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

EPTF CLL Statistics Capture Control, Function Description

Contents

[1 Introduction 3](#_Toc405383969)

[1.1 Revision history 3](#_Toc405383970)

[1.2 How to Read this Document 3](#_Toc405383971)

[1.3 References 3](#_Toc405383972)

[1.4 Scope 4](#_Toc405383973)

[1.5 Recommended way of reading 4](#_Toc405383974)

[1.6 Typographical conventions 4](#_Toc405383975)

[1.7 Abbreviations 4](#_Toc405383976)

[1.8 Terminology 4](#_Toc405383977)

[1.9 Release Information 6](#_Toc405383978)

[2 General Description 6](#_Toc405383979)

[3 Functional Interface 7](#_Toc405383980)

[4 EPTF\_StatCapture\_CT Component API 7](#_Toc405383981)

[4.1 Naming Conventions 7](#_Toc405383982)

[4.2 Public Functions 7](#_Toc405383983)

[4.2.1 Initialization 7](#_Toc405383984)

[4.2.2 Creating a Logfile for a Capture Group 8](#_Toc405383985)

[4.2.3 Creating Statistics for Capture 8](#_Toc405383986)

[4.2.4 Capture Group Creation 9](#_Toc405383987)

[4.2.5 Adding Statistics to Capture Group 10](#_Toc405383988)

[4.2.6 Get/Set Functions 11](#_Toc405383989)

[4.2.7 Starting / Stopping Capture, Capture Control by API 11](#_Toc405383990)

[4.2.8 Deleting a Capture Group 12](#_Toc405383991)

[4.3 Summary Table of the public functions for EPTF Statistics Capture Control feature 12](#_Toc405383992)

[4.4 Private Functions 14](#_Toc405383993)

[4.4.1 Logging Mechanism at Sampling Intervals 14](#_Toc405383994)

[4.4.2 Cleanup function 14](#_Toc405383995)

[5 EPTF\_StatCaptureControl\_CT Component API 15](#_Toc405383996)

[5.1 Naming Conventions 15](#_Toc405383997)

[5.2 Public Functions 15](#_Toc405383998)

[5.2.1 Initialization 15](#_Toc405383999)

[5.3 Private Functions 15](#_Toc405384000)

[5.3.1 Automatic Capture Control Functionality 15](#_Toc405384001)

[5.3.2 Cleanup function 16](#_Toc405384002)

[5.4 Summary Table of the public functions for EPTF Statistics Capture Control 16](#_Toc405384003)

[6 EPTF\_StatCaptureClient\_CT Component API 17](#_Toc405384004)

[6.1 Naming Conventions 17](#_Toc405384005)

[6.2 Public Functions 17](#_Toc405384006)

[6.2.1 Initializing the client component 17](#_Toc405384007)

[6.2.2 Subscribe statistics to a group 17](#_Toc405384008)

[6.2.3 Starting / Stopping Capture from Client 17](#_Toc405384009)

[6.3 Summary Table of the public functions for EPTF Statistics Capture Client feature 18](#_Toc405384010)

[7 Capture File Format Specification 19](#_Toc405384011)

[7.1 Readable Format 19](#_Toc405384012)

[7.2 Gnuplot Format 21](#_Toc405384013)

# Introduction

## Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Rev | Characteristics | Prepared |
| 2007-11-23 | PA1 | First draft version | <EANDDAR> |
| 2007-12-06 | PA2 | Updated after revision | <EANDDAR> |
| 2008-01-02 | PA3 | The functionality is extended with a new (optional) component EPTF\_StatCapControl\_CT and belonging functionality | <EANDDAR> |
| 2010-08-24 | PB1 | CR\_TR00017469 – client-server implementation, can capture remote variable | <EJNOSVN> |
| 2010-10-25 | PC1 | HM83202 – separators in stat names substituted; stat header written after new stat(s) added | EGBOTAT |
| 2014-07-04 | PD1 | Updated section Creating a Logfile for a Capture Group | EDNISND |
| 2014-12-02 | D | Updated for release | ESZILSZ |

## How to Read this Document

This is the Function Description for the Statistics Capture Control of the Ericsson Performance Test Framework (TitanSim), Core Library (CLL). EPTF Core Library is developed for the TTCN-3 ‎[1] Toolset with TITAN ‎[2]. For more information on the EPTF Core Library, see the Product Revision Information ‎[3]. Additionally, to understand the functionality of this feature, read the documentation of the Variable ‎[6] and the Statistics Measurement ‎[7] features.

