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

EPTF CLL Host Admin, Function Description

Contents

[1 Introduction 2](#_Toc310943620)

[1.1 Revision history 2](#_Toc310943621)

[1.2 How to Read this Document 2](#_Toc310943622)

[1.3 References 2](#_Toc310943623)

[1.4 Scope 3](#_Toc310943624)

[1.5 Recommended way of reading 3](#_Toc310943625)

[1.6 Typographical conventions 3](#_Toc310943626)

[1.7 Abbreviations 3](#_Toc310943627)

[1.8 Terminology 3](#_Toc310943628)

[2 General Description 3](#_Toc310943629)

[2.1 Host Admin Server 4](#_Toc310943630)

[3 Functional Interface 4](#_Toc310943631)

[3.1 Naming Conventions 5](#_Toc310943632)

[3.2 Host Admin Base Public Functions 5](#_Toc310943633)

[3.2.1 Initialization 5](#_Toc310943634)

[3.2.2 Cleanup 5](#_Toc310943635)

[3.2.3 Updating measurement data 5](#_Toc310943636)

[3.2.4 Get the name of a process 5](#_Toc310943637)

[3.2.5 Add process statistics 5](#_Toc310943638)

[3.2.6 Remove process statistics 5](#_Toc310943639)

[3.2.7 Get CPU load of a process 6](#_Toc310943640)

[3.2.8 Get memory usage of a process 6](#_Toc310943641)

[3.3 Host Admin Public Functions 6](#_Toc310943642)

[3.3.1 Behaviour function 6](#_Toc310943643)

[3.4 Host Admin Server Public Functions 6](#_Toc310943644)

[3.4.1 Initialization 6](#_Toc310943645)

[3.4.2 Setting the Sort Order 6](#_Toc310943646)

[3.4.3 Getting Variable Name Prefix for Process Stats 7](#_Toc310943647)

[3.5 Summary Table of all public functions for EPTF Host Admin 7](#_Toc310943648)

[3.6 DataSource Client functionality 8](#_Toc310943649)

# Introduction

## Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Rev | Characteristics | Prepared |
| 2007-11-27 | PA1 | First draft version | EGBOTAT |
| 2007-12-06 | PA2 | Final version after review | EGBOTAT |
| 2008-02-14 | PA3 | New version, new test port added | EBENMOL |
| 2010-01-22 | PB1 | Added process statistics | EGBOTAT |
| 2011-07-19 | PC1 | Added dataSource Client functionality | EJNOSVN |
| 2011-09-30 | PD1 | Added Host Admin Server | EGBOTAT |

## How to Read this Document

This is the Function Description for the Host Admin of the Ericsson Performance Test Framework (TitanSim), Core Load Library (CLL). TitanSim CLL is developed for the TTCN-3 [1] Toolset with TITAN [2]. For more information on the TitanSim CLL please consult the Product Revision Information [3].

## 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. TitanSim CLL for TTCN-3 toolset with TITAN, Reference Guide  
   <http://ttcn.ericsson.se/products/libraries.shtml>
6. 18/155 16-CNL 113 512 Uen  
   EPTF CLL Variable, Function Description
7. 2/155 16-CNL 113 512 Uen  
   EPTF\_CLL\_UIHandler, Function Description

## Scope

This document is to specify the content and functionality of the Host Admin feature of the TitanSim CLL.

## Recommended way of reading

The readers are supposed to get familiar with the concept and functionalities of TitanSim CLL [4]. They should get familiar with the list of acronyms and the glossary in Section 1.7 and 1.8, respectively.

## Typographical conventions

Important concepts are denoted by *italic* font wherever they are first used in the given context.

## Abbreviations

CLL Core Load Library

EPTF Ericsson Load Test Framework, formerly TITAN Load Test Framework

MTC Main Test Component

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

*EPTF Variable* An enhanced component variable that makes it possible to access the values of other Variables in remote components

# General Description

The aim of the EPTF CLL Host Admin feature is to provide processor and memory usage measurement for a given host it is started on. Supported platforms are Solaris 5.6 and 5.8, Linux and Win32/Cygwin.

