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

EPTF CLL Statistics Measure, User Guide

Contents

[Introduction 2](#_Toc327539307)

[1.1 Revision history 2](#_Toc327539308)

[1.2 About this Document 2](#_Toc327539309)

[1.2.1 How to Read this Document 2](#_Toc327539310)

[1.2.2 References 2](#_Toc327539311)

[1.2.3 Abbreviations 3](#_Toc327539312)

[1.2.4 Terminology 3](#_Toc327539313)

[1.3 System Requirements 3](#_Toc327539314)

[2 Statistics Measure 3](#_Toc327539315)

[2.1 Overview 3](#_Toc327539316)

[2.2 Description of files in this feature 3](#_Toc327539317)

[2.3 Description of required files from other features 4](#_Toc327539318)

[2.4 Installation 4](#_Toc327539319)

[2.5 Configuration 5](#_Toc327539320)

[3 Usage 5](#_Toc327539321)

[3.1 Special functions 6](#_Toc327539322)

[3.1.1 Delta, DeltaSum, Min and Max 6](#_Toc327539323)

[3.1.2 Mean 6](#_Toc327539324)

[3.1.3 StandardDev 6](#_Toc327539325)

[3.1.4 EPS and Chrono 6](#_Toc327539326)

[3.1.5 Density 7](#_Toc327539327)

[3.1.6 Percentile95 7](#_Toc327539328)

[3.1.7 PercentileP 8](#_Toc327539329)

[3.1.8 Limits 8](#_Toc327539330)

[4 Error messages 8](#_Toc327539331)

[5 Warning messages 9](#_Toc327539332)

[6 Examples 9](#_Toc327539333)

[6.1 Configuration file 9](#_Toc327539334)

[6.2 Demo Module 9](#_Toc327539335)

# Introduction

## Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Rev | Characteristics | Prepared |
| 2007-11-29 | PA1 | First draft version | <EANDDAR> |
| 2008-01-09 | PA2 | Updated | ETHJGI |
| 2008-01-28 | PA3 | Updated after review | ETHJGI |
| 2008-12-18 | PB1 | Updated for new statistics | EAKOPER |
| 2009-03-24 | PB2 | Chrono statistics update: stop | ETHJGI |
| 2009-05-19 | PB3 | PercentileP statistics added | EBENMOL |
| 2012-05-15 | PC1 | Limits statistics added | ETHJGI |

## About this Document

### How to Read this Document

This is the User Guide for the Statistics Measure 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 Statistics Measure 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. Additionally, to understand the functionality of this feature, read the documentation of the Variable ‎[8] feature.

### 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. 16/155 16-CNL 113 512  
   EPTF CLL Statistics Measure Function Description
7. TitanSim CLL for TTCN-3 toolset with TITAN, Reference Guide  
   <http://ttcn.ericsson.se/products/libraries.shtml>
8. 18/198 16-CNL 113 512 Uen  
   EPTF CLL Variable User Guide

### 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]

### 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

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

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

*TitanSim Statistics* are TTCN-3 variables. The values of such variables are automatically and periodically updated in the background.

## System Requirements

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

# Statistics Measure

## Overview

The EPTF CLL Statistics Measure component is a fundamental component providing an implementation for statistics measurement in a load test environment.

## Description of files in this feature

The EPTF CLL Statistics Measure API includes the following files:

* StatMeasure
  + EPTF\_CLL\_StatMeasure\_Definitions.ttcn: This TTCN-3 module contains common type definitions that should be used in all Statistics Measuring Components.
  + EPTF\_CLL\_StatMeasure\_Functions.ttcn: This TTCN-3 module contains the implementation of Statistics Measure functions.

## Description of required files from other features

The Statistics Measure 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
* Variable

## Installation

Since EPTF CLL Statistics Measure 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\_StatMeasure:

* 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 Statistics Measure demo or write your own application using StatMeasure.
* 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].

The Statistics Measure 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 Statistics Measure–specific module parameters.

