Document Number
xxxx.xxx.00.001

Topic
A tool for automatically test case generation from PCO log files is needed to easily rerun field tests within a debugging environment on a PC.

Project Info

Manager:	AK
Resources:	RK
Start:	May 2002
End:	
Manpower:	
PS7 Status:	--

History

Date	**Vers.**	**Author**	**Description**
08.04.02	001	RK	Initial

Milestones/Deliverables

Date	**PS7 Name**	**Description**

Table of Contents

0 DOCUMENT CONTROL 3

0.1 REFERENCES 3
0.2 ABBREVIATIONS 3
0.3 TERMS 3

1 BRIEF PROBLEM DESCRIPTION: 5

2 REQUIREMENTS SPECIFICATION 7

3 PROBLEMS AND SOLUTION PROPOSALS 5

4 ESTIMATED EFFORTS 7

0 Document control

0.1 References

[PCO2_UG]	8415.090.00.002, May 15, 2001, PCO2 – Tracing Environment (pco_userguide.doc)
[PCO2_D]	8415.094.00.002, May 28, 2001, PCO2 – Tracing Environment (pco_description.doc)
[FRAME]	8434.100.02.001, Januar 04, 2002, Frame Users Guide (frame_users_guide.doc)
[TAP]	8415.028.99.301, Januar 15, 2002, TCC - Test Case Control

0.2 Abbreviations

ACI	Application Control Interface (AT Commands)
G23 Stack	The Condat implementation of Layers 2 and 3 of the GSM Protocol
G23 Target System	Hardware which executes G23
LCD	Liquid Crystal Display
MM	Mobility Management
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
TAP	Test Application Process
Target System	Shortened form of 'G23 Target System'
TCGEN	Test Case Generator

0.3 Terms

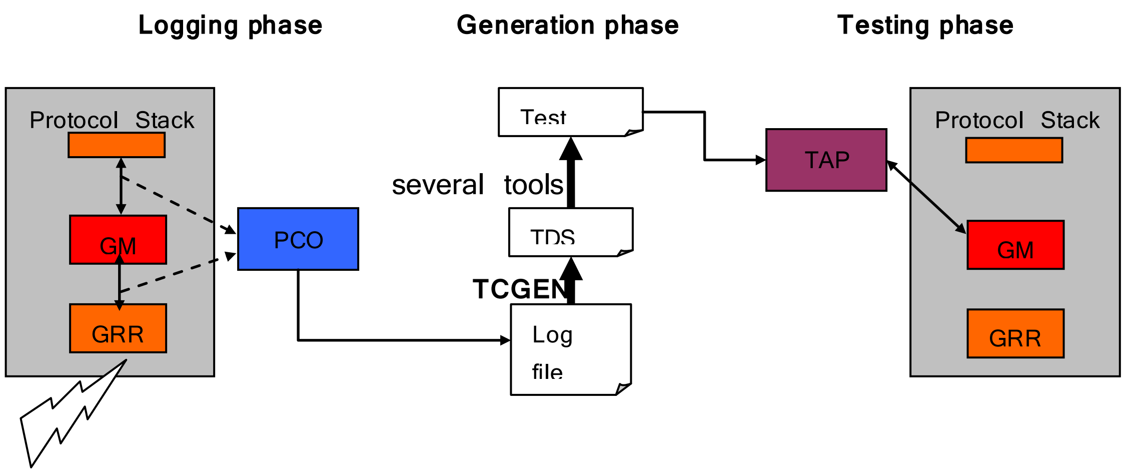
Entity	Program which executes the functions of a layer

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 Brief Problem Description:

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.

