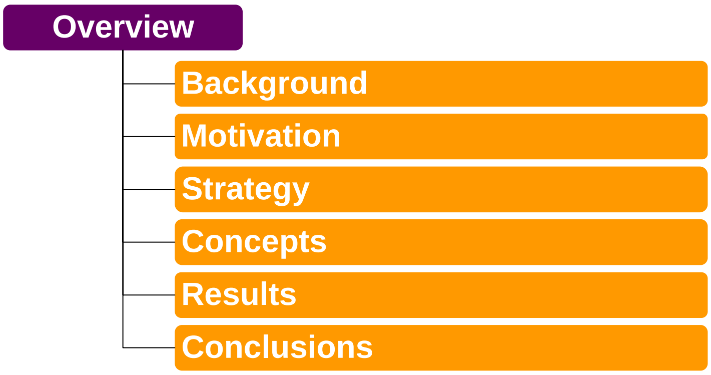
Validation of a TTCN-3 test system

Andy Rauland

Vikas Pratap Singh, Dr. Andreas Ulrich, Zhang Chao

- TTCN-3 User Conference 2007 -

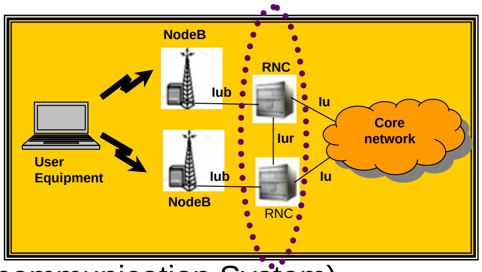
Overview



Overview

- Background
- Motivation
- Strategy
- Concepts
- Results
- Conclusions

Background



UMTS (Universal Mobile Telecommunication System)

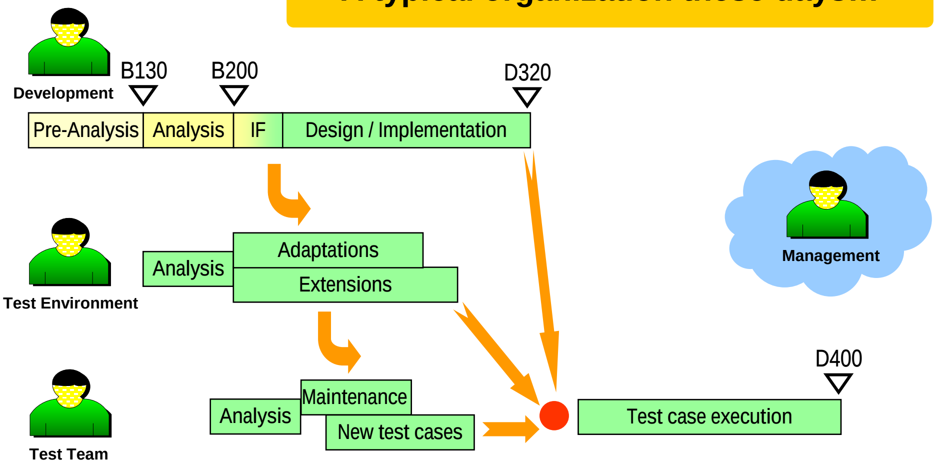
- 3rd generation mobile communication standard
- Provided by a standardization board at 3rd Generation Partnership Project (3GPP), that is a collaboration agreement in-between industrial partners
- UMTS provides mobile broadband access over the air interface for voice and data services

