Introduction to TTCN-3

Colin Willcock

Nokia Research Center
Overview

- Introduction
  - TTCN Testing Background
  - Future Testing Challenges
- The TTCN-3 Language
  - Background
  - Example
- TTCN-3 in Practice
  - Current Status
  - Future Outlook
Overview

- Introduction
  - TTCN Testing Background
  - Future Testing Challenges

- The TTCN-3 Language
  - Background
  - Example

- TTCN-3 in Practice
  - Current Status
  - Future Outlook
Test Language

Why use a standardized test language?
To know what we are testing!

Only if you can unambiguously specify a test case, you can know what you are testing.
TTCN-2

- designed purely for testing
- internationally standardized language
- well-established and widely used (TTCN-2)
- uses conformance testing methodology (ISO 9646)
- tool independent
- supports ASN.1
Testing Challenges

- Increasing complexity of products
- Number of GSM Specifications 1306
- Number of 3G Specifications 2290
Future Testing Challenges

- Pressure to shorten time to market
  - New systems and services must be available quicker
  - How can we reduce testing time?

- Pressure to improve quality
  - SW outage average time for Network elements measured in seconds per year
  - How can we improve testing quality (and quantify it)

- New types of testing
  - IP based protocols
  - Text based protocols
  - Unified testing approach for software and protocol testing

- Not unique to Telecomm industry
- Maybe proposed solutions are also general
Overview

- Introduction
  - TTCN Testing Background
  - Future Testing Challenges
- The TTCN-3 Language
  - Background
  - Example
- TTCN-3 in Practice
  - Current Status
  - Future Outlook
What is TTCN-3

- **Test and Testing Control Notation**
- New Internationally standardized testing language for formally defining test scenarios. Designed purely for testing
- Taking the best bits of TTCN-2 and combining them with a new more powerful textual notation
- Tool Independent

```plaintext
testcase Hello_Stephan ()
{
    port.send("Hello Stephan");
    alt{
        []port.receive("Hello Colin")
        {setverdict( pass)}
        [else]
        {setverdict( inconc)} //Stephan asleep!
    }
}
```
Why is TTCN-3 Important

- More Productive
  - Easier to learn
  - Easier to use

- More Powerful
  - Extended functionality
  - Support for IP protocol
  - Support text-based protocols

- More Flexible
  - Not tied to OSI model
  - For many types of testing

- More Extendable
  - Extensibility built-in

---

Software Testing

- Function
- Module

Protocol Testing

- Layer
- Unit
- Integration

TTCN-2

TTCN-3
TTCN-3 Background

- STF 133 formed by ETSI MTS in 1997 to produce TTCN version 3
- Co-operation with ITU-T SG10
- Voluntary contributions from Nortel, Ericsson, Telelogic, Nokia etc.
- More than 200 members in STF133 discussion group.
- Language standardized in October 2000
TTCN-3: Standardisation
A Comparison of TTCN Language Versions

**TTCN–2**
- machine processable source file format (.mp files)
- tabular presentation format
- supports timers and message-based communication
- static test configurations
- static adaptation layer interface (PCOs)

**TTCN–3**
- textual source file format (like programming language)
- tabular, graphical (MSC), or any user-defined presentation format
- supports timers, message-based and procedure-based communication, and external functions
- dynamic test configurations and one-to-many connections
- separate dynamic adaptation layer interface (component)
TTCN-3 Advantages

- Extend usability of TTCN-2
  - powerful features, e.g. different communication mechanisms
  - intended for several application areas
  - presentation formats independent of core language
  - standardized interfaces

- Do not reinvent
  - retain proven concepts of TTCN-2
  - retain TTCN-2 expertise of developers
Application area Conformance Testing
Application Areas

Protocol (Conformance) Testing

Software Testing

Text based testing

IP based testing

GET

http://... 404...
Separation of Content and Form

- TTCN-3 is based on textual core language

```
P.send(...);
P.receive(...);
...```

```
P!...
...
P?...
...```
Separation of Content and Form (cont.)

- Further presentation formats supported

query(...) > P

P > answer(...)

