SIEMENS

From Testing to Anti-Product Development

SIEMENS

The classical SW Developer ...

Sees the

- Definition of system architecture as most creative, demanding and satisfying
- Anticipation of future requirements as vital for the design of a robust, maintainable and extendable system
- Detailed level design as a creative act, even the lowest level of coding

SIEMENS

… now Relegated to Tester?

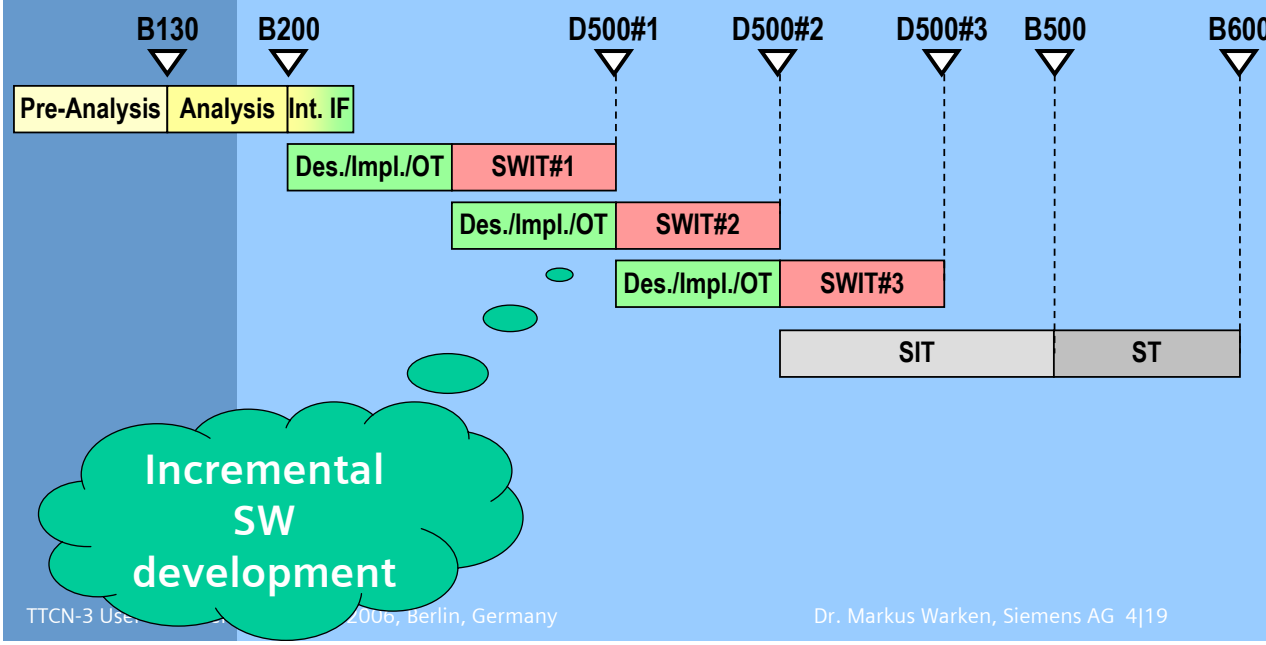
For the "classical developer" testing

- Is not considered creative at all
- Is an accepted necessity
- The focus of testing is predominantly fault finding
- Is in short: an annoying nuisance but unfortunately inevitable

Are we now relegated to testers?

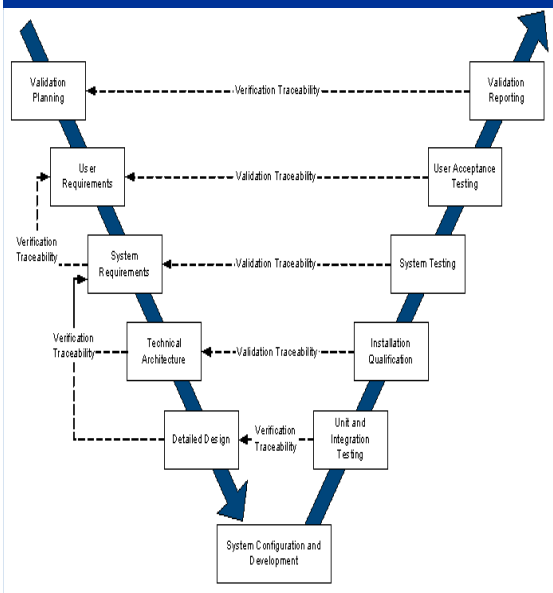
SIEMENS

A typical master plan these days ….