## References

1. ETSI ES 201 873-1 v3.2.1 (2007-02)  
   The Testing and Test Control Notation version 3. [Part 1: Core Language](http://www.etsi.org/deliver/etsi_es/201800_201899/20187301/03.02.01_60/es_20187301v030201p.pdf)
2. 1/198 17-CRL 113 200/4 Uen  
   User Guide for the TITAN TTCN-3 Test Executor
3. 109 21-CNL 113 512-20 Uen   
   EPTF Core Library for TTCN-3 toolset with TITAN, Product Revision Information
4. 155 17-CNL 113 512 Uen   
   EPTF Core Library for TTCN-3 toolset with TITAN, Function Specification
5. EPTF Core Library for TTCN-3 toolset with TITAN, [Reference Guide](http://ttcn.ericsson.se/TCC_Releases/Libraries/EPTF_Core_Library_CNL113512/doc/apidoc/html/index.html)
6. 18/155 16-CNL 113 512 Uen  
   EPTF Core Library Variable Function Description
7. 16/155 16-CNL 113 512 Uen  
   EPTF Core Library Statistics Measure Function Description
8. 4/155 16-CNL 113 512 Uen  
   EPTF Core Library Base Function Description

## Scope

This document is to specify the content and functionality of the Statistics Capture Control feature of the EPTF Core Library.

## Recommended way of reading

The readers are supposed to get familiar with the concept and functionalities of EPTF Core Library ‎[4] in general, and the Variable ‎[6] and Statistics Measure ‎[7] features. 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. Moreover, whenever a concept is mentioned that has a special meaning as described in the Glossary (Section ‎1.8) of this document, then these occurrences are marked with an initial arrow, for example: *🡪 TitanSim Statistics*.

## Abbreviations

CLL Core 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]

## Terminology

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

*TitanSim Variables* contain values, from which *🡪 TitanSim Statistics* can be calculated automatically on changes or periodically.

*TitanSim Statistics Measure* is a feature responsible for creating *🡪 TitanSim Statistics* and updating their value when required.

*TitanSim Statistics* are Statistics of TTCN-3 Variables. Possible Statistics are among others minimum, maximum, mean, standard deviation, etc. and also the content of a Variable can be a Statistics itself. The values of such Statistics are automatically updated in the background.

*Capture Group* is a concept for grouping Statistics references with the same *🡪 Sampling Time* together, capturing referenced Statistics at sampling intervals and printing contents of the group to the log files assigned to the group.

*Sampling Time* is defined per *🡪 Capture Group*, in seconds. It is the time period, at which the Statistics contained by a capture group are sampled and printed out to the log files.

*Capture Control Mode* is a functionality which allows choosing between several capture control modes for starting and stopping statistics capture, such as: *🡪 Manual* *🡪 Load-based* and *🡪 Complete* *Capture Control* modes.

*Manual Capture Control* is a *🡪 Capture Control Mode* that starts/stops the statistics capture according to a manual user interaction (for example a button pressed on the TitanSim GUI).

*Load-based Capture Control* is a *🡪 Capture Control Mode*, which makes it possible to start/stop capture according to load stability measured in the system. The load Variable can be any user-defined load-related parameter supported by the TitanSim tool (for example the cps generated by the tool, CPU load of the IUT, etc.).

*Complete Capture Control* is a *🡪 Capture Control Mode*, which starts statistics capture automatically when the test execution is started and automatically stops it when the test execution is finished.

## Release Information

This document contains actualized information on the TitanSim Release R4A. The new functionalities compared to the release R7A01 concerning the EPTF Statistics Capture Control feature are the following:

* The feature is extended with the client component. The existing component behaves like before and adds server functionalities.
* The changes are backward-compatible to the initial release R2A01 and tested according to this requirement.

# General Description

This document specifies the Statistics Capture Control feature of the EPTF Core Library.