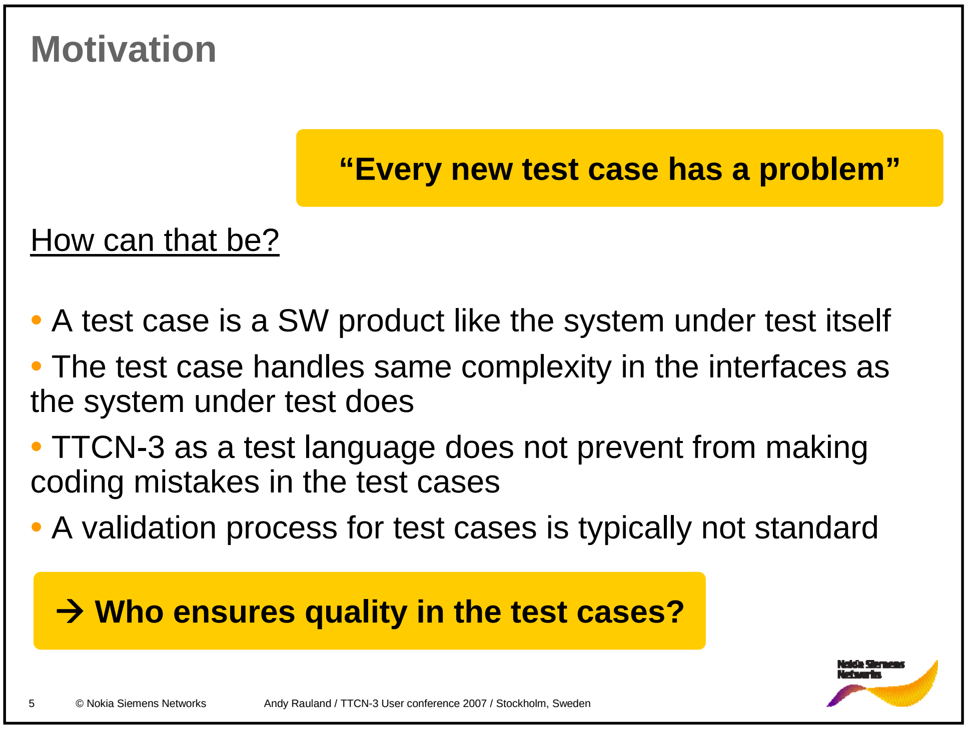
RNC (Radio Network Controller)

- Is an UMTS network element that owns and controls the radio resources available at UMTS Base stations (NodeB)
- The RNC handles well defined protocol exchanges over Iu, Iur and Iub interfaces and applies specified procedures to perform power control, initiating handover between cells, etc.

Motivation

A typical organization these days…





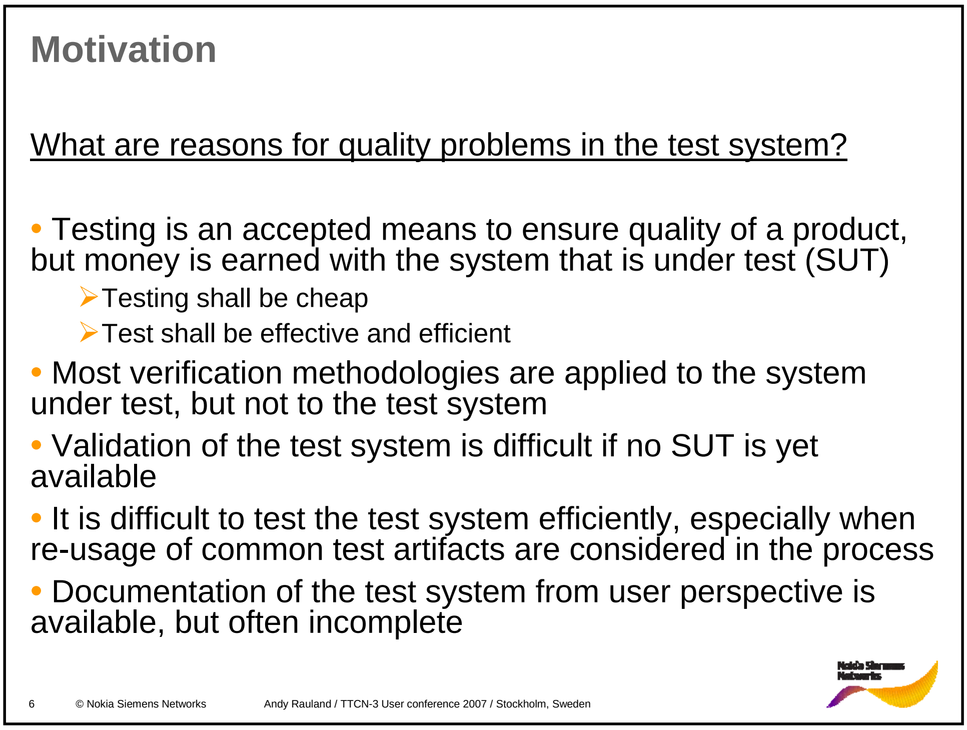
Motivation

“Every new test case has a problem”

How can that be?

