

# First Steps User's Guide

## TTplugin-TTmex

This user's guide helps you to download and install TTworkbench, and guides you through configuring and running the built-in example in TTmex plugin.

1. Requirements
2. Download
3. Installation
4. Configuration
5. Running Test Case
6. Details
7. Appendix

Testing  
Technologies



# For a successful proceeding please follow the instructions step by step!

If you need any assistance or want to report bug and error please contact Testing Technologies' customer care department:

Mr. Dirk Borowski

Phone: +49 30 726 19 19 0

Email: [ttcn3-support@testingtech.com](mailto:ttcn3-support@testingtech.com)

## 1. Requirements

Operating Systems: Microsoft Windows XP, Vista, 7, 8 (x86-32, x86-64)  
Red Hat Enterprise Linux (GTK, x86-32, x86-64)  
Fedora (GTK, x86-32, x86-64)  
SUSE Linux (GTK, x86-32, x86-64)

Java 2 Platform (JRE): Version J2SE 6.0 (1.6.x)  
Download at [www.oracle.com/technetwork/java/javase/downloads/index.html](http://www.oracle.com/technetwork/java/javase/downloads/index.html).  
Please note that we **strongly** recommend to use the above Java JDK. With the OpenJDK/IcedTea for Linux the TTworkbench license **will not work correctly**.

Memory: 2 GB (4 GB recommended)

## Reference ID and License File

Before you download TTworkbench and TTmex plugin, make sure you received a valid Reference ID and license file. Otherwise please contact our sales team at [sales@testingtech.com](mailto:sales@testingtech.com).

## 2. Download

Please use Testing Technologies' download portal: [www.testingtech.com/support/downloads.php](http://www.testingtech.com/support/downloads.php).

### Step 1

Select the latest version of TTworkbench (Express, Basic or Professional) for your platform.

### Step 2

Enter your **Download Reference ID**.

### Step 3

Download the file and save it in your favored directory.

**NOTE!** With the Linux version of Microsoft Internet Explorer, the browser saves the **.jar file** as .zip. Just rename it back to .jar.

### Step 4

Repeat steps 1-3 to download TTmex Plugin (**TTplugin-TTmex\_x.x.x\_archived-site.zip**).

### Step 5

Save the license file **license.dat** in your favored directory.

### 3. Installation

#### 3.1. TTworkbench

##### Step 1

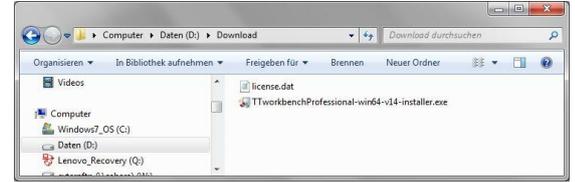
Windows Platform:

Double click on **TTworkbench-xxx-installer.exe** (to be found on desktop or selected directory).

Linux Platform:

Use command line

**java-jar TTworkbench-xxx-installer.jar**

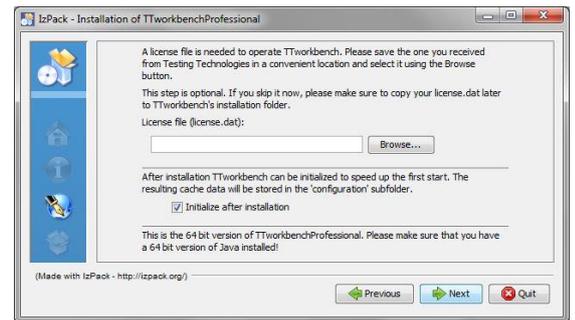


##### Step 2

Follow the pop up installation wizard...