The EPTF Statistics Capture Control feature makes it possible to

* Group Statistics into Capture Groups with a specified Sample Time
* Define output log files per Capture Group
* Print the value of Statistics in the Group to the log files periodically, at Sample Time intervals
* Modify Capture Group parameters
* Create/Delete Capture Group(s)
* Start capturing per group, stop capturing a group, and restart it
* Optionally: to automatically control the statistics capture according to the selected Capture Control Mode
* Capture on remote component

The statistics data is calculated by the EPTF Statistics Measure feature of the EPTF CLL based on EPTF Variable values. The EPTF CLL Statistics Measure feature ‎[7] makes it possible to create several Statistics belonging to the same EPTF Variable. These Statistics can have different types (content, delta, min, max, etc.) and also several Statistics of the same type can be defined to each Variable (for example min, delta, deltaSum).

In the EPTF Statistics Capture Control feature, for all Statistics, it can be defined, whether they should be reset periodically. The reset period is the same as the timer period of the capture group, in which the Statistics is inserted.

The aim of the EPTF Statistics Capture Control feature is to group certain Statistics into Capture Groups and to print Capture Group contents regularly at sampling intervals to predefined capture files.

Apart from controlling (starting/stopping/restarting) Statistics Capture with the functions of the API belonging to the EPTF\_StatCapture\_CT component, an optional component, the EPTF\_StatCaptureControl\_CT can be used to achieve automatic capture control. Note that this is an optional, extended functionality.

If the user application includes the EPTF\_StatCaptureControl\_CT component as well, then the starting and stopping of the Statistics Capture can be done automatically according to the selected Capture Control Mode.

To be able to use EPTF Statistics Capture, the user component should extend the EPTF\_StatCapture\_CT component. If the automatic Capture Control functionality is also needed, the user component should extend the EPTF\_StatCapture\_Control\_CT component. In order to capture statistics on a remote component, the generating components should extend the EPTF\_StatCaptureClient\_CT, the statistic logger component should extend the EPTF\_StatCapture\_CT.

# Functional Interface

Apart from this description, a cross-linked reference guide for the EPTF Core Library Functions can be reached for online reading ‎[5].

In this function description, the functions of the main functionality (EPTF\_StatCapture\_CT) and the optional, automatic capture control functionality (EPTF\_StatCaptureControl\_CT) are described separately in the following chapters.

# EPTF\_StatCapture\_CT Component API

## Naming Conventions

All functions have the prefix f\_EPTF\_StatCapture\_.

## Public Functions

### Initialization

Before using the EPTF Statistics Capture Control functions the

f\_EPTF\_StatCapture\_init(componentname)

function should be called. This initializes the EPTF Statistics Capture Control feature.

### Creating a Logfile for a Capture Group

f\_EPTF\_StatCapture\_openNewStatLogFile (userfilename)

This function should be called instead of FileIO functions to create a statistics logfile. It creates a file with the defined userfilename, appends self-name and component ID, and then the actual time and date to its filename, and immediately writes the TimeStampBase data and capture file version into the file.

The created file should have the following format:

e.g. *FirstFile\_main\_mtc\_2007-11-16\_14.00.02.txt* or *FirstFile\_ptc1\_3\_2007-11-16\_14.00.02.txt*

corresponding to the <username>\_<selfname>\_<component ID>\_<y-m-d>\_<h.m.s> format.

The selfname is the test component self-name and the component ID is the component identifier. The TimeStampBase means not only that the actual date and time is printed to the file, but also that a capture timer is started for that time. Later on, when data is printed to this capture file, it always has a relative time value compared to the TimeStampBase.

### Creating Statistics for Capture

f\_EPTF\_StatCapture\_createStatforCapture(statID, statname, periodicreset := false, activestat := true)

In order to enable Statistics settings for the capture, this function must be called. The user first creates a Variable with (EPTF\_Variable ‎[6]), then creates a Statistics for this Variable (with EPTF\_StatMeasure ‎[7]). Then, to define capture data for this Statistics, the createStatforCapture function should be called.

The *statID* is the ID returned by StatMeasure. It is important, that it is not allowed to create more Statistics for capture with the same *statID*. The *statname* can be set by the user. The last two parameters are optional, by default, the created Statistics is not reset on Sampling Time intervals (*periodicreset*), and they are set to active (*activestat*), i.e. their values are printed out to the capture files.

