Technical Document

GSM PROTOCOL STACK

GPF

TCGEN – TESTCASE GENERATOR

USER GUIDE

Document Number:	06-03-32-UDO-003
Version:	0.3
Status:	Draft
Approval Authority:	
Creation Date:	2002-May-13
Last changed:	2015-Mar-08 by Ronny Kiessling
File Name:	Tcgen_userguide.doc

Important Notice

Texas Instruments Incorporated and/or its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products, software and services at any time and to discontinue any product, software or service without notice. Customers should obtain the latest relevant information during product design and before placing orders and should verify that such information is current and complete.

All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment. TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI products, software and/or services. To minimize the risks associated with customer products and applications, customers should provide adequate design, testing and operating safeguards.

Any access to and/or use of TI software described in this document is subject to Customers entering into formal license agreements and payment of associated license fees. TI software may solely be used and/or copied subject to and strictly in accordance with all the terms of such license agreements.

Customer acknowledges and agrees that TI products and/or software may be based on or implement industry recognized standards and that certain third parties may claim intellectual property rights therein. The supply of products and/or the licensing of software does not convey a license from TI to any third party intellectual property rights and TI expressly disclaims liability for infringement of third party intellectual property rights.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products, software or services are used.

Information published by TI regarding third–party products, software or services does not constitute a license from TI to use such products, software or services or a warranty, endorsement thereof or statement regarding their availability. Use of such information, products, software or services may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Table of Contents

0 Document Control

0.1 Change History

Date	Changed by	Approved by	Version	Status	Notes
2002-May-13	RK		0.1		1
2003-May-20	XINTEGRA		0.2	Draft	
2003-Aug-18	RK		0.3	Draft	2

Notes:

1. Initial version
2. New Document ID introduced

0.2 List of Figures and Tables

0.3 List of References

[GSM 2.30]	ETS 300 511: July 1995 (GSM 02.30 version 4.13.0) Man-Machine Interface (MMI) of the Mobile Station (MS), ETSI
[PCO2_UG]	06-03-35-UDO, PCO2 – Tracing Environment (pco_userguide.doc)
[PCO2_D]	06-03-35-SLL, PCO2 – Tracing Environment (pco_description.doc)
[PCO2_TCGEN]	PCO2 – Tracing for TCGEN (pco_tracing4tcgen.pps)
[TCGEN_D]	06-03-32-SLL, TCGen – TestCase Generator (tcgen_description.doc)
[TCGEN_I]	TCGen – Introduction (tcgen_intro.pps)
[TCGEN_DIPL]	TCGen – Diploma Thesis (tcgen_diploma_thesis.pdf)
[XM]	06-03-55-UDO, XM –GUI-frontend for GPF m.bat (xm_userguide.doc)
[MOAN]	06-03-53-UDO, MoanBtn – Instant GUI-problem Informer (mbtn_userguide.doc)

0.4 Abbreviations

ACI	Application Control Interface (AT Commands)
G23	The Condat implementation of Layers 2 and 3 of the GSM Protocol Stack
G23 Target System	Hardware which executes G23
LCD	Liquid Crystal Display
MM	Mobility Management
MMI	Man Machine Interface
MOC	Mobile Originated Call
MTC	Mobile Terminated Call
PC	Personal Computer
PCO	Point of Control and Observation
PIN	Personal Identification Number
RS232	Serial Communication Standard
Target System	Shortened form of 'G23 Target System'
TCGEN	TestCase Generator