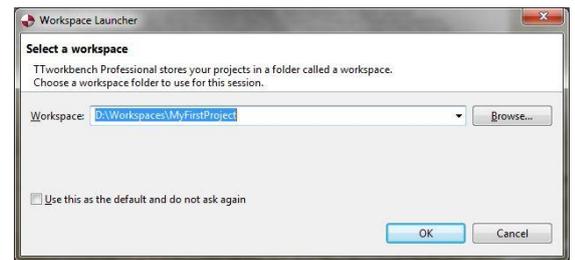
On request, browse for the valid license file **license.dat**, already saved in your favored directory.

(TTworkbench requires a **valid license file** for execution, which was sent to you by mail.)



##### Step 3

Start TTworkbench from created desktop icon or menu entry.



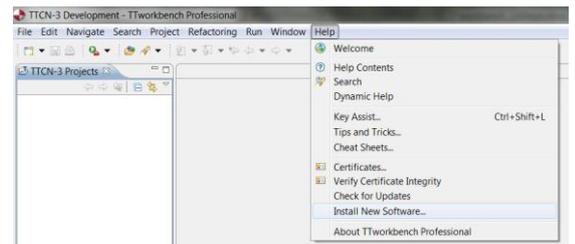
##### Step 4

Start a new workspace by accepting the default workspace location on request or choose an existing one.

#### 3.2. TTplugin – TTmex

##### Step 5

In the TTworkbench menu, click on menu item *Help* → *Install New Software...*

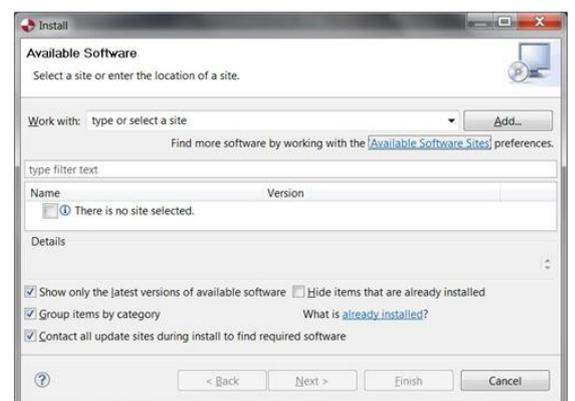


##### Step 6

Click *Add...* → *Archive*.

Choose the downloaded installation file **TTplugin-TTmex\_x.x.x\_archived-site.zip**, check the TTworkbench features box, click *Next* → *Next*.

Accept the terms of license agreements, finish and confirm restart of TTworkbench.



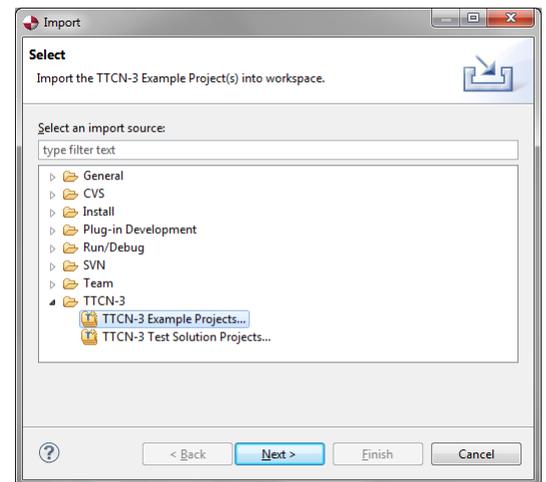
### 3.3. Import built-in example

#### Step 7

After restart you can import the built-in example project.

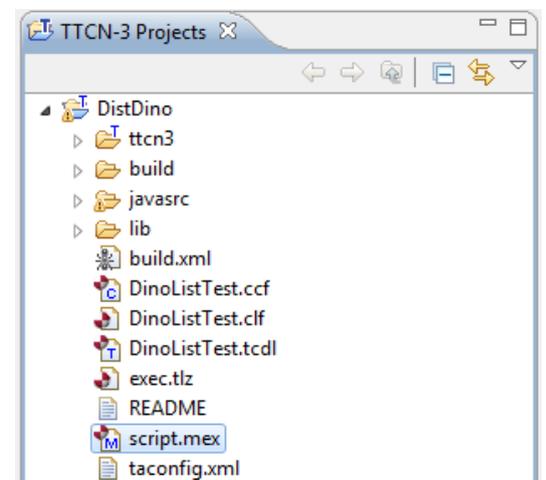
Close the welcome tab. In the project view, just click on the right mouse button, go to menu:

*File* → *Import* → *TTCN-3* → *TTCN-3 example projects*  
→ *DistDino Example* → *Finish*.