If the function is called with more than two parameters (statID, statname, true, false), the reset functionality and the statistics state can be changed. If the Statistics should be reset periodically after capture, then the *periodicreset* parameter should be set to true. This means, that after having added this Statistics to a capture group (or several capture groups), its value is reset at sampling intervals of the capture group.

**Note:** that these Statistics with periodicreset set to true can be added to *one and only one* capture group.

### Capture Group Creation

#### f\_EPTF\_StatCapture\_addNewGroup(groupname, timerperiod,logfiles, statisticsList,

groupidx, groupstatidxList)

This function creates a capture group with a unique group name; a sampling period (*timerperiod*) and a list of *logfiles* are assigned as output logfiles for the capture group. If no logfiles are passed to this function, the capture group is printed to the log file specified by the Module Parameter **tsp\_EPTF\_StatCapture\_def\_capture\_file** in the config file.

Additionally, if the *statisticsList* parameter is set (by default it is empty), the function adds a list of statistics to the capture group. In this case, the created statistics indices are also returned.

**Note**: there is no restriction for adding one statistics to several capture groups if it is a Statistics with periodic reset set to true. Also, logging several capture groups into the same logfile is possible.

The index of the created group is returned in the *groupidx* argument. This index can be used later to access this capture group. Any EPTF Capture Group is uniquely identified by the component reference where it is created and by this index.

#### f\_EPTF\_StatCapture\_parseConfigGroup()

Apart from calling API functions, it is also possible to create Capture Groups from the configuration file with module parameters. Then, in the demo application, this function should be called after the local Variables (which we would like to capture) are created. This function then reads in the configuration parameters and creates Capture Groups with the defined parameters (for example groupname, timer period, Statistics.) **Note:** Statistics are created on-the-fly from the configuration file according to Variable Name (on the demo component, local Variable which we would like to capture), Statistics type and Name, for example:

varName := "var1",

statType := delta, //content, delta, deltaSum, min, max,

// mean, standardDev statName := "MyDelta"

**Note:** As a best practice, it is advised not to use both options (API capture group creation and group creation from configuration file) at the same time in order to avoid conflicts such as group name or variable name collision.

### Adding Statistics to Capture Group

A capture group consists of a unique group name, a sampling period, a list of file descriptors to print the group and a list of Statistics (references to Statistics) each containing statistics content or statistical data.

Statistics can be either active or inactive in the capture group. By default, they are active, that means their content is printed out to the log files belonging to the capture group. When switched to inactive, a “-“ sign is printed to the log files instead of their value. Activity state of Statistics can be modified during run-time.

The reset of Statistics is carried out at each sampling interval of the capture group after they are captured for all Statistics, which have been created with the *periodicreset* field set to true.

Adding one or more statistics triggers the writing of a statistics header line on the next write of the statistics to the capture file.

Statistics can be added to the group one by one or as a list.

#### Adding a single Statistics:

f\_EPTF\_StatCapture\_addStatistics(groupindex, stat, groupstatindex)

The input parameters are the *groupindex*, i.e. the unique identifier of the capture group where we would like to add the new Statistics, and the Statistics descriptor *stat*, which contains the Statistics reference, its name, the active flag, which is set to true by default and the periodicreset flag, which is set to false by default. On successful insertion, the index of the Statistics inside the capture group is returned in *groupstatindex*.

#### Adding a list of Statistics:

f\_EPTF\_StatCapture\_addListOfStatistics(groupindex,  
 statList, groupstatidxList)

A batch alternative of the previous function. The identifier of the capture group, and the list of Statistics descriptors *statList* are the input parameters, and a list of created Statistics indexes inside the capture group *groupstatidxList* is returned.

### Get/Set Functions

#### Setting/Modifying the Sampling Time of a Capture Group:

f \_EPTF\_StatCapture\_setTimerPeriod(groupindex, timerperiod)

The sampling time period of the capture group specified by the *groupindex* parameter can be set during run-time to the float *timerperiod* input parameter (expressed in seconds). Every time the timer period is changed successfully, the new timer period is printed to the log files assigned to the capture group. From then on, the active Statistics of the capture group is captured and printed to file at these intervals.

