Automated Test Case Generation with TC-Gen

(an introduction)

TC-Gen …

Testing Concept Overview

Motivation

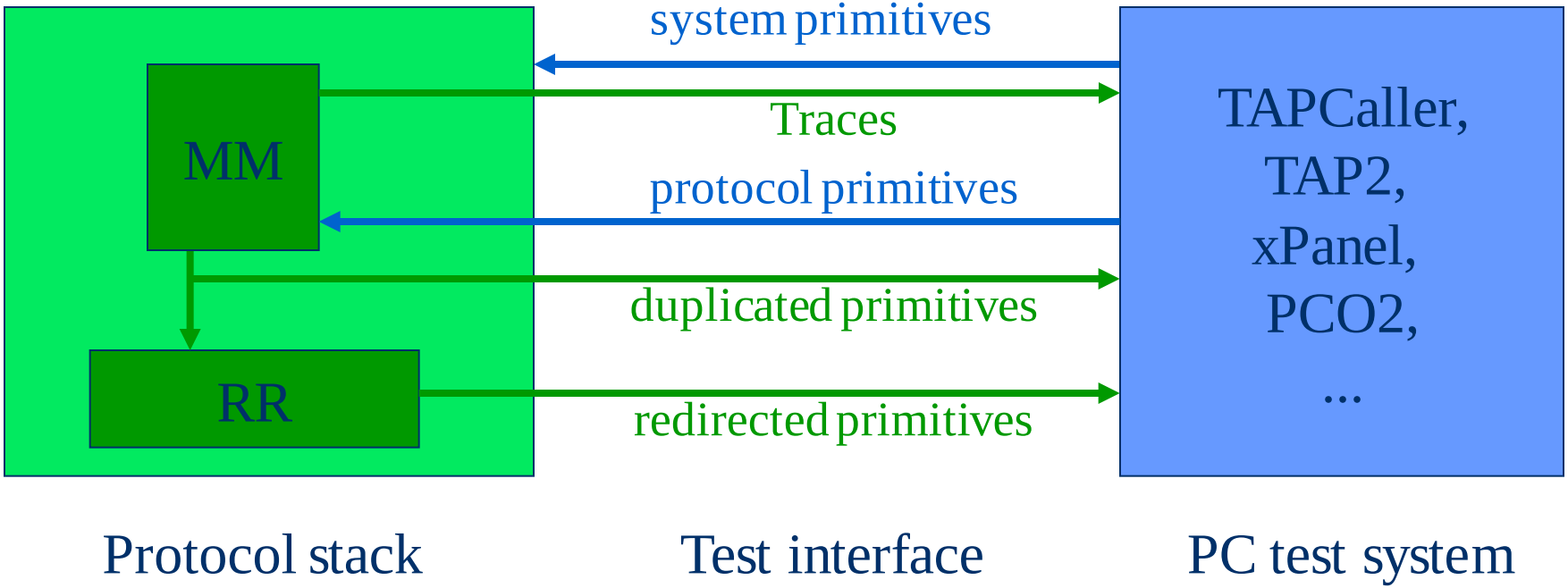
Usage

How it works

Current Limitations & Outlook

TC-Gen … *testing concept overview*

- Test interface approach:
  - ⇨ data interface between G23 protocol stack and a PC test system
  - ⇨ e.g. standard serial cable, COM-ports on both ends
  - ⇨ for running test cases: usually shared memory on one PC



TC-Gen … *testing concept overview*

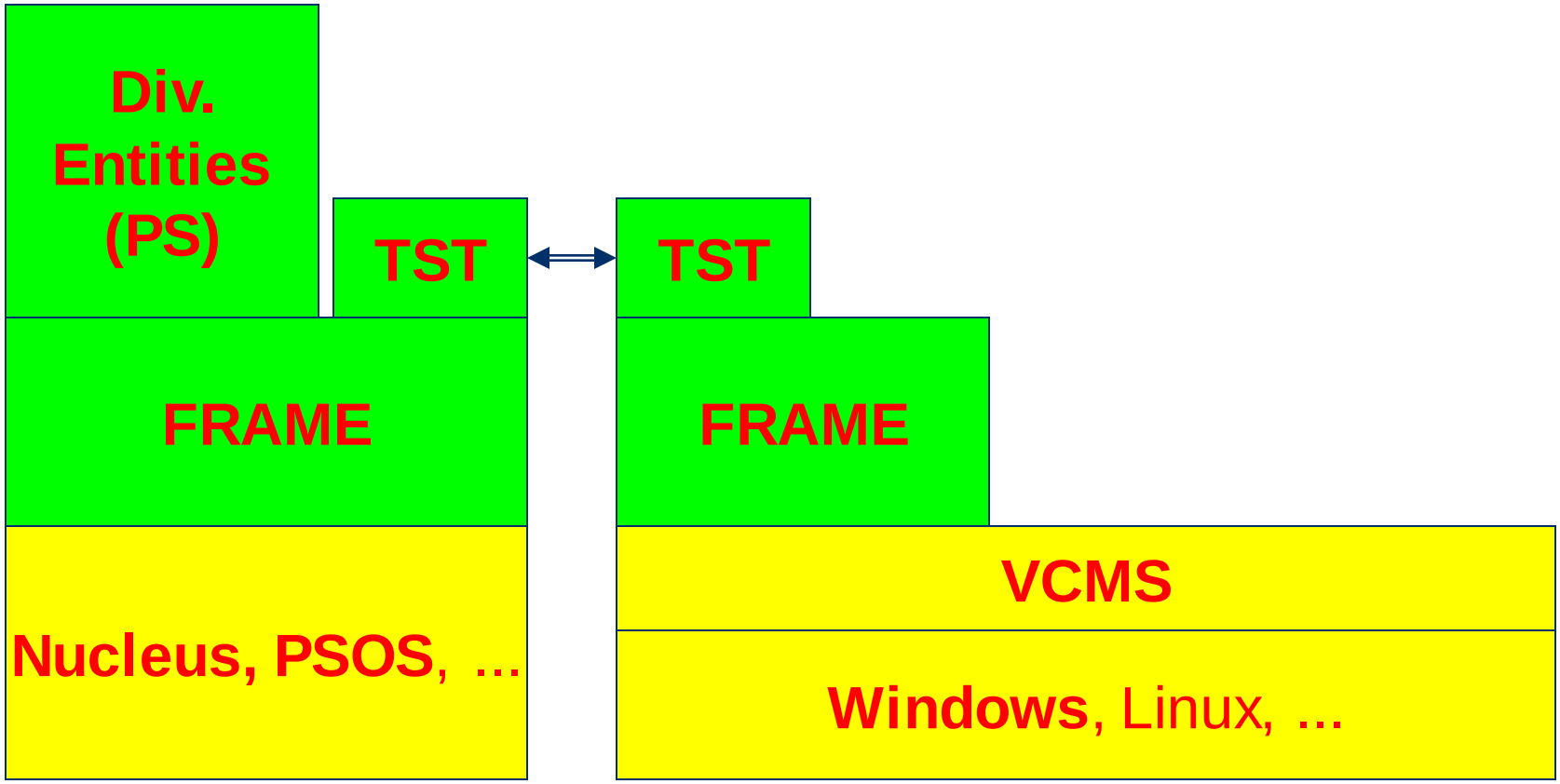
- On stack side:
  - ⇨ test interface entity included in the FRAME
  - ⇨ uses corresponding hardware driver for communication
- On PC test system side:
  - ⇨ test interface executable using the FRAME
    - ◆ connects using standard OS drivers or via a Multiplexer
  - ⇨ used by tools like TAP, xPanel & PCO
  - ⇨ these tools finally provide GUI-stack-access for testers