SIEMENS

Activities of a test team during the development cycle



New requirements for every development iteration trigger

- **Identification of test cases – analysis**
- **Specification of test cases – design**
- **Implementation of test cases – coding**
- **Execution of test cases – testing**
  - **Initial (manual) testing of new TCs**
  - **Continuous regression**

SIEMENS

Test teams and Implementation teams have ....

- The same complexity of requirements
- More or less equivalent tasks executed in parallel
- Similar team sizes and overall efforts

→ test teams develop the anti product in parallel

SIEMENS

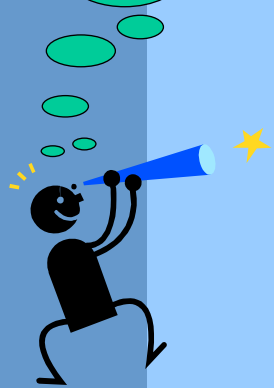
Iterative development = continuous flow process

SIEMENS

construction kit of TC parts

In an ideal world such an anti product features ....

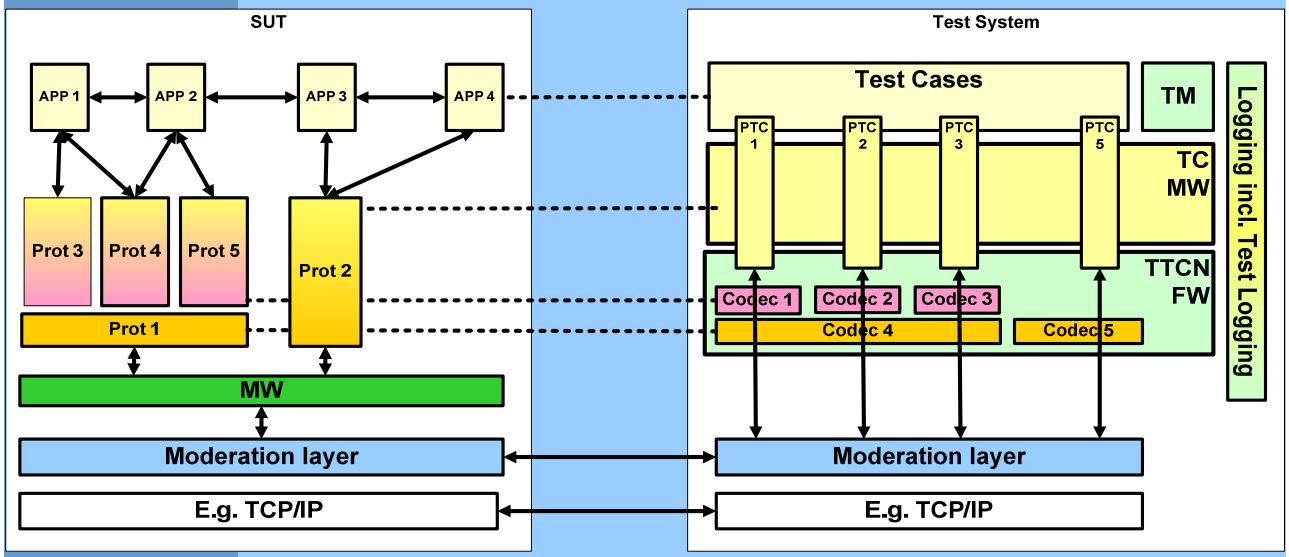
... just push the button



- Easy implementation of messages
- Flexible Implementation of flows and scenarios
- Automation
- Multi tasking
- High re-use factor
- Modularity for a divide-and-conquer approach to govern the complexity
- ...

SIEMENS

TTCN-3 – the silver bullet?



SIEMENS

TTCN-3 – the programming language for anti-products?

- TTCN-3 is from its core architecture merely a structured programming language like C or Pascal
- TTCN-3 has a number of test specific features like the template concept
- TTCN-3 has some extensions towards object orientation like inheritance e.g. for components
- "base classes" like test case, MTC, PTC, operators like send, receive, ...
- TTCN-3 has features like grouping ports in components that can be seen as a basic middle ware

SIEMENS

TTCN-3 – the programming language for anti products? (cntd)

- TTCN-3 is currently the method of choice for the implementation of messages and test case logic
- this is then compiled into a run time system adding SA, PA, …
- per construction a mixture of programming languages for the anti product

- TTCN-3 based test systems contain large parts written in "native programming languages" and does not try to re-invent the wheel

SIEMENS

Automation is crucial …

… and TCI-TM interface for TTCN-3 test management provides all necessary operations but