#### 3.3.1. Example project structure

ttn3	The source code of TTCN-3 definition.
Build	The Java .class files
Javasrc	The Java source code, here the codec example implementation
Lib	additional Jar libraries
build.xml	antbuild file for jar packaging
*.ccf, *.tcdl	Part of TTmex configuration, belong to .mex
*.clf	Test campaign loader file for standalone test execution
*.tlz	Test logging file
README	The configuration and usage instruction
*.mex	<b>The configuration and execution script file</b>
taconfig.xml	The test adaptation configuration



 **.mex** ist the key configuration and execution file to distributed testing.

## 4. Configuration

### 4.1. Test session deployment in .mex file

#### Step 8

Double click \*.mex file. You will see several tabs opened in the language editor.

#### 4.1.1. script.mex

add the following line:

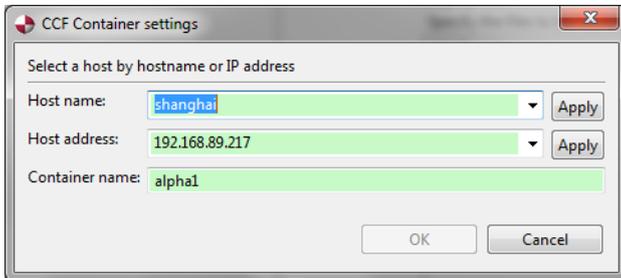
```
load -s $ses_id -c alpha2 -clf DinoListTest.clf
```

alpha2 is the slave name. Add and name the containers as you like.

The master and the single slaves will be defined in separate containers in the next step.

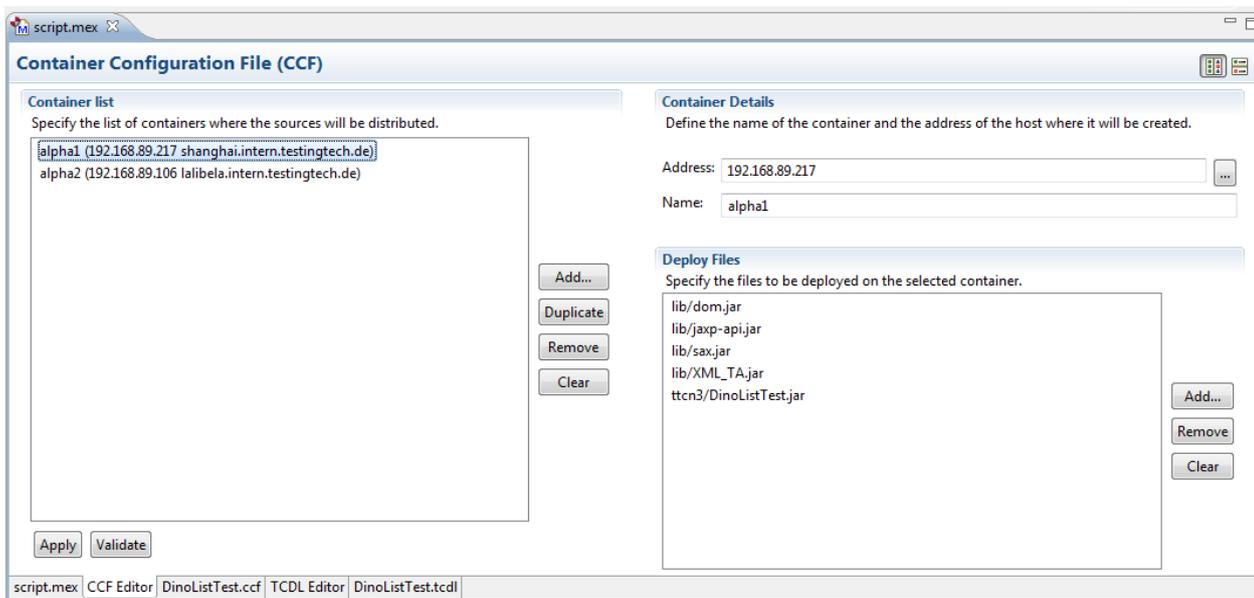
#### 4.1.2. Container Configuration File (CCF) editor