TC-Gen … *testing concept overview*

- TAP (Test Application Process) :
  - ⇨program which executes test cases by sending, receiving and comparing primitives to/from the protocol stack
  - ⇨TAPCaller provides GUI interface
- PCO (Point of Control and Observation):
  - ⇨tools cabable to stimulate the protocol stack and display traces and duplicated primitives
  - ⇨used for logging and replaying test sessions
- xPanel (eXtended Panel):
  - ⇨„virtual mobile“

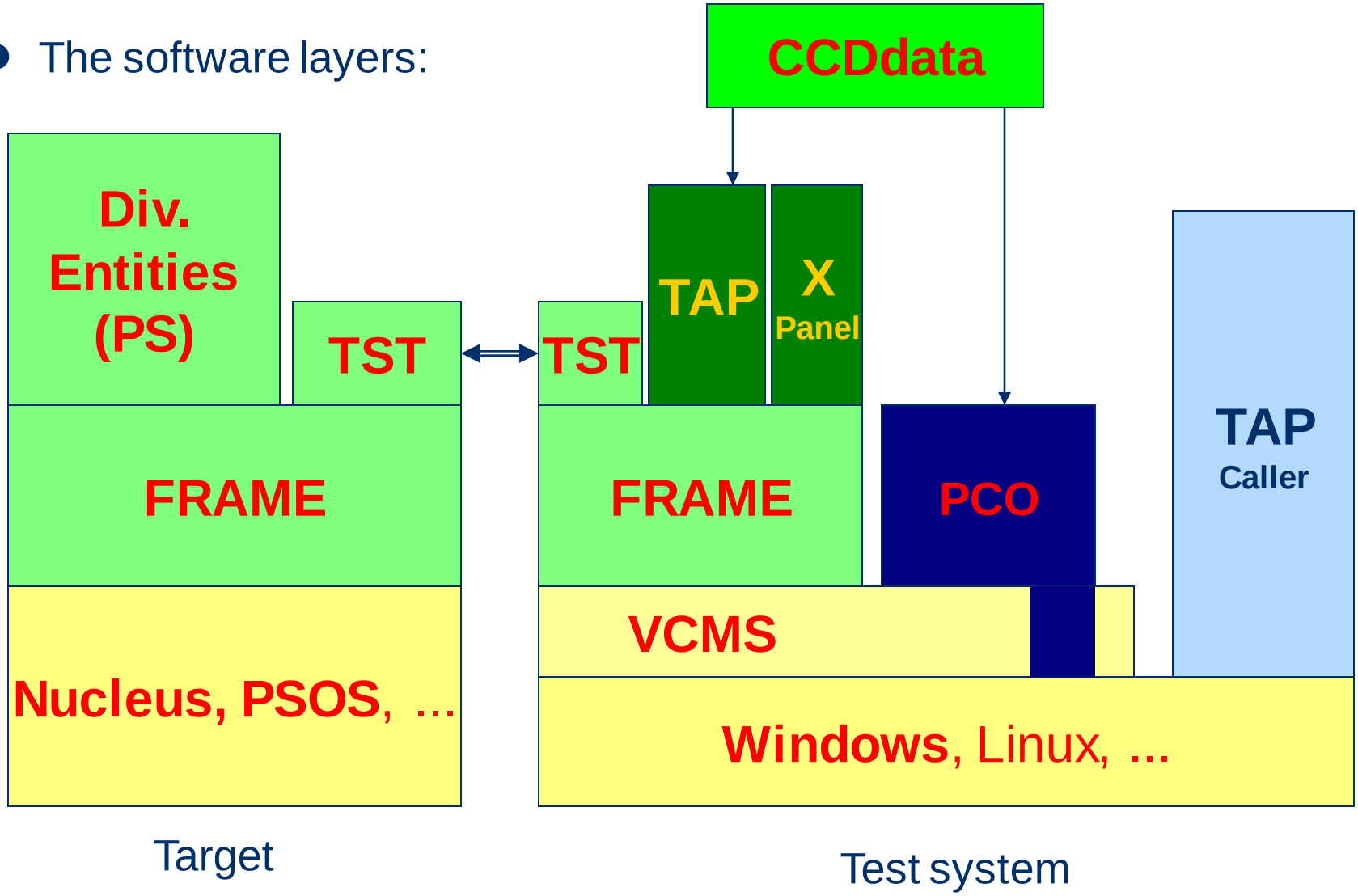
TC-Gen … *testing concept overview*

- The software layers:



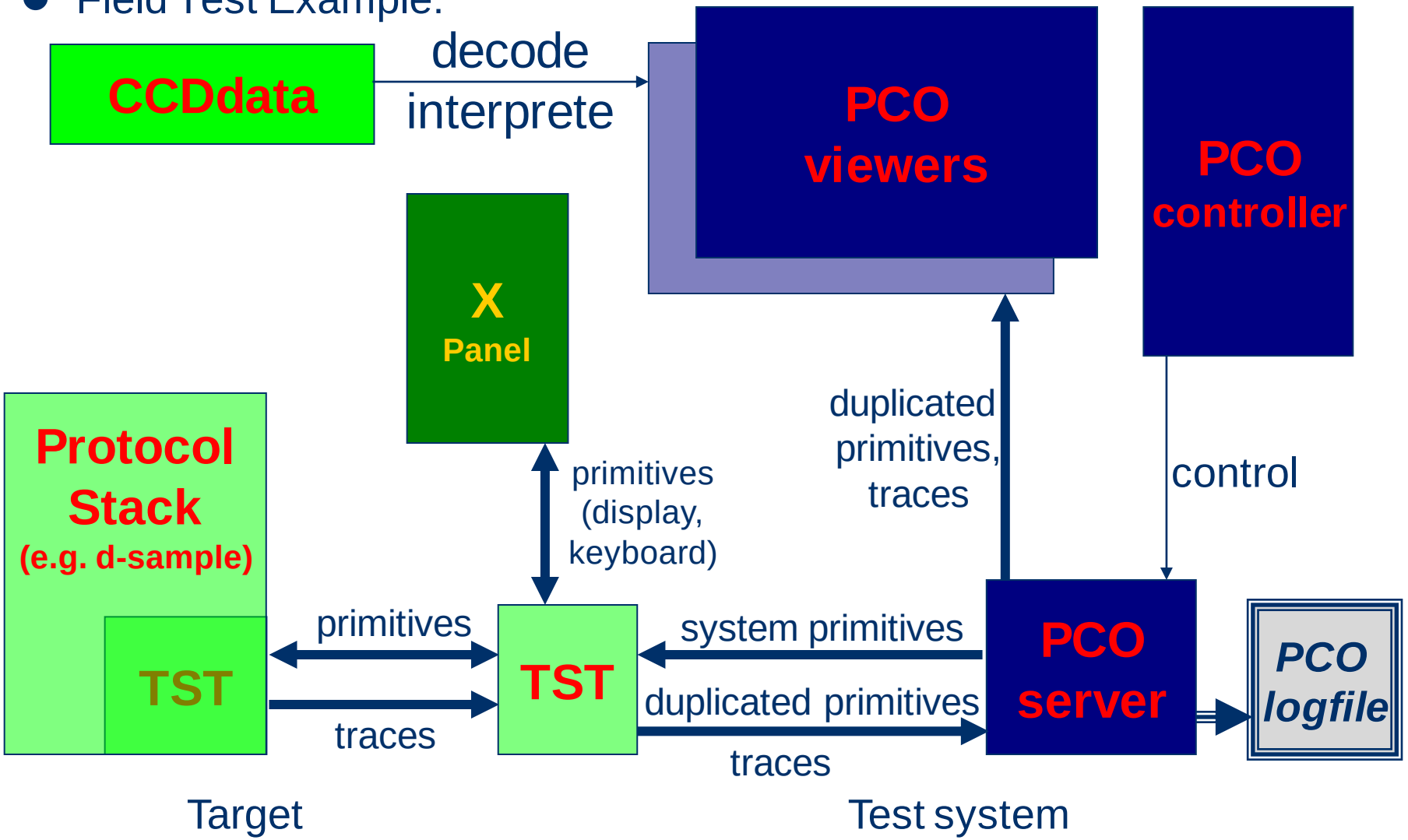
TC-Gen … *testing concept overview*

- The software layers:



TC-Gen … *testing concept overview*

- Field Test Example:

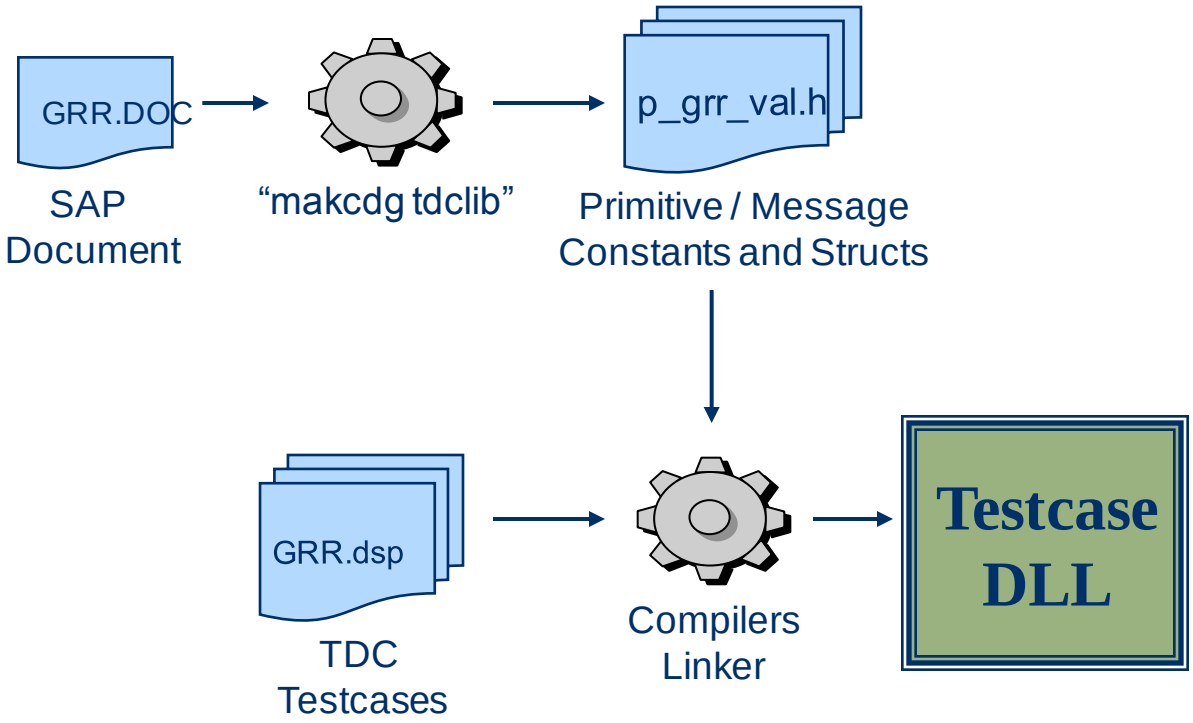


TC-Gen … *testing concept overview*

- TAP-Test Example (test case build):

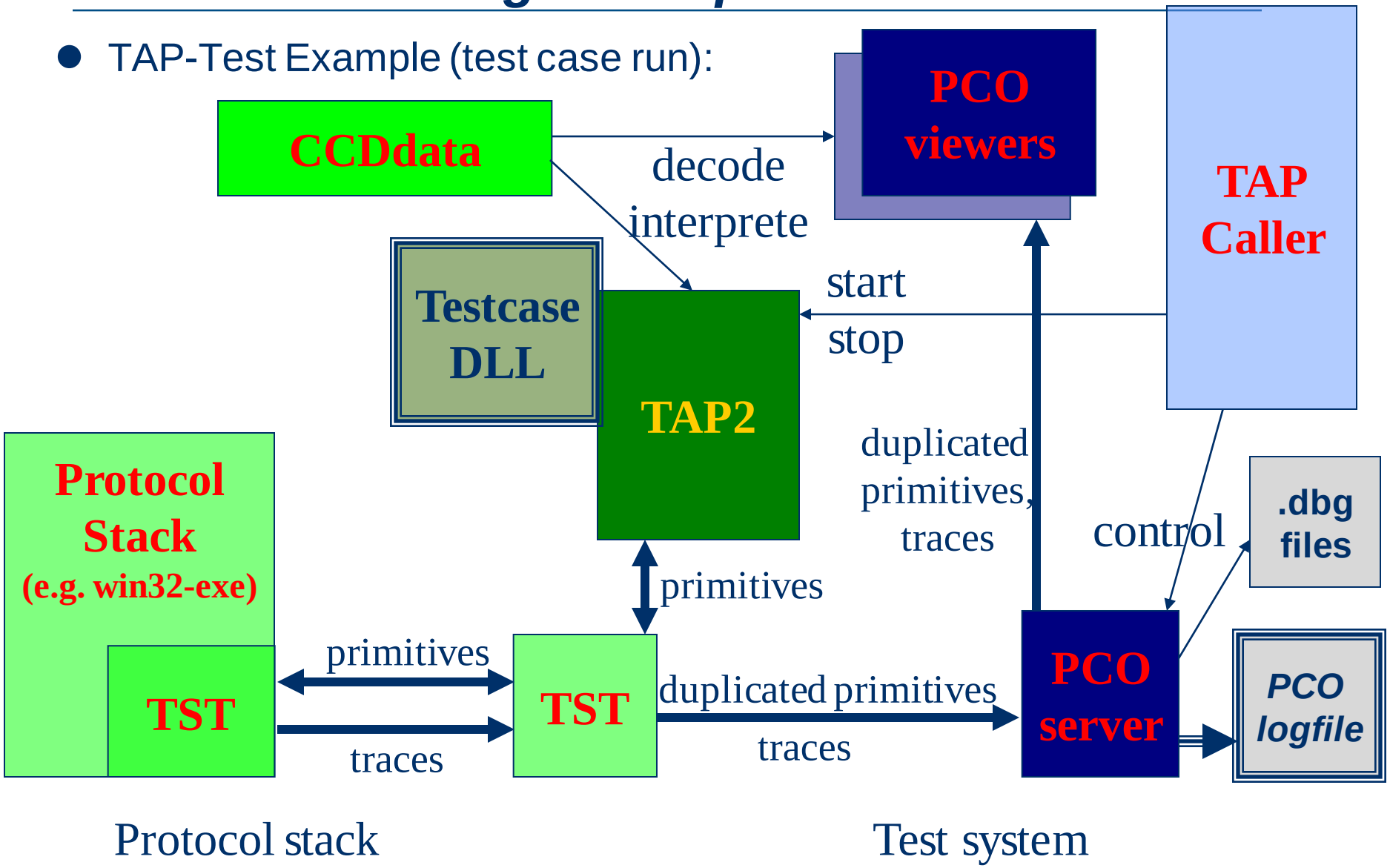
⇨ Generators
- doc2txt
- xgen
- ccdgen

⇨ Compilers
- msdev / cl
- (mktc)



TC-Gen … *testing concept overview*

- TAP-Test Example (test case run):



TC-Gen … *testing concept overview*

- What is a PCO-logfile anyway ?
  - ⇨ contains all traces and primitives received during a time period of a testsession (e.g. a field test)
    - ... defined by starting/stopping logging
  - ⇨ the format is binary but with linebreaks after each primitive
    - ... and can be interpreted by PCO-server only
  - ⇨ any kind of PCO-viewer may request (portions of) it
    - ... for replaying or other tasks

TC-Gen … *testing concept overview*

- For more information see:

  ⇨\gpf\DOC\testplatform_userguide.doc
  ⇨\gpf\DOC\tdc\tdc_intro.ppt
  ⇨\gpf\DOC\pco\pco_intro.ppt
  ⇨\gpf\DOC\tapcaller\tapcaller_intro.ppt

  ⇨... and many other documents in \gpf\DOC !