P.receive("123")

123

P.send("4")

graphical presentation format (standardized)

proprietary presentation format
Standardized TTCN-3 Runtime Interface (TRI)

- standardized interface between test suite and protocol layers
- allows independent development and portability
Standardized TTCN-3 Runtime Interface (TRI)

Test System

TTCN

System Under Test

Layer 4

Layer 3

Layer 2

Layer 1

• testing tools supporting TRI can be integrated easily
TTCN-3 Step by Step: DNS Server

- TTCN-3 core notation is introduced by developing an example test case for a Domain Name Service (DNS) server.
DNS Server – Test Purpose

- The Internet’s Domain Name System (DNS) service offers hostname to IP Address resolution
- Communication is message based (UDP)
- We want to test the correct resolution

www.nokia.com => 193.65.100.110
TTCN-3 Modules

- Main building block of TTCN-3 is a module
  - Unit of Compilation
  - Contains definitions
  - And an optional control part

```plaintext
module DNS {

// module definitions

// module control (optional)

}
```
Module Definitions

- Module definitions
- **Type definitions**
- Templates
- Port definitions
- Component defs
- Testcase
- Control part

```
type record DNSQuery {
    charstring hostname;
    AnswerType answer optional;
    QueryType qtype;
}
type union AnswerType {
    Byte address[4],
    charstring hostname;
}
type integer Byte (0 .. 255);
type enumeration QueryType {
    A, NS, CNAME, MX
```
Module Definitions contd.

- Module definitions
- Type definitions
- **Templates**
- Port definitions
- Component defs
- Testcase
- Control part

```template DNSQuery query {
  hostname := "www.nokia.com";
  answer   := omit;
  qtype    := A;
}

template DNSQuery answer modifies query {
  answer.address := {193,65,100,110};
}
```

```
<table>
<thead>
<tr>
<th>&quot;www.nokia.com&quot;</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>193, 65, 100, 110</td>
<td>A</td>
</tr>
</tbody>
</table>
```
Module Definitions contd.

- Module definitions
- Type definitions
- Templates
- Port definitions
- Component definitions
- Testcase
- Control

Port definitions

```plaintext
type port DNSPort message {
  inout DNSQuery;
  // a port may send/receive messages
  // of more than one type
}
```

Component definitions

```plaintext
type component DNSTester {
  port DNSPort P;
  // a component may have more than one port
}
```
Module Definitions contd.

- Module definitions
- Type definitions
- Templates
- Port definitions
- Component defs
- Testcase
- Control part

```java
testcase Testcase1() runs on DNSTester {
P. send(query);
P. receive(answer);
setverdict(pass);
stop;
}
// there may be more than one in a module
```
Module Definitions contd.

- Module definitions
  - Type definitions
  - Templates
  - Port definitions
  - Component defs
  - Testcase
  - **Control part**

- Controls the execution of test cases

```cpp
control {
    execute(Testcase1());
    // more testcases might follow
    // C-like control structures available
}
```
execution of a test case

```
testcase Testcase1() runs on DNSTester {
    P.send(query);
    P.receive(answer);
    setverdict(pass);
    stop;
}
```
Dealing with erroneous behavior

- \texttt{P.receive(answer)} \textbf{blocks} until it receives a message that matches \texttt{answer}.
- Any other message does not unblock the tester, which then blocks forever.
- If no message is received, the tester will also block forever.
Dealing with erroneous behavior contd.

testcase Testcase2() runs on DNSTester {
    timer t := 5.0;
    P.send(query);
    t.start;
    alt {
        [] P.receive(answer) {
            setverdict(pass);
        }
        [] P.receive { // any message
            setverdict(fail);
        }
        [] t.timeout {
            setverdict(inconc);
        }
    }
    stop;
}
Non-local DNS Query

Tester

Send fully qualified hostname

Return IP-address

Ask for remote DNS

Get remote address

Ask for IP address

Return desired IP-address

Local Domain Name Server

System Under Test

Parallel Test Component 1

Local Network Client

Main Test Component

Parallel Test Component 2

The internet‘s root name service

Parallel Test Component 3

Remote DNS

Non-local DNS Query

Send fully qualified hostname

Return IP-address

Ask for remote DNS

Get remote address

Ask for IP address

Return desired IP-address

Local Domain Name Server

System Under Test
Non-local DNS Query contd.

