

Experiences of design based testing with TTCN-3: Motivation, Configuration, and Results

Within an Ericsson design team of the GSM Base Station Controller, a need arose to improve the reliability and quality of the design based testing whilst also reducing the cost of maintaining an own test framework. A TTCN-3 based solution was piloted and subsequently deployed to all design components. TTCN-3 was chosen for the potential performance, scalability and support for code centric style of testing. A key requirement was to re-use the message types between the product and the test product. Additionally short build times were essential. Another key requirement for deployment was integration with the build system. Integration with run-time analysis tools then followed with focus on code coverage. This paper presents how TTCN-3 was used to improve the design based test environment with integration to the build system and run-time analysis tools. It also presents the performance metrics of the subsequent round-trip build process and where further improvements may be obtained from a continuous quality improvement process. The paper also details the positive and negative experiences gained in the pilot project.

Conor.White@ericsson.com



Key points

- The value of a standard scalable test framework in design based automated testing.
- Deployment characteristics of a SQA improvement in the design process.
- The value of combining a solid test framework with run-time analysis tools.

Definition of SQA

- **IEEE Definition 1.** "A planned & systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements"
- **IEEE Definition 2.** "A set of activities designed to evaluate the process by which products are developed or manufactured"
- **CMM Definition of purpose.** "The purpose of quality assurance is to provide management with appropriate visibility into the process being used by the software project and of the projects being built"
- **Ericsson's focus on SQA - Processes and activities that improve software quality supporting & delivering "operational excellence"**



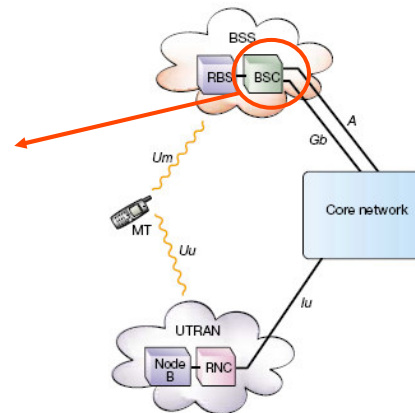
© Ericsson AB 2006

Experiences of Design Based Testing with TTCN-3

ERICSSON

GSM BSC

- Base Station Controller supports management of mobile voice and data traffic including handover between GSM and 3G networks

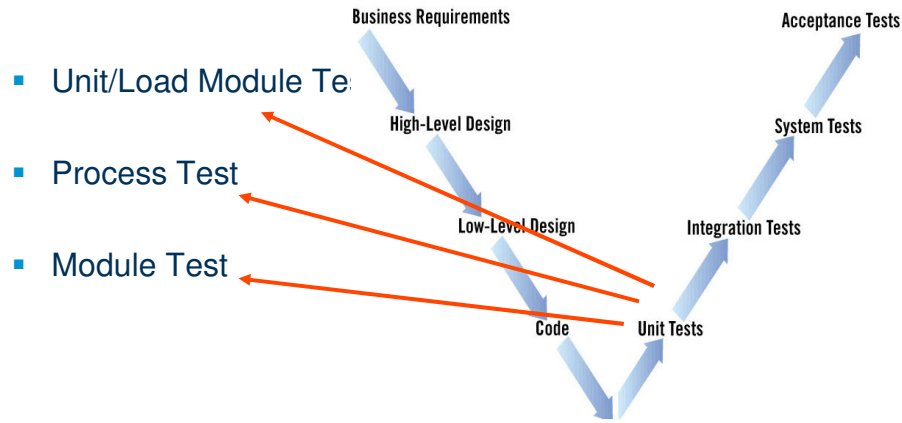


© Ericsson AB 2006

Experiences of Design Based Testing with TTCN-3

ERICSSON

Design based testing



© Ericsson AB 2006

Experiences of Design Based Testing with TTCN-3

ERICSSON

Need for Change

- Multiple test environments used in design based tests
 - Perl based test environment used with target trace tool
 - Un-supported/un-maintained trace tool
 - Unreliable performance and a costly proprietary environment maintained in-house
 - Process test instructions (PTI) designed in C and run together with the product code within simulator
 - No test independence, not scalable/re-usable
 - Costly environment maintained in-house
 - C isn't a test language therefore more coding required
 - Module test instructions (MTI) designed in C and compiled and run together on host