TC-Gen …

Testing Concept Overview

Motivation

Usage

How it works

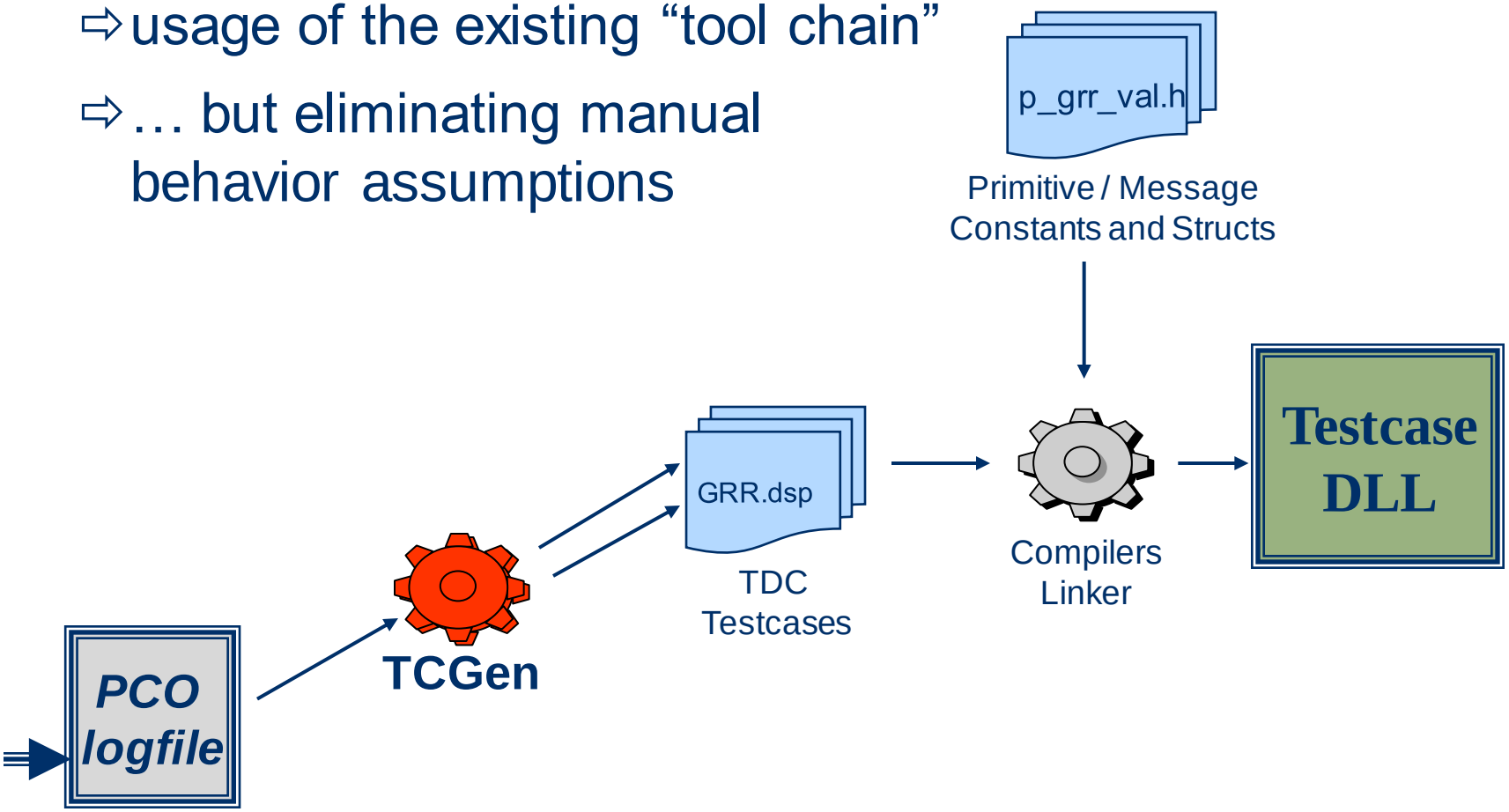
Current Limitations & Outlook

TC-Gen … *motivation*

- Why do we need another generator ?
  - ⇨ current generators (like tdsgen) don‘t actually generate
    - ... but translate e.g. Word documents into TDS-files
    - (and anyway TDS will not be used anymore)
  - ⇨ the actual test case has to be written manually
    - ... requiring detailed knowledge of the expected behavior
  - ⇨ with an automated generator taking the real and up-to-date primitive flow into account ...
    - new test cases could be easily created
    - buggy behavior could be easily reproduced

TC-Gen … *motivation*

- The Idea:
  - ⇨usage of the existing “tool chain”
  - ⇨… but eliminating manual behavior assumptions



