487)

[2.1 Overview 3](#_Toc301275488)

[2.2 Description of files in this feature 3](#_Toc301275489)

[2.3 Description of required files from other features 4](#_Toc301275490)

[2.4 Installation 4](#_Toc301275491)

[2.5 Configuration 5](#_Toc301275492)

[2.5.1 Module Parameters of the Central Scheduling Feature 5](#_Toc301275493)

[2.5.2 Configuring Traffic Cases 6](#_Toc301275494)

[2.5.3 Configuring Load Generators 6](#_Toc301275495)

[2.5.4 DataSource function of Central Scheduling 6](#_Toc301275496)

[3 Error messages 6](#_Toc301275497)

[4 Warning messages 7](#_Toc301275498)

[4.1.1 DataSource warning messages 8](#_Toc301275499)

[5 Examples 9](#_Toc301275500)

[5.1 Makefile 9](#_Toc301275501)

# Introduction

## Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Rev | Characteristics | Prepared |
| 2008-02-11 | PA1 | First draft version | EGBOTAT |
| 2008-03-17 | PA2 | New revision | EGBOTAT |
| 2009-04-06 | PB1 | Removed CBE Compatibility module | EGBOTAT |
| 2009-04-27 | PB2 | Corrections after review | EGBOTAT |
| 2011-08-16 | PC1 | DataSource warning messages | EMIHMIK |

## About this Document

### How to Read this Document

This is the User Guide for the Central Scheduling 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 Central Scheduling feature [6]. For more information on the TitanSim CLL please consult the Product Revision Information [3], the Users Guide [4] and the Function Specification [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. 24/155 16-CNL 113 512  
   EPTF CLL Central Scheduling, Function Description
7. TitanSim CLL for TTCN-3 toolset with TITAN, Reference Guide  
   <http://ttcn.ericsson.se/products/libraries.shtml>
8. 7/198 17-CNL 113 512 Uen  
   EPTF CLL LGenBase, User Guide

### Abbreviations

CLL Core Load Library

EPTF Ericsson Load Test Framework, formerly TITAN Load Test Framework

GUI Graphics User Interface

TitanSim Ericsson Load Test Framework, formerly TITAN Load Test Framework

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

### 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 Central Scheduling feature the system requirements listed in TitanSim CLL User Guide [5] should be fulfilled.

# Central Scheduling

## Overview

The EPTF Central Scheduling feature makes it possible to write load test cases using similar methodology than in function test. Load testing means executing several traffic cases in parallel.

Scheduling of the traffic case triggering is performed by the Central Scheduling Application Admin. The triggering is done by sending execution burst messages to load generators which will execute the selected traffic case(s). The load generator and the interface between the admin and the load generator must be implemented by the user.

## Description of files in this feature

The EPTF CLL Central Scheduling API includes the following files:

* Central Scheduling
  + EPTF\_CLL\_CS\_Definitions.ttcnpp: This TTCN-3 module contains common type definitions that should be used in all Central Scheduling Components.
  + EPTF\_CLL\_CS\_Admin\_Functions.ttcnpp: This TTCN-3 module contains the implementation of the Central Scheduling Application Admin Base functions.
  + EPTF\_CLL\_CS\_LB\_Functions.ttcnpp: This TTCN-3 module contains the implementation of the Load Balancing for Central Scheduling.
  + EPTF\_CLL\_CS\_UIHandler\_Definitions.ttcnpp: This module contains common type definitions that should be used if a GUI is needed for Central Scheduling.
  + EPTF\_CLL\_CS\_UIHandler\_Functions.ttcnpp: This module contains the implementation of the Central Scheduling with GUI support.