#### Modifying the capture state of a Statistics

f\_EPTF\_StatCapture\_InactivateStatistics(groupindex, groupstatidx)

f\_EPTF\_StatCapture\_ActivateStatistics(groupindex, groupstatidx)

These functions set the capture state of a Statistics specified by its index *groupstatidx* and the capture group *groupindex* it belongs to. By default, the state of the Statistics is active, that means, at each sampling period, when the capture group is captured, the content of the Statistics is printed out to the log files belonging to the capture group. During run-time, the capture state of the Statistics can be changed. If the state is set to inactive, instead of printing out Statistics content, a “-“ sign is printed out to the log files.

**Note:** if this Statistics is included in other capture groups, then its state can be set independently for each capture group.

#### Finding a Statistics in a capture group

f\_EPTF\_StatCapture\_getStatisticsID (stat, groupindex, groupstatidx) returns boolean

This function returns the Statistics index of the input parameter *stat* inside the Capture group specified by the input parameter *groupindex*.

If the Statistics is not contained in the capture group, false is returned.

### Starting / Stopping Capture, Capture Control by API

Starting and stopping capture can be done anytime during runtime. Only deleted capture groups cannot be restarted again.

#### Start capturing a capture group

f\_EPTF\_StatCapture\_startGroupCapture(groupindex)

This function starts capturing a capture group defined by *groupindex*. First, it checks whether the group is not deleted, then prints the actual sampling time, groupname and Statistics value headers for the group to each logfile specified at capture group creation. At the end of each sampling interval, the contents of the capture group Statistics are printed out, until the capturing is stopped.

#### Start capturing all capture groups

f\_EPTF\_StatCapture\_startCaptureAll()

This function starts capturing all (not deleted) capture groups. It calls the startGroupCapture function for each capture group.

#### Stop capturing a capture group

f\_EPTF\_StatCapture\_stopGroupCapture(groupindex)

This functions stops capturing a capture group defined by groupindex. As an effect, the content of the capture group is not printed out to the logfiles.

#### Stop capturing all capture groups

f\_EPTF\_StatCapture\_stopCaptureAll()

This function stops the capture of all capture groups.

### Deleting a Capture Group

f\_EPTF\_StatCapture\_deleteGroup(groupindex)

Capture groups can be deleted during run-time. This means, that their content is not printed to file any more. Note that field contents of the group are deleted and the group cannot be restarted again.

## Summary Table of the public functions for EPTF Statistics Capture Control feature

Table 1. Summary of Statistics Capture Control functions belonging to the EPTF\_StatCapture\_CT component.

|  |  |
| --- | --- |
| Function name | Description |
| f\_EPTF\_StatCapture\_init\_CT | Initializes the StatCapture component. |
| f\_EPTF\_StatCapture\_openNewStatLogFile | Opens a new statistics logfile with the actual time and date appended to the filename and prints the TimeStampBase value into the file. |
| f\_EPTF\_StatCapture\_createStatforCapture | Creates Statistics for Capture. |
| f\_EPTF\_StatCapture\_addNewGroup | Adds a new Capture Group. |
| f\_EPTF\_StatCapture\_parseConfigGroup() | Creates Statistics and adds new Capture Groups from parameters specified in the configuration file. |
| f\_EPTF\_StatCapture\_addStatistics | Adds Statistics to the Capture Group. |
| f\_EPTF\_StatCapture\_addListOfStatistics | Adds a list of Statistics to the Capture Group. |
| f \_EPTF\_StatCapture\_setTimerPeriod | Sets the Timer Period (sampling time) of the Capture Group. |
| f\_EPTF\_StatCapture\_setLogFiles | Sets capture files belonging to the Capture Group. |
| f\_EPTF\_StatCapture\_InactivateStatistics | Inactivates Statistics in a given Capture Group. |
| f\_EPTF\_StatCapture\_ActivateStatistics | Activates Statistics in a given Capture Group. |
| f\_EPTF\_StatCapture\_getStatisticsID | Looks up a Statistics in a Capture Group according to the Statistics ID. |
| f\_EPTF\_StatCapture\_startGroupCapture | Starts the Capture of a Capture Group. |
| f\_EPTF\_StatCapture\_startCaptureAll | Start Capturing all Capture Groups. |
| f\_EPTF\_StatCapture\_stopGroupCapture | Stops the Capture of a Capture Group. |
| f\_EPTF\_StatCapture\_stopCaptureAll | Stops Capturing all Capture Groups. |
| f\_EPTF\_StatCapture\_deleteGroup | Deletes a Capture Group. |

## Private Functions

These functions *should not be called* by the user of the EPTF Statistics Capture feature.