- A test case is a SW product like the system under test itself
- The test case handles same complexity in the interfaces as the system under test does
- TTCN-3 as a test language does not prevent from making coding mistakes in the test cases
- A validation process for test cases is typically not standard

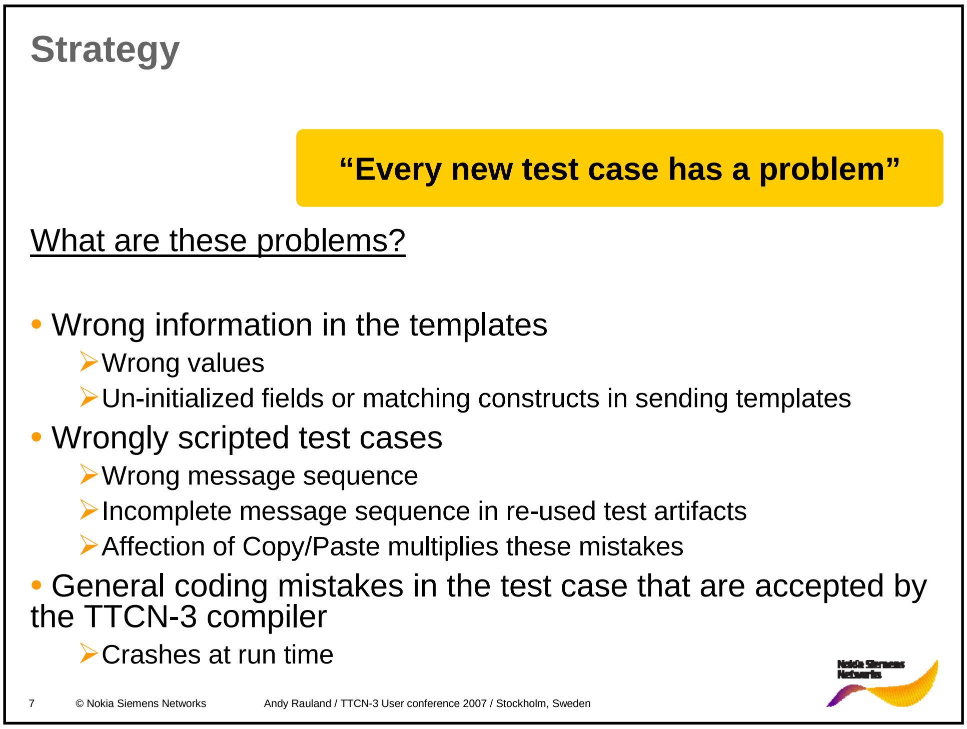
→ Who ensures quality in the test cases?



Motivation

What are reasons for quality problems in the test system?

- Testing is an accepted means to ensure quality of a product, but money is earned with the system that is under test (SUT)
  - ➢ Testing shall be cheap
  - ➢ Test shall be effective and efficient
- Most verification methodologies are applied to the system under test, but not to the test system
- Validation of the test system is difficult if no SUT is yet available
- It is difficult to test the test system efficiently, especially when re-usage of common test artifacts are considered in the process
- Documentation of the test system from user perspective is available, but often incomplete