This document specifies the Host Admin feature of the TitanSim CLL.

The EPTF Host Admin feature makes it possible to

* Measure average processor load of the host it is started on
* Query the number of processors
* Measure per-processor load
* Query the physical memory size
* Get the amount of free (available) memory
* Monitor the CPU load and memory usage of processes

The Host Admin feature is divided in two parts: Host Admin Base and Host Admin. The former provides local measurements for components that extend it, and the latter provides remote measurements via EPTF Variables [6] (or a local measurement interface running in a separate component).

To be able to use EPTF CLL Host Admin Base, the user component should extend the EPTF\_HostAdmin\_Base\_CT component.

To be able to use the EPTF CLL Host Admin, a new component of type EPTF\_HostAdmin\_CT should be created and the function f\_EPTF\_HostAdmin\_behavior() should be started on it. The measurement data can be accessed by subscribing to the respective EPTF variables.

## Host Admin Server

There is an additional component called Host Admin Server as a sub-feature of Host Admin. This component periodically queries the running EPTF processes from the Base feature. If it finds a process that was started on a new host, it automatically starts a Host Admin component on that host. Host and process statistics are collected from the started Host Admins by the server and provided as data sources/Variables.

The process statistics that are colleted by the Host Admin Server are oly for EPTF components. These can be displayed on the GUI per host, sorted either by name, CPU load or memory usage. Statistics are also available by the selfName parameter of the EPTF processes. An example use-case can be seen in the demo of the feature.

# Functional Interface

Apart from this description a cross-linked reference guide for the TitanSim CLL Functions can be reached for on-line reading [5].

## Naming Conventions

All functions in Host Admin Base have the prefix *f\_EPTF\_HostAdmin\_Base*.

All functions in Host Admin have the prefix *f\_EPTF\_HostAdmin*.

## Host Admin Base Public Functions

### Initialization

Before using the EPTF Host Admin Base feature the

*f\_EPTF\_HostAdmin\_Base\_InitMeasInterface()*

function should be called. This initializes the EPTF Host Admin Base feature.

### Cleanup

The function *f\_EPTF\_HostAdmin\_Base\_CleanupMeasInterface()* should be called before stopping the component.

### Updating measurement data

The function *f\_EPTF\_HostAdmin\_Base\_update()* can be used to update the measurement data stored as component variables in the component EPTF\_HostAdmin\_Base\_CT.

### Get the name of a process

The function f\_EPTF\_HostAdmin\_Base\_getProcessName() function returns the name (command) of a process by its PID.

### Add process statistics

The function f\_EPTF\_HostAdmin\_Base\_addProcessStat() function can be used to start measurement of CPU and memory usage of a process. The PID of the process has to be passed in as a parameter. The function returns the internal index of the process statistics.

### Remove process statistics

The function f\_EPTF\_HostAdmin\_Base\_removeProcessStat() function can be used to remove process statistics previously added by f\_EPTF\_HostAdmin\_Base\_addProcessStat, with the PID of the process as the input parameter.

The function f\_EPTF\_HostAdmin\_Base\_removeProcessStatByIdx() function is the same as f\_EPTF\_HostAdmin\_Base\_removeProcessStat, but the statistics index (as returned by f\_EPTF\_HostAdmin\_Base\_addProcessStat) has to be specified instead of the PID.

### Get CPU load of a process

The function f\_EPTF\_HostAdmin\_Base\_getProcessCpuLoad() and f\_EPTF\_HostAdmin\_Base\_getProcessCpuLoadByIdx() can be used to get the CPU load of a process by its PID or statistics index respectively.

### Get memory usage of a process

The function f\_EPTF\_HostAdmin\_Base\_getProcessMemUsage() and f\_EPTF\_HostAdmin\_Base\_getProcessMemUsageByIdx() can be used to get the memory usage of a process.