## Description of required files from other features

The Central Scheduling feature is part of the TitanSim EPTF Core Load Library (CLL). It relies on several features of the CLL. To use the Central Scheduling, the user has to obtain the respective files from the following features:

* Common
* Base
* ExecCtrl
* LGenBase
* FreeBusyQueue
* UIHandler
* Variable

## Installation

Since EPTF\_CLL\_CentralScheduling 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\_CentralScheduling:

* Copy the files listed in section [0, 2.3] to the directory of the test suite or create symbolic links to them.
* Import the Central Scheduling demo or write your own application using Central Scheduling.
* 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 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].

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

### Module Parameters of the Central Scheduling Feature

**tsp\_EPTF\_CS\_LGenHostnameList**

This EPTF\_CharstringList type module parameter is defined in module EPTF\_CLL\_CS\_Definitions. It is used to define the Hostnames on which load generators are to be created. Its default value is **{}**.

**tsp\_debug\_EPTF\_CLL\_CS\_LB\_Functions**

This boolean type module parameter is defined in module EPTF\_CLL\_CS\_LB\_Functions. It can be used for debugging purposes. Its default value is **false**.

**tsp\_debug\_EPTF\_CS\_Admin\_Functions**

This boolean type module parameter is defined in module EPTF\_CLL\_CS\_Admin\_Functions. It can be used for debugging purposes. Its default value is **false**.

**tsp\_EPTF\_CLL\_CS\_Admin\_LgenStatusGuard**

This float type module parameter is defined in module EPTF\_CLL\_CS\_Admin\_Functions. It defines the maximum time a load generator can take to initialize and send its initial status message. After this time expires with no status received, the Admin stops with an error message. The default value of this parameter is **10.0** for 10 seconds.

**tsp\_EPTF\_CLL\_CS\_Admin\_LGenCreationDelay**

This float type module parameter is defined in module EPTF\_CLL\_CS\_Admin\_Functions. It defines a delay between successive creation of load generators. Its default value is **0.1**.

### Configuring Traffic Cases

Entity group for traffic cases can be created via the LGenBase module parameter tsp\_LGenBase\_EntityGrpDeclarators with the entity name CS\_Entity\_Tc<suffix> where <suffix> is defined by v\_EPTF\_CS\_EntityName\_Suffix in the user init function.

Traffic cases and scenarios can be configured via the following LGenBase module parameters:

* tsp\_LGenBase\_TcMgmt\_tcTypeDeclarators2
* tsp\_LgenBase\_TcMgmt\_WeightedScenarioDeclarators
* tsp\_LGenBase\_Scenarios2Grps

For details, see [8]. For an example, see the demo in the demo directory.

### Configuring Load Generators

The Central Scheduler Application Admin declares entities of type CS\_Entity\_LGen<suffix> where <suffix> is defined by calling the function f\_EPTF\_CS\_Admin\_setEntityNameSuffix in the user init function. These entities can be used to define the number of load generators used by a specific type of Central Scheduling. This can be done by creating an entity group of this entity type via the LGenBase module parameter tsp\_LGenBase\_EntityGrpDeclarators.

No scenario must be stared on the load generator entities. Load generator entities are handled automatically without a scenario and their entities and entity group are only needed to be able to split load generators between multiple Admins using the ExecCtrl feature.

### DataSource function of Central Scheduling

See the natural documentation of the CentralScheduling feature.

# 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.

**f\_EPTF\_CS\_Admin\_declareTrafficCase: redeclaration of traffic case <traffic-case-name>**

The traffic case with <traffic-case-name> was declared more than once in the user defined initialization of the Central Scheduling Application Admin.

**f\_EPTF\_CS\_Admin\_declareTrafficCase: traffic cases should be declared in user defined init function.**

The function f\_EPTF\_CS\_Admin\_declareTrafficCase was called after initialization and creation of the internal traffic case type/selector lookup table.

**f\_EPTF\_CS\_Admin\_sanityCheck: missing initialization of mandatory function reference ‘<function-reference>’**

A mandatory function reference was not initialized in the user defined initialization function.