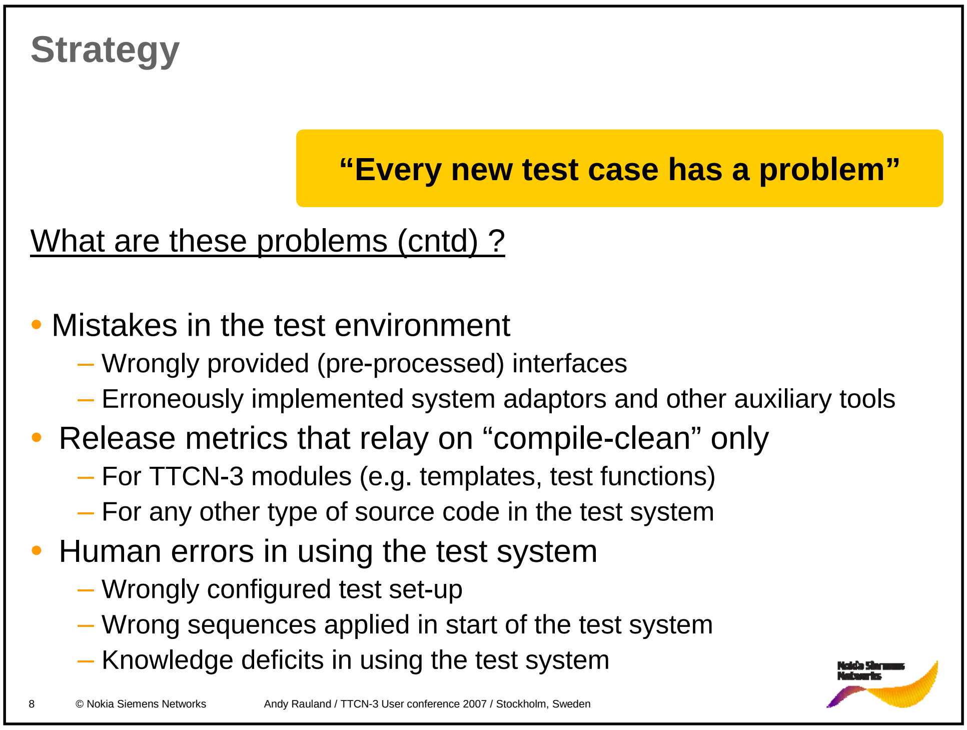


Strategy

“Every new test case has a problem”

What are these problems?

- Wrong information in the templates
  - ➢ Wrong values
  - ➢ Un-initialized fields or matching constructs in sending templates
- Wrongly scripted test cases
  - ➢ Wrong message sequence
  - ➢ Incomplete message sequence in re-used test artifacts
  - ➢ Affection of Copy/Paste multiplies these mistakes
- General coding mistakes in the test case that are accepted by the TTCN-3 compiler
  - ➢ Crashes at run time



Strategy

“Every new test case has a problem”

What are these problems (cntd) ?

- Mistakes in the test environment
  - – Wrongly provided (pre-processed) interfaces
  - – Erroneously implemented system adaptors and other auxiliary tools
- Release metrics that relay on “compile-clean” only
  - – For TTCN-3 modules (e.g. templates, test functions)
  - – For any other type of source code in the test system
- Human errors in using the test system
  - – Wrongly configured test set-up
  - – Wrong sequences applied in start of the test system
  - – Knowledge deficits in using the test system

Strategy

→ Introduction of quality ensuring mechanisms in the test area

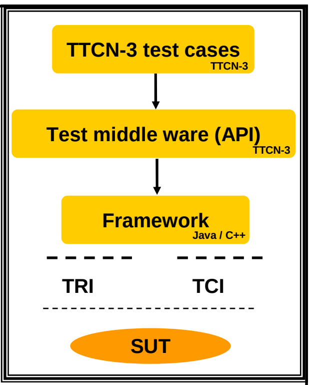
Test the test system, with Anti-Product concepts

For this:

- Simplify test case scripting and reduce complexity in the test cases to be written by introduction of a test middle-ware (API)
- Execute test case under purely simulated conditions
- Keep your test system always fully automated
- Force the documentation being always updated
- Ensure always that the test system simplifies testing

Concepts

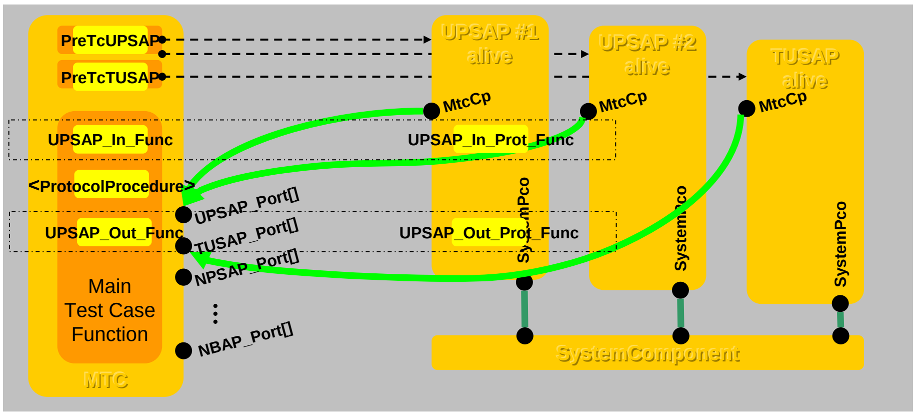
Introduction of a test middle ware (API)