# Usage

To create new Statistics one of the newStat functions should be called. Stand-alone statistics can be created by specifying -1 for the variable ID.

When the Statistics is created it is reset and enabled automatically. New data can be added by the addData functions right after the statistics was created. For Statistics created for an existing EPTF Variable data is added automatically each time the value of the EPTF Variable is refreshed.

For the EPS and chrono stats there is a special function to start the measurement. It is called startMeasurement. If it is not called the measurement is started when the first data is added by addData.

The addData functions update the value of the Statistics for most of the stats, but for EPS and chrono the update function should be called explicitly.

The value of the Statistics can be accessed by the getStat functions.

If a Statistics is disabled by calling its disable function the addData functions will not add new data to the Statistics. The Statistics can be enabled by calling the enable functions.

New measurement can be started by calling the reset function and then startMeasurement or addData.

From each Statistics it is possible to create an EPTF Variable. This Variable is refreshed automatically when the value of the corresponding Statistics changes. The EPTF Variables can be created by the function f\_EPTF\_StatMeasure\_createVarFromStat.

If you call the update function of the Statistics from the postProc function of the created Variable, disable the refresh of the EPTF Variable to avoid infinite loops! For the same reasons do not call addData from postProc function!

## Special functions

Some Statistics have additional functions. Please see the Function Description ‎[6] also for more information.

### Delta, DeltaSum, Min and Max

The function f\_EPTF\_StatMeasure\_changeToFloat\_\* can be used to change the type of these statistics to float in case they are standalone stats. The type of these standalone statistics is set to integer by default. This function has to be called before a variable is created from the statistics by the f\_EPTF\_StatMeasure\_createVarFromStat function. After a variable was created form the statistics, or a statistics was created from a variable (i.e. not standalone), this function cannot be called!

### Mean

If EPTF Variable that contain the auxiliary data N (i.e. the number of data samples) of the mean statistics is needed, it can be created by the function f\_EPTF\_StatMeasure\_createVarFromStat\_N\_mean.

### StandardDev

The standard deviation statistics measures the mean also. It can be accessed by the f\_EPTF\_StatMeasure\_getStat\_standardDev\_mean function. Also the index of the mean statistics can be accessed by f\_EPTF\_StatMeasure\_getStatIdx\_standardDev\_mean.

The functions f\_EPTF\_StatMeasure\_createVarFromStat\_N\_standardDev, f\_EPTF\_StatMeasure\_createVarFromStat\_S\_standardDev and f\_EPTF\_StatMeasure\_createVarFromStat\_Mean\_standardDev can be used to create EPTF Variables from auxiliary data of the standardDev statistics. These functions correspond to the number of data samples, sum of the deviation squares and the mean of the data samples respectively.

### EPS and Chrono

The EPS statistics can also measure the time. The functions f\_EPTF\_StatMeasure\_getMeasurementLength\_EPS and f\_EPTF\_StatMeasure\_getTime\_EPS return the length of the measurement and the time since measurement was started respectively in seconds. The value of the EPS Statistics is only updated when the f\_EPTF\_StatMeasure\_update\_EPS function is called. The same functions exist for chrono Statistics also.

The chrono statistics has one additional function which can be used to stop the chronometer: f\_EPTF\_StatMeasure\_stop\_chrono. This function updates the statistics. No more update takes place after this function call until the statistics is reset.

If an EPTF Variable is created from an EPS or chrono Statistics and it has a sampledAtSync subscriber then the Statistics can be updated automatically at the refresh event synchronously in the following way:

1. Register a postProc function for the Variable that calls update\_EPS for the statistics with refresh disabled
2. Reset the statistics and start measurement after update in the postProc function if the Statistics should be reset after readout.

In this way the value of the statistics is updated in-sync with the sync event of the EPTF Variables.

Other Statistics are updated each time a new data is added, so the Variable created from them will also be updated.

The number of events can be retrieved from the EPS Statistics if the update function is called with 1.0 sec set to the length of the measurement interval.