TC-Gen …

Testing Concept Overview

Motivation

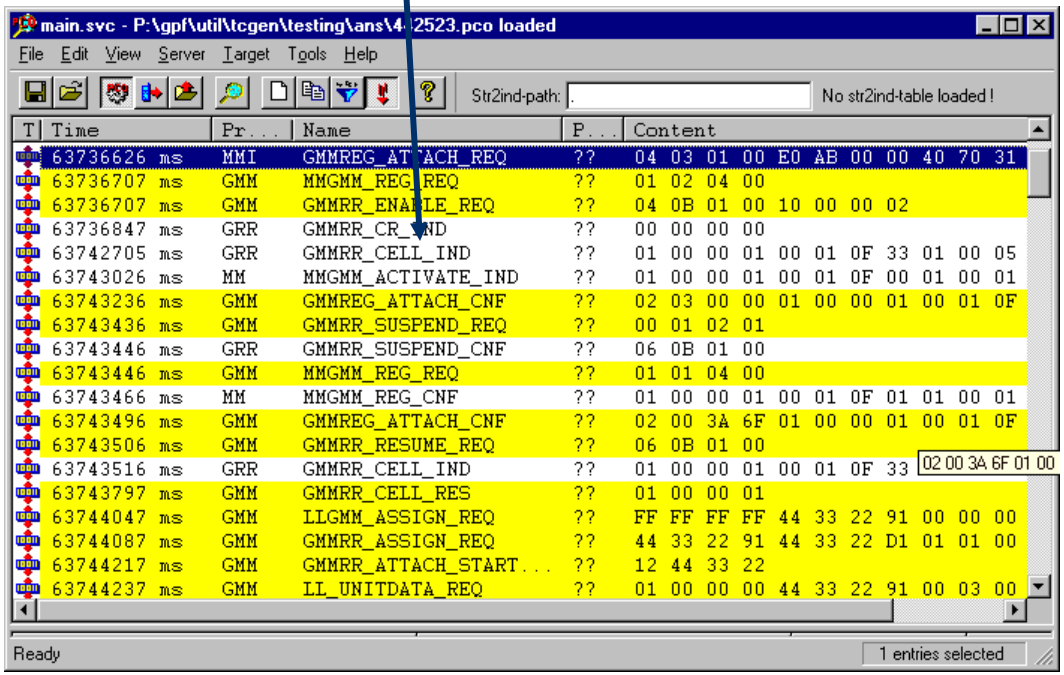
Usage

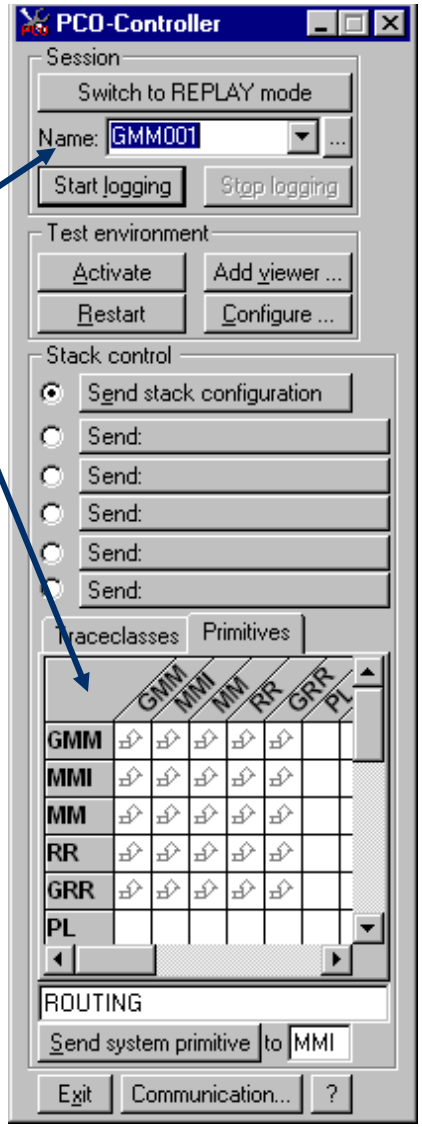
How it works

Current Limitations & Outlook

TC-Gen … *usage*

- At first we need a PCO-logfile:
  - ⇨ request duplication of primitives out of PS
    (currently manually, upcoming ccddata features will enable more comfortable selecting – „Matrix Reloaded“ :-)
  - ⇨ and start logging into a specified .pco-file
  - ⇨ optionally watch arriving primitives





TC-Gen … *usage*

- Second we use TCGen with the PCO-logfile:
  - ⇨ start the PCO-server (if not already running)

  - ⇨ call TCGen:

```
usage:
tcgen -h | -ver | [-i <ini-file>] [-wait] <pco-input-file> {-analyse | <output-d
ir> <entities> [-n <testcase-name>] [-p <project>]}
example: tcgen mmgmm001.pco c:/tdc_testing/mmgmm MM,GMM -p mmgmm -n MMGMM023
```

  - ⇨ ok, call it with appropriate parameters ☺
    (library paths and more parameters are taken from the tcgen.ini file)