© Ericsson AB 2006

Experiences of Design Based Testing with TTCN-3

ERICSSON

Requirements

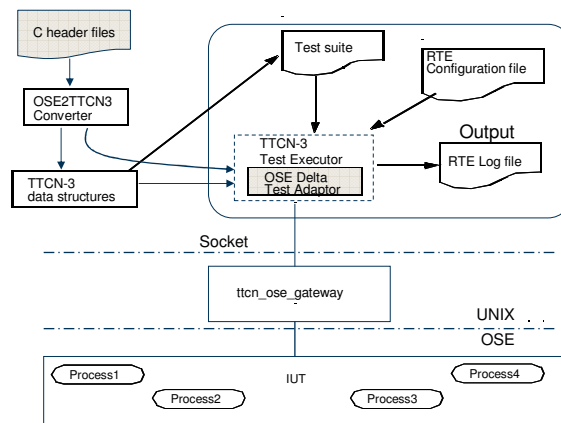
- Reliable, scalable, economic test automation framework for Load Module Test
- Solid test notation for flexible/efficient test coding
 - Support for real-time characteristics (un-predictable sequences)
- Fast compile and run-time (seconds NOT minutes)
- Integration with clearmake™ build system
- Integration with code coverage & memory analysis tools (PurifyPlus™, Test RealTime™) to isolate untested components

© Ericsson AB 2006

Experiences of Design Based Testing with TTCN-3

ERICSSON

Test System



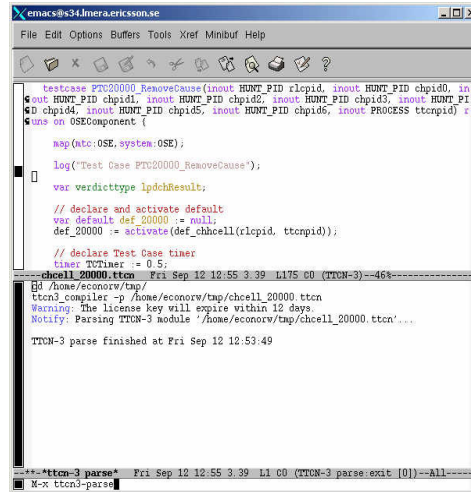
© Ericsson AB 2006

Experiences of Design Based Testing with TTCN-3

ERICSSON

Test Design

- Tools
 - Titan TTCN-3 tool suite (Ericsson)
 - Titan OSE Delta test port (Ericsson)
 - Emacs (GNU)
 - GNU emacs ttcn3 mode (Debian)
 - GNU C/C++ compiler
 - ClearCase (IBM)
 - OSE Soft Kernel (Enea)



```
emac@534.linaera.ericsson.se
File Edit Options Buffers Tools Xref Minibuf Help

testcase PTC20000 RemoveCause(inout HUNT_PID rlopid, inout HUNT_PID chpid0, in
out HUNT_PID chpid1, inout HUNT_PID chpid2, inout HUNT_PID chpid3, inout HUNT PI
D chpid4, inout HUNT_PID chpid5, inout HUNT_PID chpid6, inout PROCESS ttcnpid) :
state on OSEComponent {

    nsp Out: OSE.system:OSE);
}
log("Test Case PTC20000_RemoveCause");
var verdicttype lpdchResult;
// declare and activate default
var default def_20000 := null;
def_20000 := activate(def_chcell(rlopid, ttcnpid));
// declare Test Case timer
timer TCTimer := 0.5;
-----chcell_20000.ttcn----- Fri Sep 12 12:55 3 39 L175 C0 (TTCN-3)--46s-----
| | /home/econorv/tmp/
| ttcn3_compiler -p /home/econorv/tmp/chcell_20000.ttcn
| Warning: The license key will expire within 12 days.
| Notify: Parsing TTCN-3 module '/home/econorv/tmp/chcell_20000.ttcn'....
| TTCN-3 parse finished at Fri Sep 12 12:53:49
-----*ttcn-3 parse*----- Fri Sep 12 12:55 3 39 L1 C0 (TTCN-3 parse:exit [0])--All-----
| N-x ttcn3-parse
```

Results

- Reliable execution of tests
- Build performance
 - Total re-build, ~ 25-30 min
 - 1 Signal/Message change, ~ 2 min
 - Test Case change, ~ 55 seconds
- Integration with product build process through 'clearmake'
- Code coverage with PurifyPlus™ & Test RealTime™
- Full round-trip build integration
 - Product build, test build, test execution, post-processing, run-time analysis

Result from clearmake™ build

The screenshot shows the IBM Rational Test RealTime - [Code Coverage] application. The left pane displays a file tree with various source files. The main pane shows a summary of coverage metrics for the 'Root' directory, followed by a '2 - Files list' table.