Press Duplicate button to add a new container, change the Host name, Host address and the container name. The container names must be the same as in the script.mex setting.



All containers (here alpha1 and alpha2) must contain the **Deploy files** too.

- The **Deploy files** are runnable and configuration files needed for test execution, including:
  - the jar files compiled from ttcn3 source,
  - all the (not built-in) plugin jar files,
  - configuration files like TT3plugins.xml and taconfig.xml files.
 These will be transferred to the slave daemon(s) when starting the test session.



This setting is then saved in DinoListTest.ccf file in XML format.

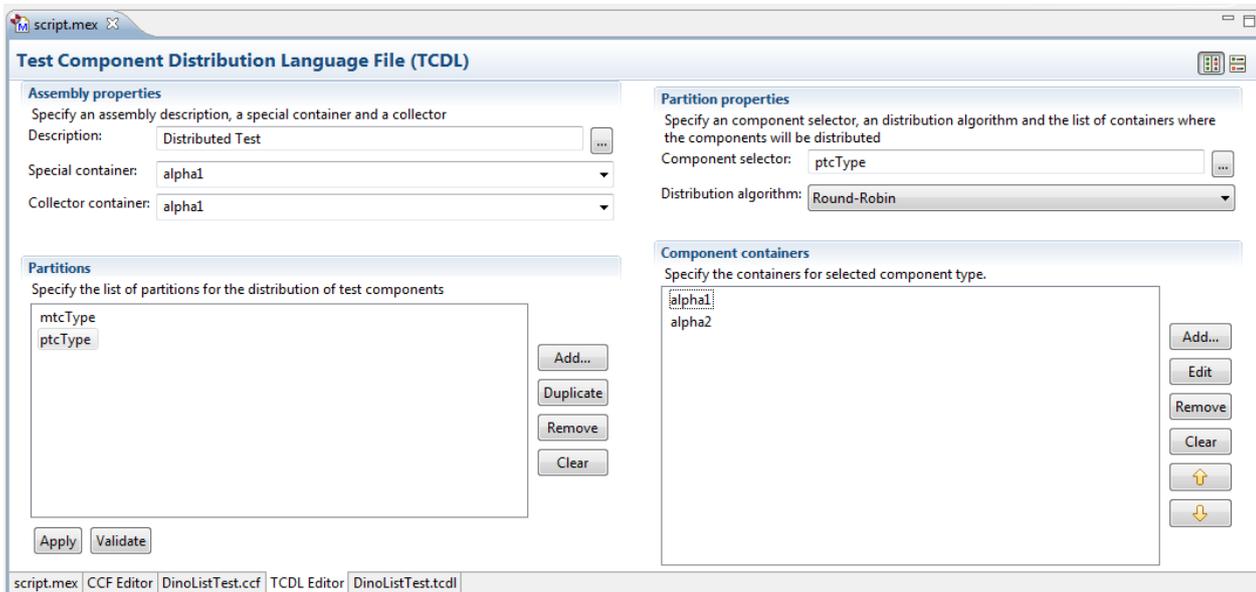
#### 4.1.3. Test Component Distribution Language (TCDL) Editor

mtcType uses the main container, here alpha1.

ptcType uses both containers, alpha1 and alpha2.