0.5 Terms

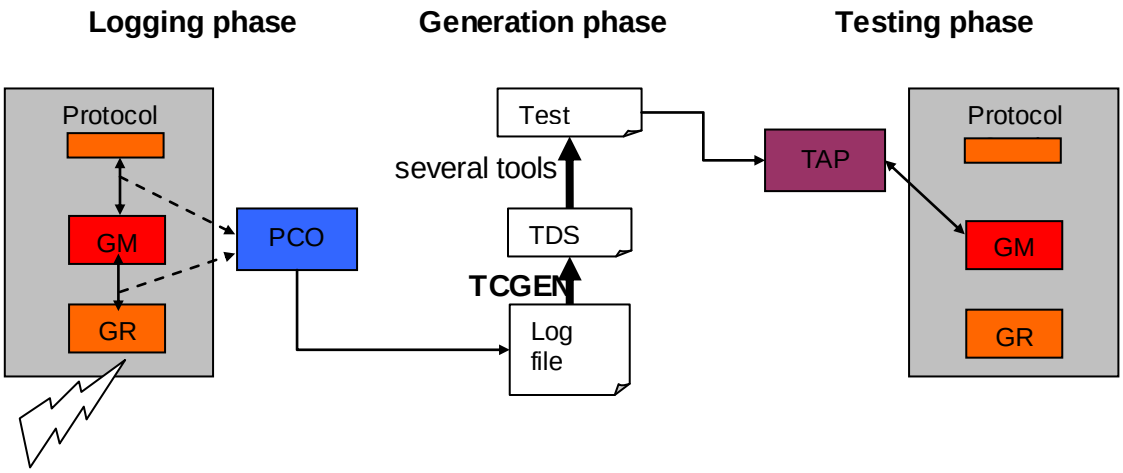
Entity	Program which executes the functions of a layer
Message	A message is a data unit which is transferred between the entities of the same layer (peer-to-peer) of the mobile and infrastructure side. Message is used as a synonym to protocol data unit (PDU). A message may contain several information elements.
Primitive	A primitive is a data unit which is transferred between layers on one component (mobile station or infrastructure). The primitive has an operation code which identifies the primitive and its parameters.

Service Access Point A Service Access Point is a data interface between two layers on one component (mobile station or infrastructure).

1 Introduction

With the tool PCO2 (see [PCO2_UG] and [PCO2_D]) it is possible to log traffic of primitives inside a running protocol stack, especially during a field test. On the other hand the tool TAP (see [TAP]) can send and receive specified primitives to/from a PS and therefore simulate a specific test situation.

TCGen is a tool for automatically test case generation from PCO log files. It can be used to e.g. rerun field tests within a debugging environment on a PC. It parses a log file containing all crucial primitives sent from/to the entities under test and creates a TDS-file which can be transformed to a test case DLL-file in the usual way.
The following graphic gives an overview about the general process:



See [TCGEN_D] and [TCGEN_DIPL] for detailed descriptions of the internals of TCGen.

2 Application Manual

2.1 Environment / Installation

To use TCGen some environmental constraints have to be taken into account.

For use under Windows:
At first you'll have to make sure that several DLL/EXE-files are available to the system. In the Condat development directory structure you can find them in „<View>/GPF/Bin" :

- Frame-DLLs (frame.dll, tif.dll, misc.dll)
- CCD-DLLs (ccddata_load.dll, ccddata-DLL)
- VCMS-DLL (CMS.dll)
- Testinterface of the Frame (tst.exe)

So just make sure „<View>/GPF/Bin" is in your PATH-variable.

That means if you start "tcgen.exe" from „<View>/GPF/Bin" everything will be fine.

2.2 Getting started

For a comprehensive introduction with a lot of screen shots please refer to [TCGEN_I].
For an introduction on how to use PCO together with TCGen see [PCO2_TCGEN].

2.3 Command line parameters and ini-files

At the command line of tcgen.exe you can specify several parameters as follows:

- -h … print information
- -ver … print version of TCGEN
- -i <ini-file> … optionally use specified ini-file
- -wait … optionally wait for key input before exit
- -entries <start-index> <end-index>
  … optionally use only the entries from <start-index> to <end-index> of the .pco-logfile
- -analyze … analyze the .pco-logfile only and print report
- -n <testcase-name>
  … optionally use specified name for the new testcase (default is same as .pco-logfile)
- -p <project> … optionally use specified name for DevStudio project an test dll (default is name of first entity under test)
- <pco-input-file> … the name of the .pco-logfile to be used
- <output-dir> <entities>
  … if "-analyze" has not been used the output directory for the new test case as well as a comma separated list of entity names to be tested has to be specified

A proper example of a call to tcgen would be:

```
tcgen mmgmm001.pco \g23m\condat\ms\src\mm\test_mmgmm MM,GMM -p mmgmm -n
MMGMM023
```

TCGen sources various information from an ini-file. Per default "tcgen.ini" (in the current dir or usually in ..\cfg\) is used, but a different one can be specified at command line.