A tool (codename: TCGEN) for automatically test case generation from PCO log files is needed to e.g. rerun field tests within a debugging environment on a PC. It should parse a log file containing all crucial primitives sent from/to the entities under test and create 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 suggested process:



2 Problems and Solution Proposals

The following table lists the problems found during the meetings concerning tcgen as well as possible solutions (advised solutions are printed **bold**):

Problem	Possible solution
- no continuous time stamps in log file with current FRAME time	- PCO server modifies time stamp to PC time **- introduction of new TST-header with bigger time field**
- currently the originally receiver of a primitive is lost ("PCO" is inserted by the FRAME instead)	- the OPC/SAP can be used to retrieve informations about the receiver **- introduction of new TST-header supporting**

	the “original receiver”-field of FRAME-header
- the routings while logging have to ensure that all necessary primitives will be available	- appropriate combinations of FRAME system primitives have to be arranged **(GMM will be used as the entity under test for a first try)**
- length of primitives which can be sent via the test interface is currently restricted to 255 bytes by the TI-Multiplexer	- compression of primitives - usage of PCON - primitive splitting - improved TI-Multiplexer
- many runtime depending parameters exist (e.g. IMSI, transaction numbers, ...) which will not match the later test environment	- collection of all critical parameters in a list and modification of the generated test case by a developer supported by TCGEN **- all critical parameters are logged (esp. SIM primitives have to be traces)**
- start state of the field test PS may differ from the initial state of the PS in test environment	**- reset before each logging/testing process** - defined state set using CONFIG primitives
- CONFIG primitives routed by the FRAME, too ?	- to be clarified and evtl. implemented by MP **- PCO writes SYSTEM primitives into log file as well**
- load/traffic due to massive routing critical ?	**- to be tested by GPRS group**
- there are no SAP/primitive/message-description documents for some entities - decoding may fail	**- support of pure binary primitives inTDS-files by using BEGIN_ARRAY instead of FIELD**
- logging should start with switching on	- routings setting in stack init-function
- timing is different on target vs. PC	**- tolerance should be adjustable**
- timing vs. debugging with breakpoints	- synchronous TAP - dedicated test entity
- descriptor lists/ pointers are currently not traced	- introduction of test primitives on target as in simulation stack - FRAME enhancement
- functional interfaces	- monitor SAPs - function to primitive as for GTI

- txt-format as produced by saving a word-test-document in ASCII may be more readable/better editable for developer	- a dedicated TDS-2-TXT converter would be needed
- TAP usually stops on every small error (e.g. in some parameters)	**- TAP should support a "tolerant mode"**

3 Requirements Specification

The following notes describe the requirements for a first version of TCGEN (and depending tools):

- The TST-header has to be expanded to contain bigger time values and the original receiver.
- Descriptor lists have to be supported in any way.
- The TAP has to be extended to support a "tolerance mode" and more detailed timing issues.
- The PCO-server has to write system primitives into the logfile
- The output of TCGEN will be in TDS format.
- Primitives/Air messages which cannot be interpreted/decoded have to be stored as binary arrays.
- GMM will be used as the first entity under test -> Appropriate loggings have to be done.

Tasks for later versions may be:

- Dedicated loggins for other/more entities have to be done, e.g. upper/lower edges have to be defined.
- Functional interfaces should be supported in any way.
- The TAP should be able to run synchronously with the PS.
- Critical timings, e.g. in GRR, should be supported – maybe by a TAP substituting test entity running directly in the PS.
- A GUI interface should be a implemented.

4 Estimated Efforts

The following table contains the currently identified tasks and estimated minimal efforts for a first version of TCGEN:

Task	**Who**	**Hours**	**Hours** finished
Getting used to TDS format	RK	3	
Requirements specification	RK et al.	3	√
Pre-Tests with enabled redirection and log file production	ANS, RK	5	
Concept and specification	RK, HSC, AK, FR	10	√
Implementation of first command line version	RK	60	
FRAME modifications	MP		
TAP modifications	CKR		
PCO2 modifications	RK	10	
Integration test with other test case tools and TAP	RK, CKR	20	
General Tests	RK, CKR, GPRS-Team	50	
User guide	RK	20	
Developers description	RK	40	
Summe		**221**	

(Ronny Kiessling (RK) currently works 20 hours a week.)