Generally the distribution algorithm of Round-Robin is used, i.e., the PTC instances are distributed on the master and the slaves in row and then in circle. For example in case of 4 PTCs, PTC[0] and PTC[2] on the master apla1, PTC[1] and PTC[3] on the slave alpha2.

This setting is saved in DinoListTest.tcdl file in XML format.



## 4.2. Daemon preparation

### 4.2.1. Master daemon

Copy and unzip the **TTmexDist.zip** from  
 <TTworkbench installation directory>\plugins\com.testingtech.ttworkbench.ttmex.rt\_x.x.x.x  
 to your favored folder locally (on your master computer).

The unzipped **TTmex\_x.x.x** directory is now your **TTmex root path**.

### 4.2.2. Slave daemon(s)

- Copy the **TTmex\_x.x.x** directory to your slave computer(s)
- Copy the plugin folders from tt3plugins folder from your project to the tt3plugins folder in the **TTmex\_x.x.x** directory
- If you have installed some plugin in your TTworkbench, copy that from your TTworkbench installation directory under plugins folder to the tt3plugins folder in the **TTmex\_x.x.x** directory
- Add the license file in the the **TTmex\_x.x.x** directory. Make sure you have enough features for TTworkbench, TTmex daemon and all the plugins in your license. Each (master and slave) daemon needs a set of features in the license.

## 5. Running Test Case

### 5.1. Starting master daemon

Before running the test case, the master daemon must be started from the TTmex root path. Open a command line console, go to the TTmex\bin path and start the **master daemon** with the .bat/.sh file.

```
.\TTmex_1.1.17\bin>startMasterDaemon.bat
```

Press any key. Two more consoles will be started – Session manager and daemon. If you see the following, they are started successfully.

```
[session manager started successfully]
```

```
[daemon started successfully (IP=<You master IP address>)]
```

Keep them open.

## 5.2. Starting slave daemon

On the slave computer, edit startDaemon.bat with any text editor and modify the following line:

```
if not defined %NS_HOST (set NS_HOST=<Master IP address>)
```

Open a command line console, go to the TTmex\bin path and start the **daemon** with the .bat/.sh file.

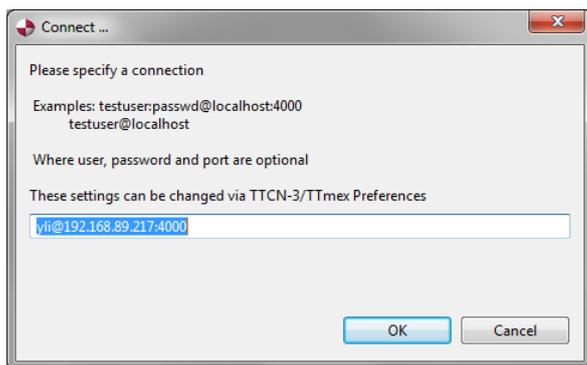
```
.\TTmex_1.1.17\bin>startDaemon.bat
```

Attention: Not the master but the (slave) daemon.

## 5.3. Start test case in TTworkbench on the master computer

In TTworkbench open TTmex console over Window > Show view > Other > TTmex > TTmex console. You will see a new tab "TTmex Console" under the core language editor.

Click the connect button and confirm the selected connection.



You are asked to start the (master and slave) daemons if not done.

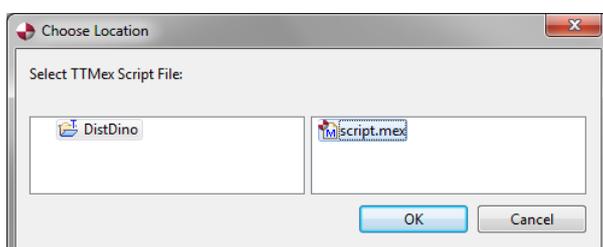
You will see the following **on the slave daemon**:

```
[daemon started successfully (IP=<You slave IP address>)]
```

Press the execution button to run the test case.