In an ini-file only lines of the following format will be interpreted:
<parameter><whitespace><value>

An overview of the available parameters follows:

- rules <rules-file> the rules xml-file to be used
- tdcincdir <dir> TDC include dir (e.g. \g23m\condat\ms\tdcinc)
- tdcinclib <lib-path> TDC include library (e.g. \g23m\condat\ms\tdclib\tdcinc.lib)
- tdclib <lib-path> TDC library (e.g. \GPF\LIB\win32\tdc.lib)
- pcon <value> Use PCON (1|0)
- param_constr <value> Use parameterized TDC constructors (0|1)
- param_level <value> Level up to which parameterized structs shall be used (0..n)
- param_nl <value> Level up to which new lines after each parameter in parameterized structs will be generated (0..param_level)

You can find examples in the standard tcgen.ini file in GPF/cfg or in the “tcgen.ini.*”-files in GPF/util/tcgen/cfg.

2.4 Rules files

Rules can be used to instruct TCGen to apply certain changes to the primitives and their elements/values during generation.

In general so-called “skip”- and “change”-rules are distinguished. The first allow explicit skipping of whole primitives or individual elements. By the latter names and values can be changed. Concerning the specification of changes which shall be applied all possibilities offered by the test script language can be used, since the test case generator will just substitute the original value by the specified statement. If placeholders are found the value from the logfile is inserted for each ‘%v’.

Rules files are xml-files matching the following BNF:

```
<RulesDescription> ::=<Options> <RuleList>
<RulesList>        ::=<Rule> [<RulesList>]
<Rule>             ::=<SkipRule> | <ChangeRule>
<Options>          ::='options' ['max_timegap='<Integer>] ['timeouts='<Integer>]
<SkipRule>         ::='skip' 'primitive='<Mask> ['param='<Mask>]
<ChangeRule>       ::='change' 'primitive='<Mask> ['param='<Mask>]
                      <ChangeStatement>
<ChangeStatement>  ::=<Test-Script-Statement
                      [including one ore more <Placeholder>]>
<Mask>             ::=<String>['*']
<Placeholder>      ::='%v'
```

An example can be found in GPF/cfg: tcgen_rules.xml.default.