## Host Admin Public Functions

### Behaviour function

The following function should be started on the EPTF Host Admin component:

*f\_EPTF\_HostAdmin\_behavior(pl\_selfName)*

This function automatically initializes the Host Admin component and periodically updates the measurement variables and dependent EPTF Variables.

## Host Admin Server Public Functions

### Initialization

Before using the EPTF Host Admin Server feature the

*f\_EPTF\_HostAdminServer\_init\_CT(pl\_selfName, pl\_dataSource\_compRef)*

function should be called. This initializes the EPTF Host Admin Base feature.

### Setting the Sort Order

The sort order of process statistics can be set with the following function:

*f\_EPTF\_HostAdminServer\_setSortBy(pl\_sortBy)*

### Getting Variable Name Prefix for Process Stats

The function

*f\_EPTF\_HostAdminServer\_getProcessVarNamePrefix(pl\_processSelfName)*

can be used to get a variable name prefix for process statistics by the EPTF selfName of a process.

Currently, the following constants are defined for variable name suffixes:

* c\_HostAdminServer\_dataElementHostName: the host name of the process, type: charstring.
* c\_HostAdminServer\_dataElementPID: process id, type: integer.
* c\_HostAdminServer\_dataElementCpuLoad: CPU load, type: float.
* c\_HostAdminServer\_dataElementMemUsage: memory usage in kilobytes, type: integer.
* c\_HostAdminServer\_dataElementMemUsageHRF: memory usage in human readable format, type: charstring.

For example, the Variable name for the CPU usage statistics of EPTF component with selfName “SIPLGen2” can be get with the following:

f\_EPTF\_HostAdminServer\_getProcessVarNamePrefix(“SIPLGen2”) & c\_HostAdminServer\_dataElementCpuLoad

## Summary Table of all public functions for EPTF Host Admin

Table 1. Summary of Host Admin functions

|  |  |
| --- | --- |
| Function name | Description |
| f\_EPTF\_HostAdmin\_Base\_InitMeasInterface | Initializes the Host Admin Base Component |
| f\_EPTF\_HostAdmin\_Base\_CleanupMeasInterface | Should be called before stopping the Host Admin Base component |
| f\_EPTF\_HostAdmin\_Base\_update | Updates measurement data |
| f\_EPTF\_HostAdmin\_Base\_getProcessName | Get the name of a process by PID |
| f\_EPTF\_HostAdmin\_Base\_addProcessStat | Add process statistics by PID |
| f\_EPTF\_HostAdmin\_Base\_removeProcessStat | Remove process statistics by PID |
| f\_EPTF\_HostAdmin\_Base\_removeProcessStatByIdx | Remove process statistics by index |
| f\_EPTF\_HostAdmin\_Base\_getProcessCpuLoad | Get CPU load of process by PID |
| f\_EPTF\_HostAdmin\_Base\_getProcessCpuLoadByIdx | Get CPU load of process by index |
| f\_EPTF\_HostAdmin\_Base\_getProcessMemUsage | Get memory usage of process by PID |
| f\_EPTF\_HostAdmin\_Base\_getProcessMemUsageByIdx | Get memory usage of process by index |
| f\_EPTF\_HostAdmin\_behavior | Behavior function that should be started on Host Admin component |
| f\_EPTF\_HostAdminServer\_init\_CT | Initilize Host Admin Server |
| f\_EPTF\_HostAdminServer\_setSortBy | Set the sort order for pre host process statistics |
| f\_EPTF\_HostAdminServer\_getProcessVarNamePrefix | Get Variable name prefix for process stats by selfName |

## DataSource Client functionality

The Host Admin feature provides DataSourceClient functionality. This means that a HostAdmin component has several Variables, that are available to use with the help of a DataSource server, for example the UIHandler. With the call of the function f\_EPTF\_UIHandler\_createGUI(), a custom GUI is built, and the data of the features are freely to use. For more info about the DataSource feature, see the UIHandler Function Description [7]. For the list of the existing iterators and external data elements see the natural documentation.