- Software layer for TTCN-3 test scripts
- Reduces complexity of test cases
- Support mapping of system requirements modeled in MSCs to TTCN-3 functions
  - Support of alternatives & parallel MSC behaviors
- Support of Protocol Procedures from 3GPP standards
- Well documented user interface (API) with excellent usability
- Provided TTCN-3 artifacts (API) are well tested in advance and free of errors/warnings
- Optimized for compilation time
- Principal support for distributed test case execution

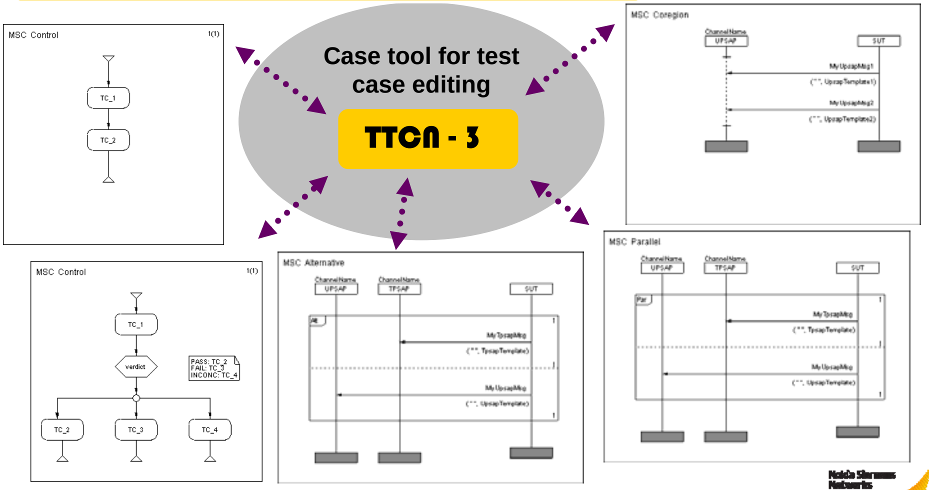
Concepts

Introduction of a test middle ware (API) **TcSkeletonV3**



Concepts

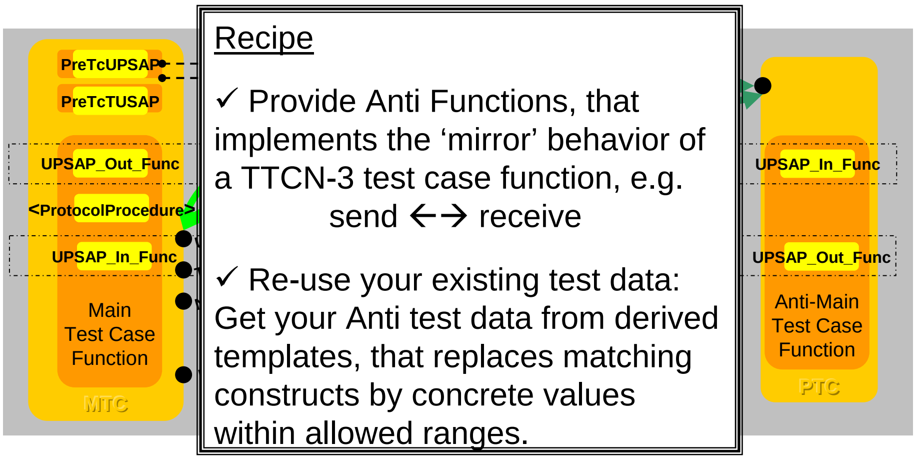
Introduction of a test middle ware (API) **TcSkeletonV3**