2.5 Working with TCGen

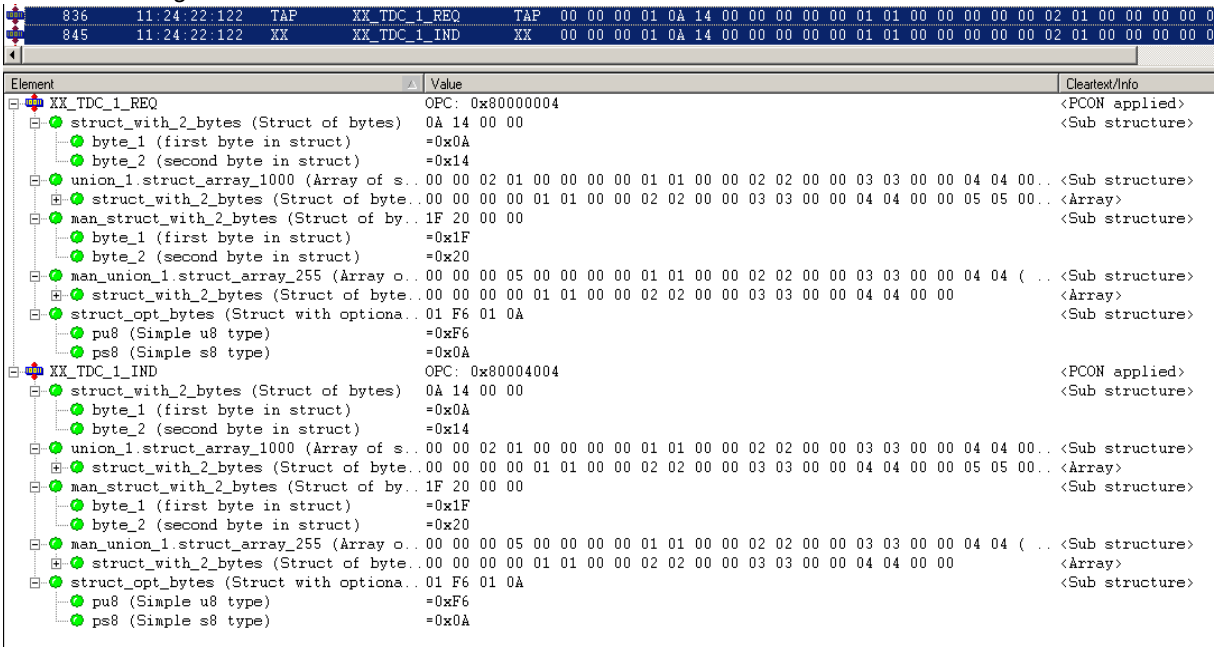
2.5.1 Usage examples

Following examples should explain the most common usage situations of TCGen.

1) No parameterization, no rules

This mode can be used with every TDCINC-files independent whether they contain parameterized constructors or not.
For each primitives send or awaited in the generated step a corresponding function will be created in the constraints.
For each sub structure used inside the generated primitives a corresponding function will be created in the constraints.

Part of PCO-logfile:



836	11:24:22:122	TAP	XX_TDC_1_REQ	TAP	00 00 00 01 0A 14 00 00 00 00 00 01 01 00 00 00 00 00 02 01 00 00 00 00 0	
845	11:24:22:122	XX	XX_TDC_1_IND	XX	00 00 00 01 0A 14 00 00 00 00 00 01 01 00 00 00 00 00 02 01 00 00 00 00 0	

Element	Value	Cleartext/Info
XX_TDC_1_REQ	OPC: 0x80000004	<PCON applied>
struct_with_2_bytes (Struct of bytes)	0A 14 00 00	<Sub structure>
byte_1 (first byte in struct)	=0x0A	
byte_2 (second byte in struct)	=0x14	
union_1.struct_array_1000 (Array of s..	00 00 02 01 00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00..	<Sub structure>
struct_with_2_bytes (Struct of byte..	00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00 00 05 05 00..	<Array>
man_struct_with_2_bytes (Struct of by..	1F 20 00 00	<Sub structure>
byte_1 (first byte in struct)	=0x1F	
byte_2 (second byte in struct)	=0x20	
man_union_1.struct_array_255 (Array o..	00 00 00 05 00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 (..	<Sub structure>
struct_with_2_bytes (Struct of byte..	00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00 00	<Array>
struct_opt_bytes (Struct with optiona..	01 F6 01 0A	<Sub structure>
pu8 (Simple u8 type)	=0xF6	
ps8 (Simple s8 type)	=0x0A	
XX_TDC_1_IND	OPC: 0x80004004	<PCON applied>
struct_with_2_bytes (Struct of bytes)	0A 14 00 00	<Sub structure>
byte_1 (first byte in struct)	=0x0A	
byte_2 (second byte in struct)	=0x14	
union_1.struct_array_1000 (Array of s..	00 00 02 01 00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00..	<Sub structure>
struct_with_2_bytes (Struct of byte..	00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00 00 05 05 00..	<Array>
man_struct_with_2_bytes (Struct of by..	1F 20 00 00	<Sub structure>
byte_1 (first byte in struct)	=0x1F	
byte_2 (second byte in struct)	=0x20	
man_union_1.struct_array_255 (Array o..	00 00 00 05 00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 (..	<Sub structure>
struct_with_2_bytes (Struct of byte..	00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00 00	<Array>
struct_opt_bytes (Struct with optiona..	01 F6 01 0A	<Sub structure>
pu8 (Simple u8 type)	=0xF6	
ps8 (Simple s8 type)	=0x0A	

Part of TCGen ini-file:

```
# Use parameterized TDC constructors (0|1)
param_constr  0

# Level up to which parameterized structs shall be used (0..n)
param_level 0
```

Part of generated TDC testcase:

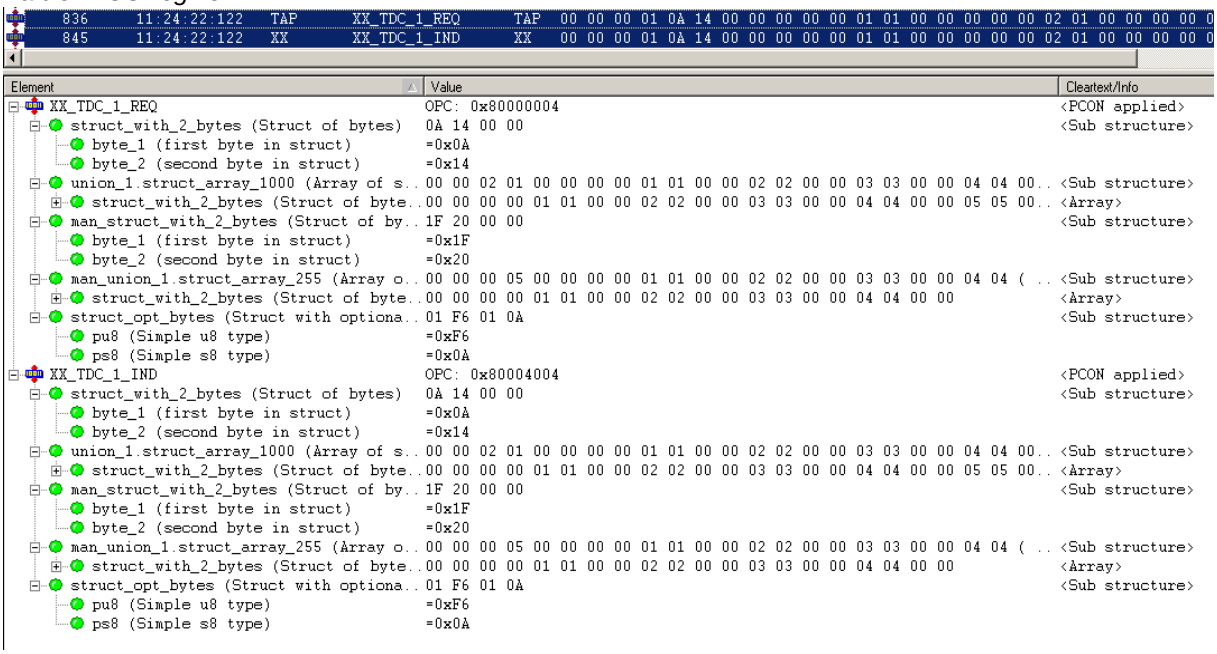
```
  BEGIN_STEP ("run_logged_stuff_from_XX_TEST23")
  {
     port2XX.SEND (xx_tdc_1_req_XX_TEST23_0());
     AWAIT (xx_tdc_1_ind_XX_TEST23_1());
  }
T_PRIMITIVE_UNION xx_tdc_1_req_XX_TEST23_0()
{
   T_XX_TDC_1_REQ prim;
   prim->struct_with_2_bytes= struct_with_2_bytes_XX_TEST23_0(); //
   prim->union_1.struct_array_1000= struct_array_1000_XX_TEST23_1(); //
   prim->man_struct_with_2_bytes= man_struct_with_2_bytes_XX_TEST23_2(); //
   prim->man_union_1.struct_array_255= struct_array_255_XX_TEST23_3(); //
   prim->struct_opt_bytes= struct_opt_bytes_XX_TEST23_4(); //
   return prim;
} // xx_tdc_1_req_XX_TEST23_0

T_XX_TDC_1_struct_with_2_bytes struct_with_2_bytes_XX_TEST23_0()
{
   T_XX_TDC_1_struct_with_2_bytes pstruct;
   pstruct->byte_1= 0x0A; //
   pstruct->byte_2= 0x14; //
   return pstruct;
} // struct_with_2_bytes_XX_TEST23_0
```

2) Parameterization w/o parameterized constructors, no rules

This mode can be used with every TDCINC-files independent whether they contain parameterized constructors or not.

For each primitive-type send or awaited in the generated step a corresponding parameterized function will be created in the constraints.
For each sub structure used inside the generated primitives a corresponding function will be created in the constraints.