Tester

Client

(“www.nokia.com”, A)

Tester

SUT

DNS

root NS

NS

(“www.nokia.com”, A)

(“nokia.com”, NS)

(“nokia.com”, “ns.nokia.com”, NS)

(“www.nokia.com”, 193.65.100.110, A)

(“www.nokia.com”, 193.65.100.110, A)
Parallel Test Components

- Test system interface

```plaintext
type component TSI {
    port DNSPort CLIENT,
    port DNSPort ROOT,
    port DNSPort NS
}
```
Parallel Test Components contd.

- TTCN-3 functions define the behavior of the parallel test components.

```
function RootBehaviour() runs on DNSTester {
    alt { [] P.receive(rootquery) {
        P.send(rootanswer);
        setverdict(pass);
    }
    [] P.receive { setverdict(fail);}
} stop;
}
```
Dynamic Configuration

```plaintext
testcase Testcase3() runs on MTC
    system TSI {
        var DNSTester RootComp, NSComp, ClientComp;
        RootComp := DNSTester.create;
        NSComp := DNSTester.create;
        ClientComp := DNSTester.create;
        map (RootComp:P, system :ROOT);
        map (NSComp:P, system :NS);
        map (ClientComp:P, system :CLIENT);
        RootComp.start (RootBehaviour());
        NSComp.start (NSBehaviour());
        ClientComp.start (ClientBehaviour());
        ClientComp.done;
        // block until ClientComp is done
        stop;
    }
```

- Re-configuration during run time is possible
Overview

- Introduction
  - TTCN Testing Background
  - Future Testing Challenges
- The TTCN-3 Language
  - Background
  - Example
- TTCN-3 in Practice
  - Current Status
  - Future Outlook
Current Status

- A number of commercial TTCN-3 toolsets are available
- ETSI is already using TTCN-3 for the TIPHON SIP and IPv6
- TTCN-3 is already being used in SIP and IPV6 bake-off events
- Major telecoms companies are developing migration plans for moving from TTCN-2 to TTCN-3.
- European Projects to disseminate technology
TT-Medal Project

Project Goals:

- To develop the methodologies, tools and industrial experience to enable European industry to test more effectively and more efficiently.
- To drive the deployment of TTCN-3 testing technology into European industry.
TT-Medal Partners

CONFORMIQ
VIT
NOKIA

TT-Medal Partners

logicaCMG
CWI
IMPROVE

TT-Medal Partners

DAIMLERCHRYSLER
Fraunhofer Institute for Open Communication Systems

<testing_tech>
TT-Medal Results

- TTCN-3 Asset Box will provide all the tools for:
  - generic testing processes,
  - systematic test development and testing methodology,
  - reuse,
  - architecture,
  - new application domains,
  - tool prototypes,
  - UML 2.0 testing profile,
  - test validation, and
  - lightweight testing process & tools
TT-Medal Results

- **Training package**: to show *how* the new technologies should be used, help project partners and others in European industry with effective testing in their business processes.

- **Experience package**: to show *where* these new technologies should be applied and *why*, will provide the means for sharing data on adequate tools, testing methods, applicability of processes, available test cases, and conformance specification, how to obtain sufficient testing resources (from management) and how to leverage these.
TT-Medal Structure

WP 4
Industrial Cases

WP 3
Tester Tools

WP 2
Test Development Tools

WP 1
Methodology

initial requirements

methods and process
evaluations

theory
evaluations
theory
TT-Medal Application Areas

Industrial Demands on Testing:
- Mobile Communications
- Railways
- Automotives

Efficient Testing Platform
Methodologies, Tools and Standards

Industrial Case Studies
TTCN-3 Outlook

- **Short term:**
  - Replacing TTCN-2 in functional and conformance testing as standard language.
  - Increasing use within the IP world especially for text based protocols
  - Possible key technology in the IP/telecom convergence.

- **Medium term:**
  - Expanding from pure protocol testing to software testing and interworking testing.
  - Possible key technology for unifying testing technology across whole product development.

- **Long term:**
  - Real time and performance testing? (European INTERVAL project)
  - Integration with UML (UML testing profile)
The Truth About TTCN-3....

Doggert's School for the Socially Oblivious

Today I'll teach you to recognize when you're boring.

This is called a yawn. When you see one, stop talking about yourself.

Breakout Session

And then I used TTCN-3

Look, look!