Choose the TTmex script.



After execution, a pop up window indicates the script execution is terminated.

The log file exec.tlz is saved in the project root directory (as set in the script.mex file).

## 5.4. Analysis

- Double click on the exec.tlz file and switch to the execution perspective.
- Click on the TTCN-3 Graphical Logging tab.  
It shows the results of the execution as a graphical presentation.
- A single click on a send/receive message arrow provides further details of the TTCN-3 template representation in the Test Data View.
- A single click on a receive arrow shows you the received message compared to the expected TTCN-3 template. Mismatched values are marked red.
- In the Dump tab you can check the message in octetstring format.

The screenshot displays the TTCN-3 Execution Management interface. It is divided into several panes:

- Management View:** Shows the test case 'DinoListTest\_1' with 1 run.
- Test Parametrization:** Shows a parameter 'NUMBER\_OF\_PTCS' with a value of 10.
- Test Data View:** Displays 'Expected TTCN-3 Template' and 'Data'. The 'Data' pane shows a table with columns 'Name' and 'Value'. It lists two instances of 'dinoListType'. The first instance has values: name: Brachiosaurus, len: 35 m, mass: 80 tonnes, time: Albian to Cenomanian, place: Argentina. The second instance has values: name: Brachiosaurus, len: 22 m, mass: 30 tonnes, time: Kimmeridgian, place: Portugal. Mismatched values are highlighted in red.
- Graphical Online Logging:** Shows a sequence diagram with components like 'SYSTEM', 'DinoList...', and 'httpPort...'. It displays message flows (receive d:hoistType) and local timer events (localTimer(P.739), localTimer(P.852), localTimer(P.886)).

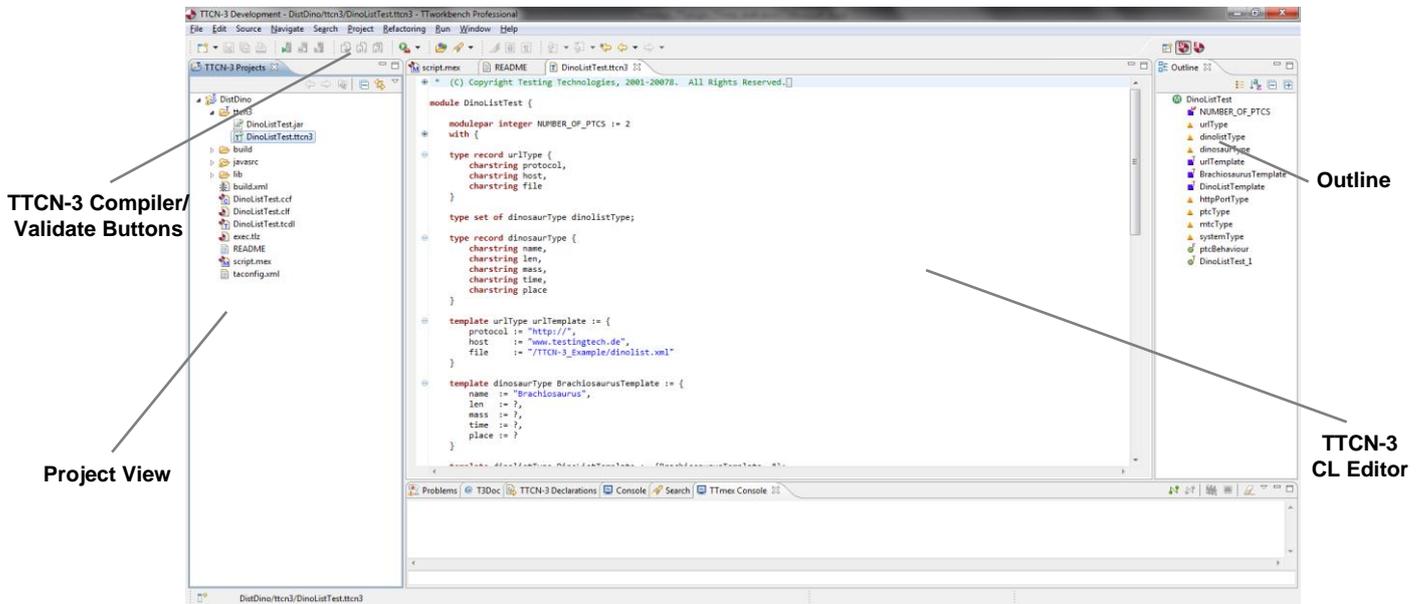
Annotations with arrows point to these panes: 'Management View' points to the top-left pane, 'Test Parametrization' points to the bottom-left pane, 'Result Analyser' points to the 'Data' table in the top-right pane, and 'Graphical Online Logging' points to the bottom-right pane.