Item	Functions	Functions and exits	Calls	Statement blocks	Decisions	Loops
zcnomain	none	none	N/A	none	N/A	N/A
bit.c	0.00 % 0/1	0.00 % 0/2	N/A	0.00 % 0/1	N/A	N/A
start_REOS3	0.00 % 0/1	0.00 % 0/2	N/A	0.00 % 0/1	N/A	N/A
run_header1	0.00 % 0/1	0.00 % 0/2	N/A	0.00 % 0/1	N/A	N/A
run_header2	0.00 % 0/1	0.00 % 0/2	N/A	0.00 % 0/1	N/A	N/A
zcnmain	0.00 % 0/1	0.00 % 0/2	N/A	0.00 % 0/1	N/A	N/A
create_emb_and_env	0.00 % 0/1	0.00 % 0/2	N/A	0.00 % 0/1	N/A	N/A

What went right?

- Good support from tool suppliers
 - Ericsson, Hungary (Titan TTCN-3), IBM Rational
- Good support from management
- Pilot strategy
 - 1st pilot was not promising, 2 hour build times
 - 2nd pilot with upgraded Titan TTCN-3 went better
- Test environment extended with tcp/ip application test ports
- Tool sourcing in line with Ericsson corporate strategy
- Move from PurifyPlus™ to Test RealTime™

What went wrong?

- Long initial test environment setup
- Build/Execution times in initial pilot were too long
 - 2 hour build times
- Integration with PurifyPlus™ was unstable
 - code coverage logging not stable, memory profiling not useful
- Unclear project requirements on code coverage goals
- Unclear project requirements on managing complexity
- Memory analysis still not well supported

What could we have done better?

- Initial test setup pilot phase could have been a lot shorter
 - on-site deployment support from tool supplier
- Involve test simulator supplier a lot earlier in the pilot phase
 - Save time in run-time analysis tool integration
- Improvement focus on SQA process and project goals earlier

Additional benefits

- How to test tcp/ip application interface?
 - Add TTCN-3 tcp/ip test port to test configuration
 - Test Case easily adapted to new test interface
- Much more economic test capability for unit/load module tests compared to pure host based module/class/function tests
- Test RealTime™ provided complexity metrics
 - Input to system improvements & test planning

Improvements

- Methods
 - Software Quality Ranks
 - Better component test planning and follow-up
 - Fault-slip-through analysis & Root Cause Analysis
 - System performance profiling -> unit performance profiling
- Tools
 - Test RealTime™ memory profiling on real target or H/W simulator (e.g. Virtutech Simics)
 - Improved static checking, e.g. Coverity, Polyspace
 - IDE integration, Eclipse Test & Performance Tools Platform
 - Model based test generation (e.g. Conformiq)

Why Eclipse?



- Open source IDE supporting modelling, coding, test design, test execution, run-time analysis
- IBM Rational tooling is behind Eclipse & TPTP
- TestingTech TTCN-3 toolset based on Eclipse TPTP
- Major tool vendors behind Eclipse
 - Mentor, WindRiver, Enea, QNX, IBM, Borland, BAE, Computer Associates, Mercury,

Eclipse Test & Performance Toolkit Project

- [TPTP Platform Project](#)
- [Testing Tools Project](#)
- [Tracing and Profiling Tools Project](#)
- [Monitoring Tools Project](#)



© Ericsson AB 2006

Experiences of Design Based Testing with TTCN-3

ERICSSON

References

- ETSI TTCN-3
<http://www.etsi.org/ptcc/ptccttcn.htm>
- GNU Emacs
<http://www.gnu.org/software/emacs/emacs.html>
- GNU Emacs ttcn3 mode
<http://packages.debian.org/testing/editors/ttcn-el>
- GNU C/C++ compiler
<http://gcc.gnu.org>
- IBM Rational ClearCase
<http://www-306.ibm.com/software/awdtools/clearcase>
- IBM Rational PurifyPlus
<http://www-306.ibm.com/software/awdtools/purifyplus>
- IBM Rational Test RealTime
<http://www-306.ibm.com/software/awdtools/test/realtime/index.html>
- Eclipse Test & Performance Tools Platform Project
<http://www.eclipse.org/tptp>
- Ericsson Software Improvement through Software Quality Ranking
[Sigrid Eldh, RTiS 2003](#)

© Ericsson AB 2006

Experiences of Design Based Testing with TTCN-3

ERICSSON