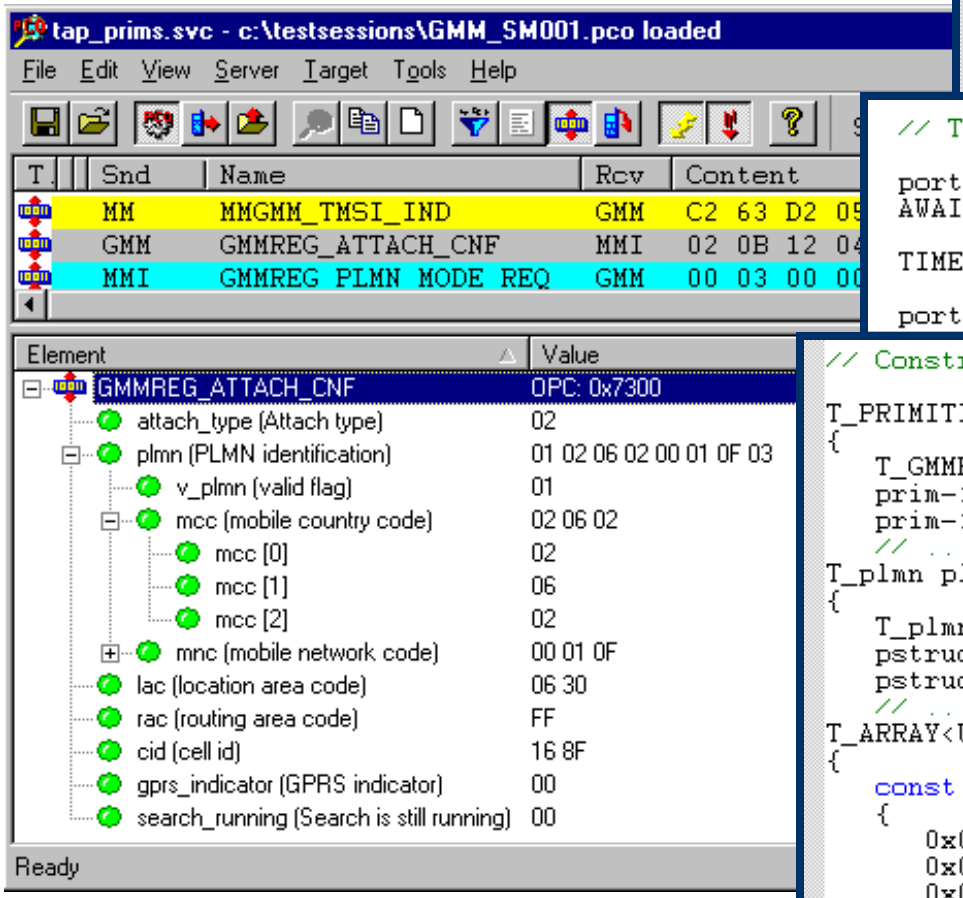
```
[P:\gpf\BIN]tcgen c:\testsessions\GMM_SM001.pco \g23m\condat\ms\test\dsample GMM
,SM -p gmmsm -n GMMSM028
sourcing P:\gpf\BIN/../cfg/tcgen.ini ...
sourcing P:\gpf\BIN/../cfg/pco.ini ...
generating testcase ...
... generation succeeded !
-> See "\g23m\condat\ms\test\dsample" for results.
```

  - ⇨ evtl. inspect the generated files in the VisualStudio
    (if the project already contained cases the new stuff will be merged at the beginning of all files)
  - ⇨ … or just build the testcase DLL from command line:

```
[P:\gpf\BIN]msdev \g23m\condat\ms\test\dsample/gmmsm_test.dsp /MAKE
Building default target: gmmsm_test - Win32 Debug
```

TC-Gen … *usage*

- A look at the results:



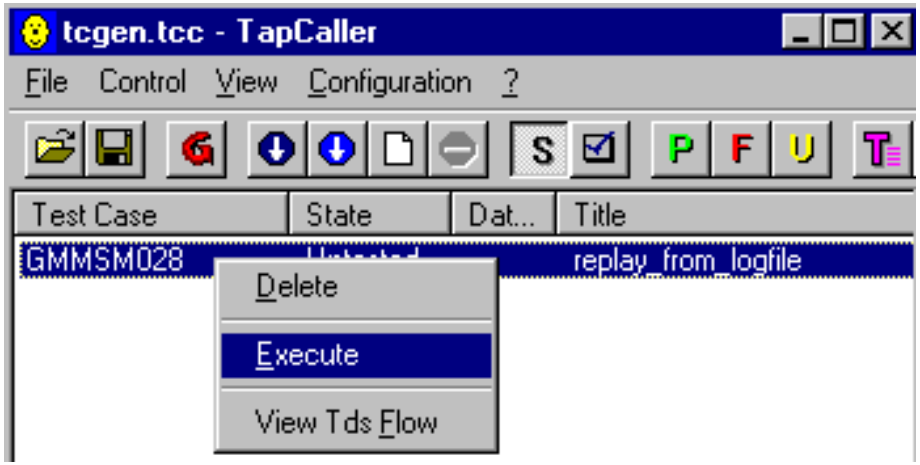

```
// Test Cases: gmmsm_cases.cpp

T_CASE GMMSM028()
{
   BEGIN_CASE ("replay_from_logfile")
   {
      setup_routings_for_GMMSM028();
      run_logged_stuff_from_GMMSM028();
```

```
// Test Step Excerpt: gmmsm_steps.cpp

port2GMM.SEND (mmgmm_tmsi_ind_GMMSM028_8());
AWAIT (gmmreg_attach_cnf_GMMSM028_9());

TIMEOUT(300);

port2GMM.SEND (gmmreg_plmn_mode_req_GMMSM028_10());
```

```
// Constraints Excerpt: gmmsm_constraints.cpp

T_PRIMITIVE_UNION gmmreg_attach_cnf_GMMSM028_9()
{
   T_GMMREG_ATTACH_CNF prim;
   prim->attach_type= 0x02;
   prim->plmn= plmn_GMMSM028_6();
   // ...
T_plmn plmn_GMMSM028_6()
{
   T_plmn pstruct;
   pstruct->v_plmn= 0x01;
   pstruct->mcc= mcc_array_GMMSM028_7();
   // ...
T_ARRAY<U8> mcc_array_GMMSM028_7()
{
   const U8 array_elements[3]=
   {
      0x02,
      0x06,
      0x02
   };
   return T_ARRAY<U8>(array_elements);
} // mcc_array_GMMSM028_7;
```

TC-Gen … *usage*

- Now what to do with it ?
  - ⇨ you might use TAP and run the fresh test case with a Windows-PS



  - ⇨ you might edit the TDC files and adapt them to your special needs
    - e.g. change „online parameters“ like IMSI
    - unfortunately manually by using the VisualStudio
    - … due to the lack of a dedicated „test case editor“
    - of course the beloved perl scripts can do nice work, too ☺
  - ⇨ … then rebuild the test case and use TAP again