Part of PCO-logfile:



836	11:24:22:122	TAP	XX_TDC_1_REQ	TAP	00 00 00 01 0A 14 00 00 00 00 00 01 01 00 00 00 00 00 02 01 00 00 00 0
845	11:24:22:122	XX	XX_TDC_1_IND	XX	00 00 00 01 0A 14 00 00 00 00 00 01 01 00 00 00 00 00 02 01 00 00 00 0

Element	Value	Cleartext/Info
XX_TDC_1_REQ	OPC: 0x80000004	<PCON applied>
struct_with_2_bytes (Struct of bytes)	0A 14 00 00	<Sub structure>
byte_1 (first byte in struct)	=0x0A	
byte_2 (second byte in struct)	=0x14	
union_1.struct_array_1000 (Array of s..	00 00 02 01 00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00..	<Sub structure>
struct_with_2_bytes (Struct of byte..	00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00 00 05 05 00..	<Array>
man_struct_with_2_bytes (Struct of by..	1F 20 00 00	<Sub structure>
byte_1 (first byte in struct)	=0x1F	
byte_2 (second byte in struct)	=0x20	
man_union_1.struct_array_255 (Array o..	00 00 00 05 00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 (..	<Sub structure>
struct_with_2_bytes (Struct of byte..	00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00 00	<Array>
struct_opt_bytes (Struct with optiona..	01 F6 01 0A	<Sub structure>
pu8 (Simple u8 type)	=0xF6	
ps8 (Simple s8 type)	=0x0A	
XX_TDC_1_IND	OPC: 0x80004004	<PCON applied>
struct_with_2_bytes (Struct of bytes)	0A 14 00 00	<Sub structure>
byte_1 (first byte in struct)	=0x0A	
byte_2 (second byte in struct)	=0x14	
union_1.struct_array_1000 (Array of s..	00 00 02 01 00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00..	<Sub structure>
struct_with_2_bytes (Struct of byte..	00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00 00 05 05 00..	<Array>
man_struct_with_2_bytes (Struct of by..	1F 20 00 00	<Sub structure>
byte_1 (first byte in struct)	=0x1F	
byte_2 (second byte in struct)	=0x20	
man_union_1.struct_array_255 (Array o..	00 00 00 05 00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 (..	<Sub structure>
struct_with_2_bytes (Struct of byte..	00 00 00 00 01 01 00 00 02 02 00 00 03 03 00 00 04 04 00 00	<Array>
struct_opt_bytes (Struct with optiona..	01 F6 01 0A	<Sub structure>
pu8 (Simple u8 type)	=0xF6	
ps8 (Simple s8 type)	=0x0A	

Part of TCGen ini-file:

```
# Use parameterized TDC constructors (0|1)
param_constr  0

# Level up to which parameterized structs shall be used (0..n)
param_level 1
```

Part of generated TDC testcase:

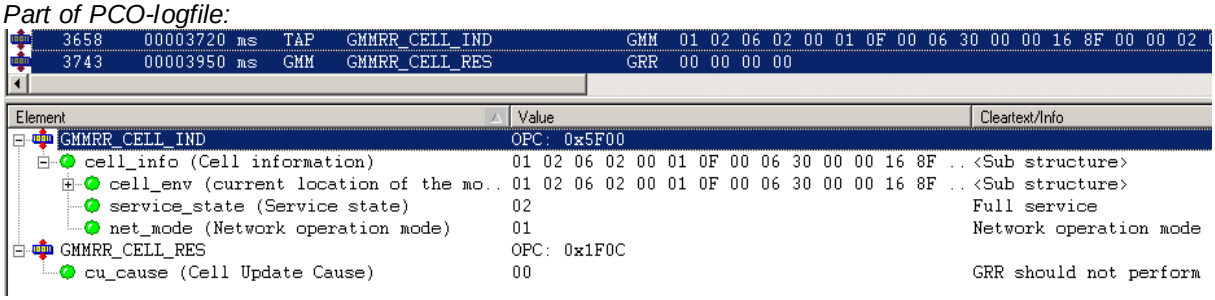
```
 BEGIN_STEP ("run_logged_stuff_from_XX_TEST23")
 {
    port2XX.SEND (xx_tdc_1_req_XX_TEST23(
      struct_with_2_bytes_XX_TEST23_0(),
      struct_array_1000_XX_TEST23_1(),
      man_struct_with_2_bytes_XX_TEST23_2(),
      struct_array_255_XX_TEST23_3(),
      struct_opt_bytes_XX_TEST23_4()));
    AWAIT (xx_tdc_1_ind_XX_TEST23(
      struct_with_2_bytes_XX_TEST23_5(),
      struct_array_1000_XX_TEST23_6(),
      man_struct_with_2_bytes_XX_TEST23_7(),
      struct_array_255_XX_TEST23_8(),
      struct_opt_bytes_XX_TEST23_9()));
 }
T_PRIMITIVE_UNION xx_tdc_1_req_XX_TEST23(T_XX_TDC_1_struct_with_2_bytes struct_with_2_bytes,
{
   T_XX_TDC_1_REQ prim;
   prim->struct_with_2_bytes= struct_with_2_bytes; //
   prim->union_1.struct_array_1000= struct_array_1000; //
   prim->man_struct_with_2_bytes= man_struct_with_2_bytes; //
   prim->man_union_1.struct_array_255= struct_array_255; //
   prim->struct_opt_bytes= struct_opt_bytes; //
   return prim;
} // xx_tdc_1_req_XX_TEST23
```