### Density

The density statistics measures the distribution of the data values. It counts how many data falls between the user-given boundaries of values. The boundaries can be set by the function f\_EPTF\_StatMeasure\_setBoundaries\_density. This function automatically resets the statistics. Also the boundaries can be generated automatically between a minimal and a maximal value on a linear or a logarithmic scale by the function f\_EPTF\_StatMeasure\_setScale\_density. This function also resets the statistics. The function f\_EPTF\_StatMeasure\_getStat\_density returns the counters belonging to the intervals, whereas the function f\_EPTF\_StatMeasure\_getStat\_normalized\_density returns the normalized distribution function, i.e. when the sum of all elements is normalized to 1.0.

Note, that the type of the value of this statistics is an EPTF\_IntegerList. The 0th element of the list measures the number of elements that fall below the lowest boundary; the 1st counts the elements between the lowest boundary and the next boundary above it, and so on. The last element is for all data that falls above the highest boundary. The lower boundary in an interval belongs to the interval, i.e. if the value of the data equals the lower boundary it is considered to fall into the interval, while the upper boundary itself does not belong to the interval.

### Percentile95

The percentile95 statistics is based on density statistics so it also counts how many data falls between the user-given boundaries of values. The boundaries can be set by the function f\_EPTF\_StatMeasure\_setBoundaries\_percentile95. This function automatically resets the statistics. Also the boundaries can be generated automatically between a minimal and a maximal value on a linear or a logarithmic scale by the function f\_EPTF\_StatMeasure\_setScale\_ percentile95. This function also resets the statistics.

### PercentileP

The percentileP statistics is based on density statistics as percentile95, so it also counts how many data falls between the user-given boundaries of values. The boundaries can be set by the function f\_EPTF\_StatMeasure\_setBoundaries\_percentileP. This function automatically resets the statistics. Also the boundaries can be generated automatically between a minimal and a maximal value on a linear or a logarithmic scale by the function f\_EPTF\_StatMeasure\_setScale\_ percentileP. This function also resets the statistics. The p value can be set with the function f\_EPTF\_StatMeasure\_setValueP\_percentileP, which resets the statistics too.

### Limits

The limits statistics can be used to show a StatusLED with a colour that depends on the last measured data. The last data value is compared to preset limits, and the colour that corresponds to the highest limit value below the data will be used to set the colour of the LED. The text field of the LED is set to the latest value of the data if enabled. Otherwise it is set to empty string.

The limiting values can be defined to set the lower margins of the LED colours. Also the default colour of the LED is customizable. When new limits are set by the function f\_EPTF\_StatMeasure\_setLimits\_limits, the statistics is reset automatically. The limit values can be requested from the statistics by the function f\_EPTF\_StatMeasure\_getLimits\_limits.

When the reference value is defined, the statistics is calculated as the LED colour with highest limit margin below the data difference in percentage from the reference value. For example when the last measured data value is 6.0 and the reference value is 4.0, the difference in percentage is: 100.0\*(6.0-4.0)/4.0= 50%. The LED colour with biggest limit that is smaller than 50 will be used for the colour of the statistics LED, whereas the text will be set to “6.0”.

# 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.

* Invlid scale boundaries specified for logarithmic scale: min: <minVal>, max: <maxVal>

Boundaries defined for density statistic are invalid.

* <f\_EPTF\_StatMeasure\_getStat\_custom> function for custom statistics (<statNem>) is undefined!

The given function must be defined if it is intended to be used.

# 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.

* Variable for the value of stat <statIdx> (<statContent>) already exists: <varIdx>:

Create variable from statistics function has been called to a statistic from which a variable has already been created.

* It is not allowed to create stand-alone content statistics!

Content statistic can’t be created as stand alone statistic.

# Examples

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

* main.cfg
* EPTF\_Stat\_Measure\_test.ttcn

## Configuration file

The used configuration file (main.cfg) is for the Statistics Measure example is placed in the demo directory.

## Demo Module

The demo module EPTF\_StatMeasure\_Test.ttcn illustrates a typical usage of the Statistics Measure feature. It is placed in the demo directory of the released feature.