TC-Gen …

Testing Concept Overview

Motivation

Usage

How it works

Current Limitations & Outlook

TC-Gen … *how it works*

- TCGen is implemented as a PCO-Viewer …
  - ⇨ by internally deriving from viewer core C++ classes
  - ⇨ which enables it to easily request all primitives from a specified PCO-logfile
  - ⇨ ... as often as it wants to
- In a first pass ...
  - ⇨ all necessary routings are generated in TDC
    - using information about the primitive sender and receiver
    - ... and the SAP name, currently taken from primitive name
  - ⇨ all declarations of constants, structs a.s.o.
    - are written to the TDC header files

TC-Gen … *how it works*

- Many further passes …
  - ⇨ look inside the primitive hierarchy level by level
  - ⇨ generate:
    - SEND() / AWAIT() commands for primitives
    - parameter values and sub-structures
    - TIMEOUT() commands for delays
  - ⇨ ... and finally the whole test case

- More implementational details …
  - ⇨ will be documented in \gpf\DOC\tcgen\tcgen_description.doc
  - ⇨ can be found out from the sources in \gpf\util\tcgen\... ☺

TC-Gen …

Testing Concept Overview

Motivation

Usage

How it works

Current Limitations & Outlook

TC-Gen … *current limitations & outlook*

- Unfortunately in many cases the pure generated test case will fail due to:
  - ⇨ a lot of parameter values coming from network
    - ... and with Windows-PS we don‘t have that
  - ⇨ sometimes missing initial primitives
    - e.g. with BMI-image the first SIM_GMM_INSERT_IND gets lost
  - ⇨ several parameters which could be ignored
    - ... but which are expected with exactly the recorded value
- *A quite funny side effect:*
  - ⇨ *if the PCO-logfile comes from a TAP run which failed*
  - ⇨ *… the new generated test case will pass ☺*

TC-Gen … *current limitations & outlook*

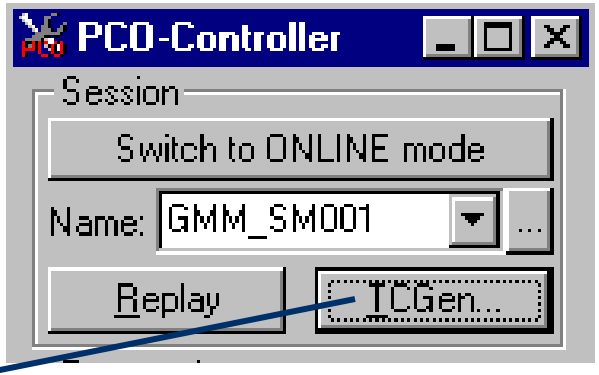
- Further problems are …
  - ⇨ quit long time differences between primitives
    - … which are not necessary with the Windows-PS
    - but a maximal timeout value can be defined
  - ⇨ primitive duplication does not work with all entities by now
    - because of changed timing
    - e.g. for LLC the PS will crash
  - ⇨ currently no support of duplication of pointers by the FRAME
    - e.g. for descriptor lists
- And finally …
  - ⇨ air-messages are currently handled as binary blocks
  - ⇨ no support of other formats then TDC
    - but TTCN-3 should be no problem
  - ⇨ no complete GUI exists
    - allowing execution of all processes from one place

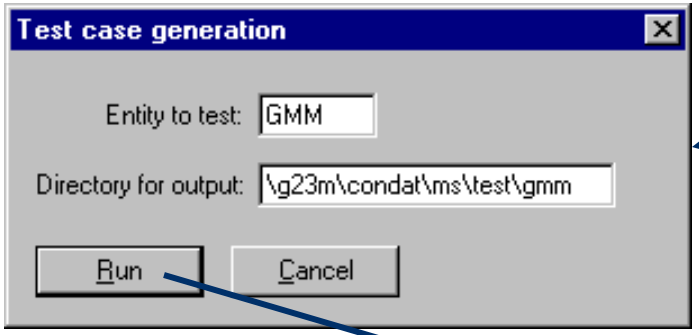
TC-Gen … *current limitations & outlook*

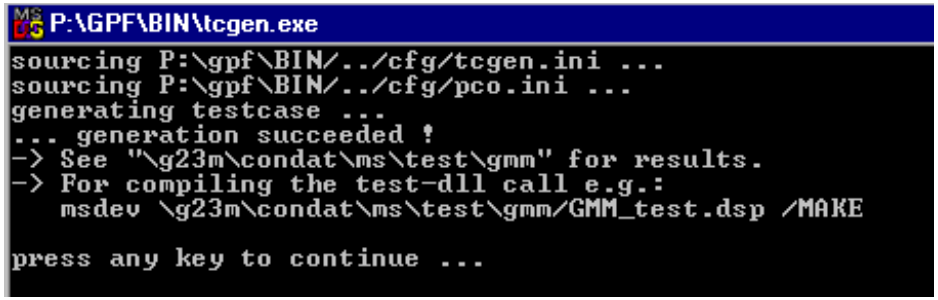
- Some solutions seem obvious:
  - ⇨ the generated TDC files could be post processed to ...
    - set parameters to „not used“ etc.
  - ⇨ the TST queue on target could be temporarily enlarged
    - to reduce timing influences
  - ⇨ the support of pointer duplication could be implemented
    - e.g. similarly as done with PCON on UMTS stacks

TC-Gen … *current limitations & outlook*

- Concerning the GUI support …
  - ⇨ a first step is already done: TCGen can be started right from the PCO-controller







TC-Gen … *current limitations & outlook*

- TCGen could/will be enhanced to support:
  - ⇨ interpretation of air messages using ccddata-DLL
  - ⇨ upcoming ccddata features to find out redirections independent from „random“ primitives in logfile
  - ⇨ the option to generate primitives with direct parameter passing
  - ⇨ configuring the level of structure expanding
  - ⇨ rules specified in an extra file and applied during generation
    - to e.g. set/ignore special parameter values
  - ⇨ a dedicated GUI for convenient generation
- ... after all TCGen is still in Beta state ☺

Thanx for your patience !