3) Parameterization with parameterized constructors, no rules

This mode can only be used with TDCINC-files containing parameterized constructors. To get these ccdgen has to be called with parameter –ipaco (or –paco).
No functions will be created for any of the primitives send or awaited in the generated step and, depending on the specified level of parameterization, no corresponding functions are created for the sub structures, either. Instead, the parameterized constructors are used directly.
For all sub structures residing deeper in the primitive than the specified level of parameterization a corresponding function will be created in the constraints.

CAVE! Parameterized constructors are currently not supported for union members, so the level of parameterization should be set appropriately.

Part of PCO-logfile:



Part of TCGen ini-file:

```
# Use parameterized TDC constructors (0|1)
param_constr  1

# Level up to which parameterized structs shall be used (0..n)
param_level 2

# Level up to which new lines after each parameter in
# parameterized structs will be generated (0..param_level)
param_nl  1
```

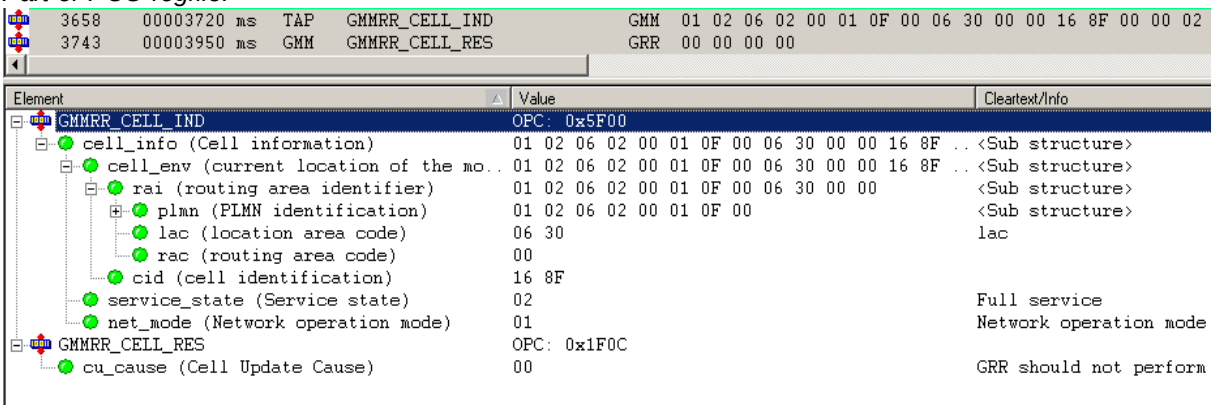
Part of generated TDC testcase:

```
  BEGIN_STEP ("run_logged_stuff_from_XX_TEST23")
  {
     port2GMM.SEND (T_GMMRR_CELL_IND(
       T_cell_info(cell_env_XX_TEST23_0()/**/, 0x02/*Full service*/,
     AWAIT (T_GMMRR_CELL_RES(
       0x00/*GRR should not perform Cell Update Access*/,0/*_aux*/));
  }
T_cell_env cell_env_XX_TEST23_0()
{
   T_cell_env pstruct;
   pstruct->rai= rai_XX_TEST23_1(); //
   pstruct->cid= 0x8F16; //
   return pstruct;
} // cell_env_XX_TEST23_0
```

4) No Parameterization, some rules

Rules can be applied in every parameterization mode and fully independent whether the TDCINC-files contain parameterized constructors or not.
Skip-rules will usually be resolved by un-commenting the specific element or primitive.
Change-rules are applied as specified.
See chapter 2.4 for detailed information concerning rules-files.