### Logging Mechanism at Sampling Intervals

#### f\_EPTF\_StatCapture\_handleEvent(actionID, eventindex)

is a private function, it serves for logging the content of the capture group to the log files belonging to the capture group at sampling intervals. Inactive Statistics are printed out with “-“.

**Note:** this function should not be called by the user, the mechanism is automatic.

### Cleanup function

#### f\_EPTF\_StatCapture\_cleanup()

This function is responsible for a peaceful cleanup after termination of the program module or in case of errors.

**Note:** this is a private function; it should not be called by the user. On the other hand, it is important that the user application should *register this cleanup function* in its init function with

f\_EPTF\_Base\_RegisterCleanup(refers(f\_EPTF\_StatCapture\_cleanup())

and make sure that you call the f\_EPTF\_Base\_cleanup\_CT() function (not the private StatCapture\_cleanup function!) before terminating the component execution. See more details on the usage of the EPTF Base functions in its Function Description ‎[8].

**Note:** other private functions not mentioned in the documentation can be found in the source code after the //Private Functions comment. These functions should not be called by the user.

# EPTF\_StatCaptureControl\_CT Component API

## Naming Conventions

All functions have the prefix f\_EPTF\_StatCaptureControl\_.

## Public Functions

### Initialization

In order to use the (optional) EPTF Statistics Capture Control functionality the

f\_EPTF\_StatCaptureControl\_init(componentname, nameserverRef,

autoControlEnabled)

function should be called. The EPTF\_StatCaptureControl\_CT component extends the EPTF\_NS\_Client\_CT Name Server Client component to use the Name Service functionality. Therefore, in the *nameserverRef* parameter, the component reference to the belonging Name Service Server Component is needed. This initializes the EPTF Statistics Capture Control component, processes the configuration parameters (for example capture control mode), registers component references to the Name Service Server and adds PostProc functions (see ‎5.3.1) for capture control. These PostProc functions automatically control the statistics capture according to the changes of the capture control Variables.

In case the function is called with the optional *autoControlEnabled* parameter set to false, the component behaves exaclty like the EPTF\_StatCapture\_CT component (see Section ‎4). (Automatic capture control is switched off). By default, this parameter is set to true.

## Private Functions

These functions *should not be called* by the user of the EPTF Statistics Capture Control feature.

### Automatic Capture Control Functionality

The capture control functionality is implemented with PostProc functions belonging to the control Variables of the Capture Control Modes respectively.

f\_EPTF\_StatCaptureControl\_User\_PostProc (idx, argList)

f\_EPTF\_StatCaptureControl\_Execution\_PostProc (idx, argList)

f\_EPTF\_StatCaptureControl\_Load\_PostProc (idx, argList)

If certain guard conditions are fulfilled, they call the statistics starting/stopping functions (f\_EPTF\_StatCapture\_startCaptureAll(), f\_EPTF\_StatCapture\_stopCaptureAll()) of the EPTF\_StatCapture\_CT API.

### Cleanup function

f\_EPTF\_StatCaptureControl\_cleanup()

This function is responsible for a peaceful cleanup after termination of the program module or in case of errors.

**Note:** this is a private function; it should not be called by the user. On the other hand, it is important that the user application should *register this cleanup function* in its init function with

f\_EPTF\_Base\_RegisterCleanup(refers(f\_EPTF\_StatCaptureControl  
 \_cleanup())

and make sure that you call the f\_EPTF\_Base\_cleanup\_CT() function (not the private StatCaptureControl\_cleanup function!) before terminating the component execution. See more details on the usage of the EPTF Base functions in its Function Description ‎[8].