Case tool for test case editing

TTCN - 3

Concepts

Execute test case under purely simulated conditions



Recipe

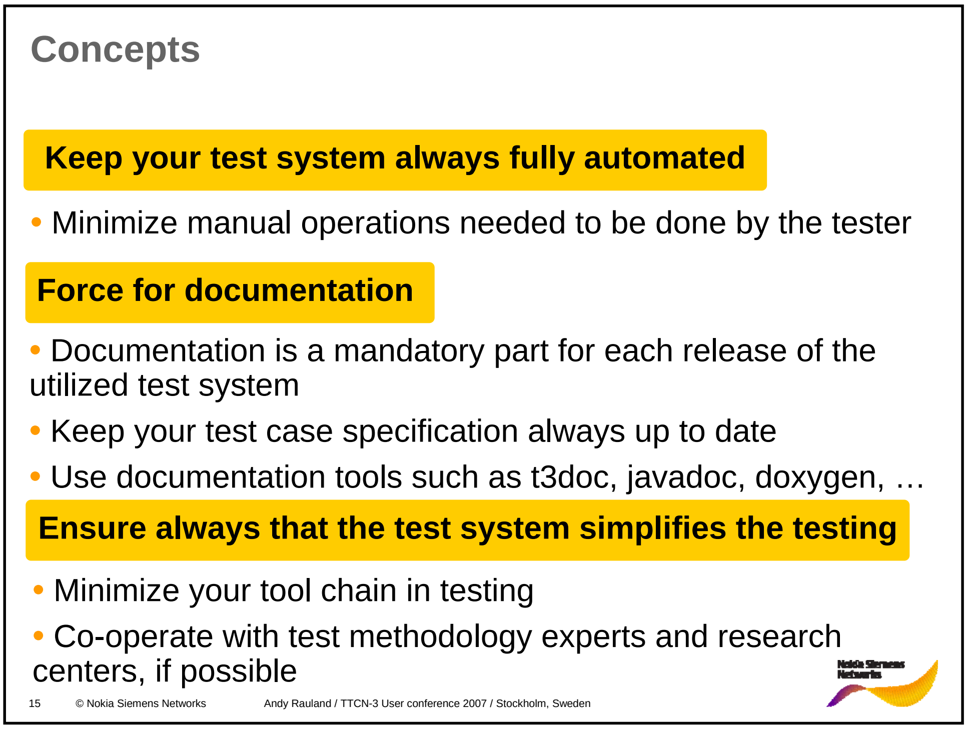
✓ Provide Anti Functions, that implements the 'mirror' behavior of a TTCN-3 test case function, e.g.
send ← → receive

✓ Re-use your existing test data: Get your Anti test data from derived templates, that replaces matching constructs by concrete values within allowed ranges.

Concepts

Apply unit test concepts for each test artifact

- For TRI, TCI:
  - Prove concepts with TTCN-3 test cases and a SUT that simulates the product under test. The SUT can be within or outside your TE
- For test middle ware (API):
  - Validate API with a set of TTCN-3 test cases running inside your TE
- For templates:
  - Generate dummy test cases around each template and execute them, to identify problems that are not detectable by the TTCN-3 analyzer.
  - Note: Problematic are templates that are parameterized and/or includes matching mechanisms
- For pre-processed interfaces:
  - Ensure that the templates still going through the CoDec at run-time
- For TTCN-3 functions acting as generic operations
  - Apply unit test concepts for each TTCN-3 function



Concepts

Keep your test system always fully automated

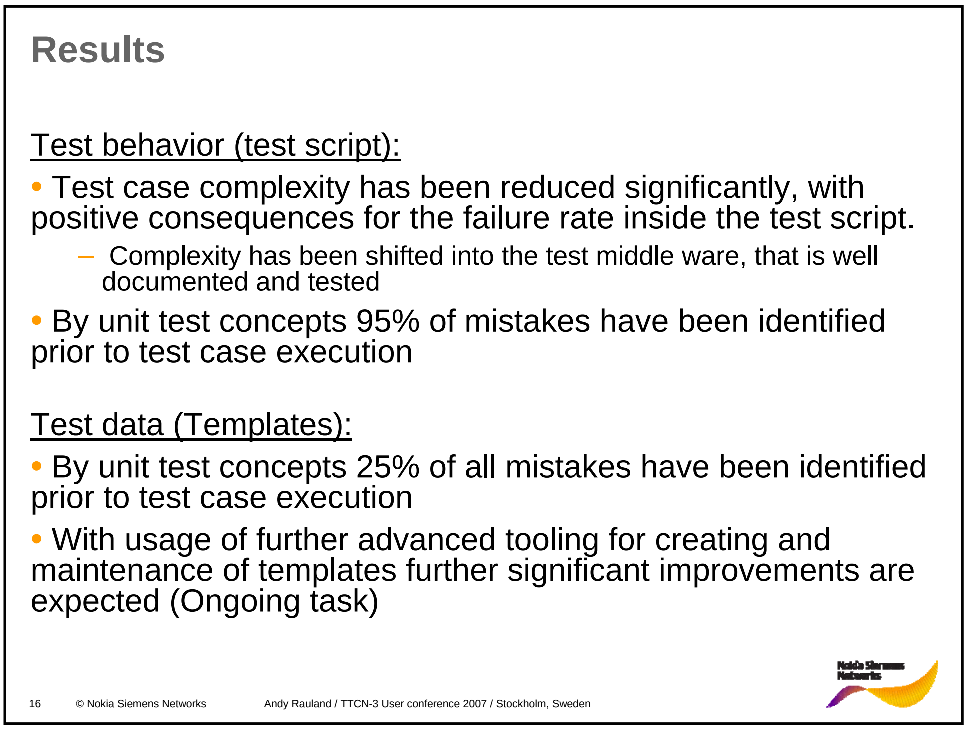
- Minimize manual operations needed to be done by the tester