For more details, please refer to the built-in manual in TTworkbench over [Help](#) → [Help contents](#) → [Testing Technologies TTworkbench User's Guide](#) > [Using TTworkbench TTman](#) → [Using TTman](#)

## 6. Details

### 6.1. Core Language Editor (CL Editor)

Each TTCN-3 file can be opened by double clicking in the project view and then be edited in the CL Editor. After opening the files, the source code will be syntactically checked and highlighted. In addition, an outline will be generated automatically and the go-to-declaration feature will be enabled.



### 6.2. TTthree (TTCN-3 Compiler)

For recompiling all TTCN-3 files open **DinoListTest.ttcn3** in the CL Editor and press the **Rebuild** button .

**NOTE!** Compilation is only necessary if changing test cases or creating new ones. Otherwise, the generated **\*.jar** files from TTCN-3 source are already available for execution.

### 6.3. TTman (Test Execution Management)

In order to execute the standalone test cases and configure parameters, the CLF has to be loaded into the Test Execution Management.

For more details, please refer to the built-in manual in TTworkbench over *Help* → *Help contents* → *Testing Technologies TTworkbench User's Guide* > *Using TTworkbench TTman* → *Using TTman*

Additionally you can find further documentations of TTmex usage.

Built-in manual in TTworkbench: *Help* → *Help Contents* → *Testing Tech TTmex User's Guide*.

In the DistDino Example project: *README*

In the TTmex-xxx root directory: *README*

## 7. Appendix

### 7.1. Acronyms

<b>CCF</b>	Container Configuration File
<b>CD</b>	Coding/Decoding
<b>CL Editor</b>	Core Language Editor
<b>CH</b>	Component Handler
<b>CLF</b>	Campaign Loader File
<b>IUT</b>	Implementation Under Test
<b>MTC</b>	Main Test Component
<b>PTC</b>	Parallel Test Component
<b>TCDL File</b>	Test Component Distribution Language File

### 7.2. Notes

This document is subject to change without notice.

Testing Technologies IST GmbH  
Michaelkirchstraße 17/18  
10179 Berlin, Germany

Phone: +49 30 726 19 19 -0  
Fax: +49 30 726 19 19 -20  
Email: [info@testingtech.com](mailto:info@testingtech.com)  
Internet: [www.testingtech.com](http://www.testingtech.com)

Testing Technologies, the Testing Tech logo, TTworkbench and TTsuite are trademarks or registered trademarks of Testing Technologies IST GmbH. Other terms are used for identification purposes and are trademarks or registered trademarks of their respective companies. Testing Technologies' TTsuite is powered by Eclipse technology and includes Eclipse plug-ins that can be installed and used with other Eclipse 3.7.2-based offerings. It includes software developed by the Apache Software Foundation (<http://www.apache.org/>), ANTLR (<http://www.antlr.org/>), Tigris.org (<http://gef.tigris.org/>), and L2FProd.com (<http://L2FProd.com/>).