**Note:** other private functions not mentioned in the documentation can be found in the source code after the //Private Functions comment. These functions should not be called by the user.

## Summary Table of the public functions for EPTF Statistics Capture Control

Table 1. Summary of Statistics Capture Control functions belonging to the EPTF\_StatCaptureControl\_CT component

|  |  |
| --- | --- |
| Function name | Description |
| f\_EPTF\_StatCaptureControl\_init\_CT | Initializes the StatCaptureControl component and activates event handling. |

# EPTF\_StatCaptureClient\_CT Component API

## Naming Conventions

All functions have the prefix f\_EPTF\_StatCaptureClient\_.

## Public Functions

### Initializing the client component

If the statistic generating and the logging should be on another component, the generating side should extend the StatCaptureClient\_CT feature component. Before using the EPTF Statistics Capture Client functions the client should call the

f\_EPTF\_StatCaptureClient\_init(componentname)

function. This initializes the EPTF Statistics Capture Client feature.

### Subscribe statistics to a group

In order to access the statistics on the client component, the client should subscribe the statistics on the server grouped by a groupName with the function   
f\_EPTF\_StatCaptureClient\_addStatsToGroup( pl\_groupName, pl\_stats, pl\_subscriptionMode).   
The subscription mode is defined by the pl\_subscriptionMode.

### Starting / Stopping Capture from Client

Starting and stopping capture can be done anytime during runtime.

#### Start capturing a capture group

f\_EPTF\_StatCaptureClient\_startGroup(pl\_groupName)

This function starts capturing a capture group defined by pl\_groupName.

#### Start capturing all capture groups

f\_EPTF\_StatCaptureClient\_startGroupAll()

This function starts capturing all (not deleted) capture groups.

#### Stop capturing a capture group

f\_EPTF\_StatCaptureClient\_stopGroup(pl\_groupName)

This function stops capturing a capture group defined by pl\_groupName. As an effect, the content of the capture group is not printed out to the logfiles.

#### Stop capturing all capture groups

f\_EPTF\_StatCaptureClient\_stopGroupAll()

This function stops the capture of all capture groups on the server.

#### Delete capturing a capture group

f\_EPTF\_StatCaptureClient\_deleteGroup(pl\_groupName)

This function deletes a capture group defined by pl\_groupName. As an effect, the content of the capture group is not printed out to the logfiles and cannot be started again until the f\_EPTF\_StatCaptureClient\_addStatsToGroup() function is called to that group.

## Summary Table of the public functions for EPTF Statistics Capture Client feature

Table 2. Summary of Statistics Capture Control functions belonging to the EPTF\_StatCaptureClient\_CT component

|  |  |
| --- | --- |
| Function name | Description |
| f\_EPTF\_StatCaptureClient\_init\_CT | Initializes the StatCaptureClient component. |
| f\_EPTF\_StatCaptureClient\_addStatsToGroup | Add statistics to a Capture Group on the server. |
| f\_EPTF\_StatCaptureClient\_startGroup | Starts the Capture of a Capture Group on the server. |
| f\_EPTF\_StatCaptureClient\_startGroupAll | Start Capturing all Capture Groups on the server. |
| f\_EPTF\_StatCaptureClient\_stopGroup | Stops the Capture of a Capture Group on the server. |
| f\_EPTF\_StatCaptureClient\_stopGroupAll | Stops Capturing all Capture Groups on the server. |
| f\_EPTF\_StatCaptureClient\_deleteGroup | Deletes a Capture Group on the server. |

# Capture File Format Specification

Using the TCCFile\_IO functionality, the Capture Control feature defines two different file formats (the user can select the required file format from config file). The default file format (*readable*) is easy to process, filter, sort, etc. with standard POSIX text processing utilities (grep, sort, awk, sed. etc.). The second file format (*gnuplot*) is compatible with the input format of Gnuplot, a simple command-line data plotting utility.

For efficiency reasons statistics shall be sampled, that is they shall be dumped to the file at regular intervals only.

In the gnuplot format, output files can contain comment lines. A comment line must start with a “#” character.

The actual version of the capture file is printed out in a header line (for example #Capture File Version: 2.2).