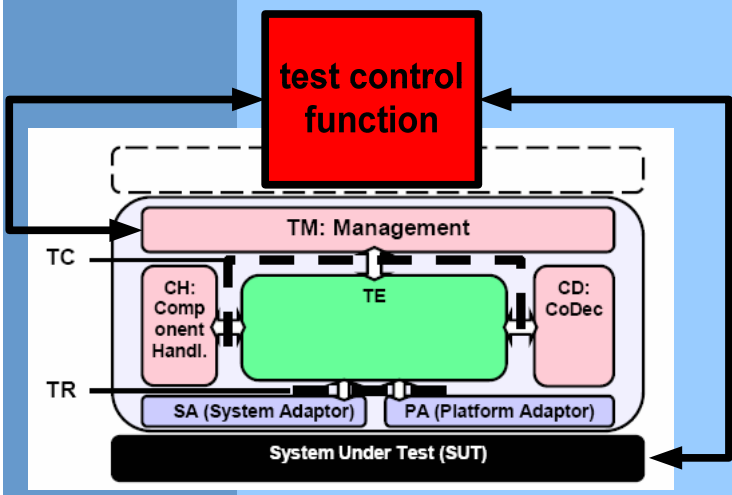
- automated test execution requires defined states of SUT AND TS at every point of time
- how to deal with failed TCs or runtime errors of the TS?

beyond execution automation there are also other aspects like

- (semi-) automated updates of test suites to changed interfaces,
- compilation of (parts of) the test system,
- conversion of interface definitions to TTCN-3 code
- …

SIEMENS

Test Management and Test Bed Control



a test bed control function takes the place of the test system user in many ways like
- running tests
- evaluate and store logs for improved verdicts
- re-initialise SUT after failed TCs
- re-initialise the test system if necessary

most can be done via TM
but the last item suggests an external function

SIEMENS

Re-use and maintainability of Test Suites

TTCN-3 allows a high re-use of TC fragments e.g. in subsequent test phases
TTCN-3 supports modularity for a divide and conquer approach to govern complexity
parameterised templates is an easy way to handle similar messages
module parameters allow to combine several similar TC fragments to a single piece of TTCN-3 code

but be careful!
- testers often have no strong programming skills
- testers often do not resist the copy-and-paste seduction → enforce the single source principle

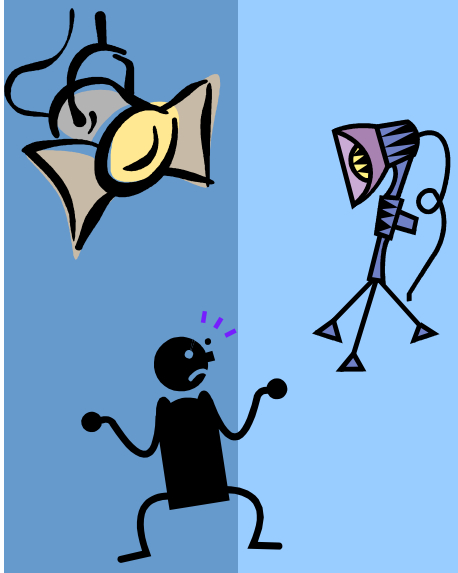
SIEMENS

Expert Roles in the Test Team

- the domain know-how is predominantly in the TTCN-3 code
- about 60-75% of the code volume of the test system is TTCN-3 code, for late release more
- experts for different functions in SUT to ease debugging might be a good idea
- sub-team for TM, TL, maybe a separate team for automation
- sub-team for interfacing the SUT, i.e. codecs, PA, SA

SIEMENS

Easily in the lime light



Test teams come after the development teams in the food chain

Significant problems in the testing phases ensure a high management attention ....

SIEMENS

Crucial Questions ...

Testing is meant to prove the required functionality of the product, but ....

Do you expect the test system to have a lower fault density than the SUT?

How many quality measures have the different SUT components undergone when it comes to test execution?

Who is right – SUT or TS?

SIEMENS

Test the Test System!

Make sure that you have done your homework when test execution starts!

- Use the same development process as for SUT
- Focus on quality control at the milestones, reviews, module testing etc → stabilise the TS as much as possible before the execution starts
- TTCN-3 is „just" a programming language and the anti product „just a SW system"

You not want to pull the short straw!

SIEMENS

Conclusions

TTCN-3 is the current method of choice for the implementation of messages and test case logic of an automated test system

TTCN-3 allows a high re-use but this requires a clever test system design

TTCN-3 provides standardised interfaces to integrate the test cases to a complete test system

Apply state of the art SW development processes

Test the test system before it comes to test execution!