Force for documentation

- Documentation is a mandatory part for each release of the utilized test system
- Keep your test case specification always up to date
- Use documentation tools such as t3doc, javadoc, doxygen, …

Ensure always that the test system simplifies the testing

- Minimize your tool chain in testing
- Co-operate with test methodology experts and research centers, if possible



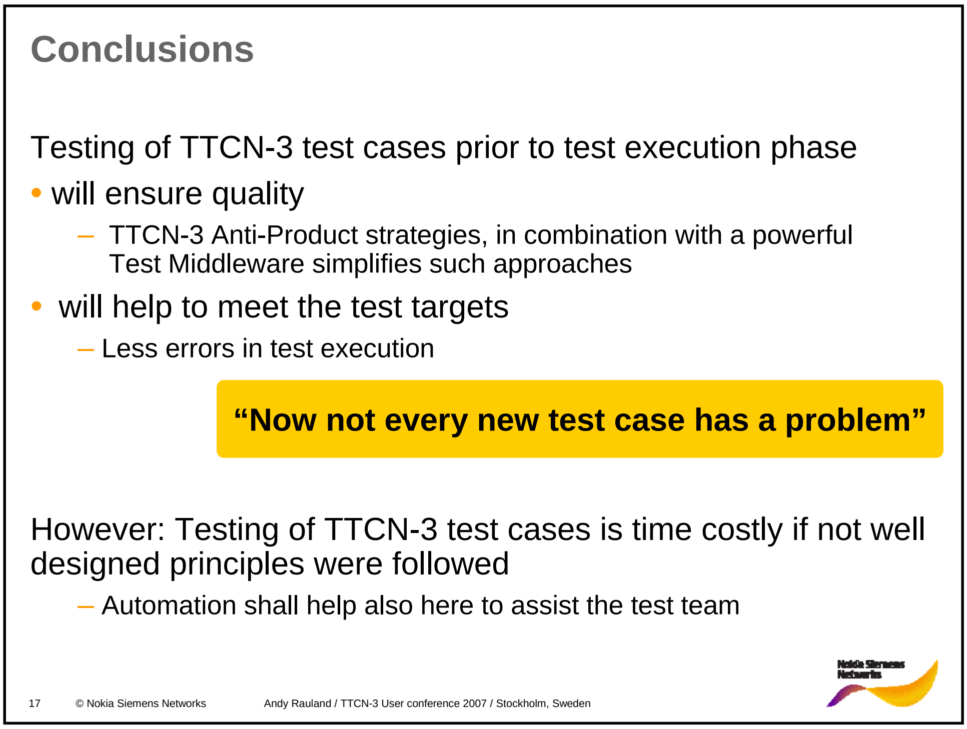
Results

Test behavior (test script):

- Test case complexity has been reduced significantly, with positive consequences for the failure rate inside the test script.
  - Complexity has been shifted into the test middle ware, that is well documented and tested
- By unit test concepts 95% of mistakes have been identified prior to test case execution

Test data (Templates):

- By unit test concepts 25% of all mistakes have been identified prior to test case execution
- With usage of further advanced tooling for creating and maintenance of templates further significant improvements are expected (Ongoing task)



Conclusions

Testing of TTCN-3 test cases prior to test execution phase

- will ensure quality
  - TTCN-3 Anti-Product strategies, in combination with a powerful Test Middleware simplifies such approaches
- will help to meet the test targets
  - Less errors in test execution

“Now not every new test case has a problem”

However: Testing of TTCN-3 test cases is time costly if not well designed principles were followed

- Automation shall help also here to assist the test team

Thank you for both…

…your participation and attention