## Readable Format

In the default file format, each line has the following general syntax:

1. Empty lines are ignored. Whitespaces between fields are ignored.
2. Non-empty lines start with a heading declaring the type of the line. The heading is separated by a colon from the fields.
3. The heading might be structured, this is described at each heading
4. String values are to be given as TTCN-3 charstring literals surrounded by double quotes

The following line types are defined:

|  |  |
| --- | --- |
| **Line heading** | **Line fields and meaning** |
| TimeStampBase | The actual snapshot time when the dumping has been started |
| CaptureFileVersion: x.x | The actual version of the capture file format. |
| CaptureGroup[“GroupName“] | Defines a CaptureGroup with the given GroupName. It also means that the capture of this CaptureGroup has been started. |
| SampleTime[“GroupName“] | <aa.bbbbbb>: Sampling time in seconds |
| ValueHeader[“GroupName“] | Each field defines the identifier of the respective variable in the corresponding CaptureGroup line |
| [“GroupName“, Relative Time] | Each field defines the value of the respective variable in the corresponding ValueHeader line.  The Relative Time represents the time passed since the TimeStampBase value. |
| Capture\_Started [“GroupName”, ID,  user/execution/load-based /programmed] | This text is printed out in all Capture Control Modes (in case of the programmed API control as well). It signals that the capture measurement period is started for the given Capture Group and an ID is assigned to the measurement period. Eg. the ID 2\_12 signifies the 12th measurement period belonging to the Capture Group with ID 2. This ID identifies the measurement period when it is stopped. |
| Capture\_Finished ["GroupName", ID]  Capture\_Aborted  [“GroupName”, ID] | This text is printed out when the capture measurement period is stopped. It is noted, whether the measurement period (with belonging Capture Group Name and measurement period ID) is finished with success or it is aborted before its expected duration. |

Example:

TimeStampBase 2007-11-23-15:52:06.712812 0.578000

CaptureFileVersion: 2.2  
CaptureGroup["Stat1"]

Capture\_Started[“Stat1”, 1\_1, programmed]

SampleTime["Stat1"]: 1.000000 sec

ValueHeader["Stat1"]: TestCaseId respRecOK respRecNOK respTimeout

["Stat1",0.578000]: “User” 1 0 0.005000

["Stat1",1.578000]: “130.100.72.230” 1 0 0.020000

Capture\_Finished[“Stat1”, 1\_1]

CaptureGroup[“ErrorCodes“]

Capture\_Started[“ErrorCodes”, 2\_1, programmed]

SampleTime[“ErrorCodes“]: 2.000000 sec  
ValueHeader[“ErrorCodes“]: TestCaseId respCode numReceived description

[“ErrorCodes“, 13.650000]: “User” 503 5 “User does not exist”

SampleTime[“Stat1“]: 3.000000 sec

## Gnuplot Format

In the Gnuplot format, all headers are commented out with the “#” character. The line containing groupname, relative time and actual variable values is changed to the following format:

GroupName RelativeTime value1 value2 … valueN.

String values and the GroupName are charstring literals, surrounded by double quotes.

That means, all non-commented lines are of this format, the separator between fields is the whitespace character. Thus, the file can be easily processed by Gnuplot. An example for this format can be the following:

# TimeStampBase 2007-11-23-15:52:06.712812 0.578000

# Capture File Version: 2.2  
# CaptureGroup["Stat1"]

# Capture\_Started[“Stat1”, 1\_1, programmed]

# SampleTime["Stat1"]: 1.000000 sec

# ValueHeader["Stat1"]:TestCaseId respRecOK respRecNOK respTimeout

“Stat1” 0.578000 “User” 1 0 0.005000

“Stat1” 1.578000 “130.100.72.230” 1 0 0.020000

# Capture\_Finished[“Stat1”, 1\_1]

# CaptureGroup[“ErrorCodes“]

# Capture\_Started[“ErrorCodes”, 2\_1, programmed]

# SampleTime[“ErrorCodes“]: 2.000000 sec  
# ValueHeader[“ErrorCodes“]: TestCaseId respCode numReceived description

“ErrorCodes” 13.650000 “User” 503 5 “User does not exist”

# SampleTime[“Stat1“]: 3.000000 sec