**f\_EPTF\_CS\_Admin\_initFSM: Error while declaring FSM events**

The admin could not initialize the FSM.

**f\_EPTF\_CS\_handleLGenStatusMsg: problem reported by LGen during initialization**

The load generator sent disabled status after its creation and start.

**f\_EPTF\_CS\_lookupBySenderAddress: component reference not found in LGen database**

Invalid component reference was passed to f\_EPTF\_CS\_lookupBySenderAddress.

**f\_EPTF\_CS\_Admin\_buildTcTypeIdx2tcSelector: undefined traffic case <traffic-case-name> for entity type <entity-type>**

The configuration file contains a traffic case that was not declared in the user initialization function of the Central Scheduling Application Admin.

**f\_EPTF\_CS\_Admin\_scenarioCreated: No scenario should be started on LGen entities.**

The configuration file tried to assign a scenario to the load generator entities.

**f\_EPTF\_CS\_Admin\_createLGens: timeout when waiting for LGen status message**

The created load generator did not send a status message within the status guard time.

# 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.

**f\_EPTF\_LB\_selectLoadGen: WARNING: no free LGens found.**

The load balancing feature did not find a free load generator.

**f\_EPTF\_CS\_Admin\_declareTrafficCase: WARNING: traffic case selector <selector> assigned to traffic case <traffic-case> is already used by <another-traffic-case>**

The traffic case selector <selector> was assigned to more than one traffic case.

**f\_EPTF\_CS\_Admin\_sanityCheck: WARNING: no traffic case was declared. Traffic case selectors will be auto-assigned.**

No traffic case was declared from the user init function. Traffic cases will be assigned a selector automatically, in the order of definition in the configuration file.

**WARNING: f\_EPTF\_CS\_SelectAndTriggerLGen: no trigger message is buffered.**

f\_EPTF\_CS\_SelectAndTriggerLGen was called without traffic cases in the buffer.

**WARNING: f\_EPTF\_CS\_SelectAndTriggerLGen: no free LGen: saturation!!!**

No free load generators are available to send the trigger to.

### DataSource warning messages

**f\_EPTF\_CS\_DSProcessData: Invalid iterator or externalData or parameter:**

This warning message is reported, when the DataSource functionality is used and something with the request was wrong. Wrong iterator or externalData or parameter(s) were used.

**f\_EPTF\_CS\_DSProcessData: Unhandled element:**

This warning message is reported, when the DataSource functionality is used and the current data element is unknown for the feature.

**f\_EPTF\_CS\_getParams: Parameters are not correct:**

This warning message is reported, when the DataSource functionality is used and the parameters of the current data element are not correct. For example the name of the parameter was incorrect.

**f\_EPTF\_CS\_getParams: Too few parameter is given:**

This warning message is reported, when the DataSource functionality is used and the parameters of the current data element are less than required.

**f\_EPTF\_CS\_getParams: Too many parameters are given:**

This warning message is reported, when the DataSource functionality is used and the parameters of the current data element are more than required.

# Examples

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

* Common/Common\_Definitions.ttcn
* EPTF\_CS\_demo.cfg
* EPTF\_CS\_demo.prj
* LoadRegulator/LoadRegulatro\_Definitions.ttcn
* LoadRegulator/LoadRegulatro\_Functions.ttcn
* Main/MainModule.ttcn
* protocolA/A\_defs.ttcn
* protocolA/CS\_A\_Definitions.ttcn
* protocolA/CS\_A\_Functions.ttcn
* protocolA/LGen\_A\_Functions.ttcn
* protocolB/B\_defs.ttcn
* protocolB/CS\_B\_Definitions.ttcn
* protocolB/CS\_B\_Functions.ttcn
* protocolB/LGen\_B\_Functions.ttcn

## Makefile

The demo uses the HostAdmin feature of the CLL which needs the Makefile script demo/HostAdmin/makefile\_patch.sh to be executed after Makefile generation.