Part of PCO-logfile:



```
3658   00003720 ms  TAP   GMMRR_CELL_IND          GMM  01 02 06 02 00 01 0F 00 06 30 00 00 16 8F 00 00 02
3743   00003950 ms  GMM   GMMRR_CELL_RES          GRR  00 00 00 00
```

Element	Value	Cleartext/Info
GMMRR_CELL_IND	OPC: 0x5F00	
cell_info (Cell information)	01 02 06 02 00 01 0F 00 06 30 00 00 16 8F ..	<Sub structure>
cell_env (current location of the mo..	01 02 06 02 00 01 0F 00 06 30 00 00 16 8F ..	<Sub structure>
rai (routing area identifier)	01 02 06 02 00 01 0F 00 06 30 00 00	<Sub structure>
plmn (PLMN identification)	01 02 06 02 00 01 0F 00	<Sub structure>
lac (location area code)	06 30	lac
rac (routing area code)	00	
cid (cell identification)	16 8F	
service_state (Service state)	02	Full service
net_mode (Network operation mode)	01	Network operation mode
GMMRR_CELL_RES	OPC: 0x1F0C	
cu_cause (Cell Update Cause)	00	GRR should not perform

Part of TCGen ini-file:

```
# Level up to which parameterized structs shall be used (0..n)
param_level 0
```

Part of rules-file:

```
<skip primitive="GMMRR_CELL_RES"></skip>
<skip primitive="GMMRR_CELL_*" param="service_state"></skip>
<change primitive="GMMRR_CELL_IND" param="net_mode">(%v>=1 ? 0x5 : %v)</change>
```

Part of generated TDC testcase:

```
BEGIN_STEP ("run_logged_stuff_from_XX_TEST23")
{
   port2GMM.SEND (gmmrr_cell_ind_XX_TEST23_0());
   // AWAIT (gmmrr_cell_res_XX_TEST23_1());
}
T_PRIMITIVE_UNION gmmrr_cell_ind_XX_TEST23_0()
{
   T_GMMRR_CELL_IND prim;
   prim->cell_info= cell_info_XX_TEST23_0(); //
   return prim;
} // gmmrr_cell_ind_XX_TEST23_0

T_PRIMITIVE_UNION gmmrr_cell_res_XX_TEST23_1()
{
   T_GMMRR_CELL_RES prim;
   prim->cu_cause= 0x00; // GRR should not perform Cell Update Access
   return prim;
} // gmmrr_cell_res_XX_TEST23_1

T_cell_info cell_info_XX_TEST23_0()
{
   T_cell_info pstruct;
   pstruct->cell_env= cell_env_XX_TEST23_1(); //
   // pstruct->service_state= 0x02; // Full service
   pstruct->net_mode= (0x01>=1 ? 0x5 : 0x01); // <change rule applied on 0x01>
   return pstruct;
} // cell_info_XX_TEST23_0
```

3 Known problems and future tasks

This paragraph is meant to show which bugs are already found (but not removed yet) and to provide an impression of future plans concerning this product.

3.1 Known bugs

•

3.2 „Soon implemented“

•

3.3 „Nice to have “

- GUI interface (but the PCO-GUI already contains some dialogs concerning TCGen)

Appendices

A. Acronyms

DS-WCDMA	Direct Sequence/Spread Wideband Code Division Multiple Access

B. Glossary

International Mobile Telecommunication 2000 (IMT-2000/ITU-2000)	Formerly referred to as FPLMTS (Future Public Land-Mobile Telephone System), this is the ITU's specification/family of standards for 3G. This initiative provides a global infrastructure through both satellite and terrestrial systems, for fixed and mobile phone users. The family of standards is a framework comprising a mix/blend of systems providing global roaming. <URL: http://www.imt-2000.org/>