The TTCN-3 Language

An Introduction to TTCN-3

Motivation
Theofanis Vassiliou-Gioles
Founder and CEO
vassiliou@testingtech.de

<testing_tech>

Testing Technologies
Rosenthaler Str. 13
10119 Berlin, Germany
Agenda

- Motivation
- Basic concepts
- TTCN-3 by example
- Applications in fixed and wireless communications
- Conclusion
TTCN-3 – What is This?

- A testing technology
  - Telecommunication, Software Industry, Automotive
- A textual and graphical test scripting language
  - Familiar look-and-feel
- A test implementation language
  - You know today how your test system looks like
- A test and test implementation framework
  - Don’t reinvent the wheel, use existing resources
- A philosophy
  - If you know it you like it
TTCN-3 – Who is Using It?

- It is supported by multiple
  - Tool vendors
  - Test devices
  - Standardization bodies

- Used by thousands of test engineers and test specialists worldwide
  - On corporate level for test automation and documentation
  - In education and research
  - In different domains like telecom, software and automotive
Testing Today

- Is
  - Important
  - Expensive
  - Time critical

- But
  - Only rarely practiced
  - Unsystematic
  - Performed by hand
  - Error-prone
  - Uncool ("If you are a bad programmer you might be a tester.")
  - Destructive
Why Using TTCN-3 (1)
Why Using TTCN-3 (2)
Testing Is

- a technical process
- performed by experimenting with a software product
- in a controlled environment
- following a specified procedure
- with the intent of observing one or more characteristics of the product
- by demonstrating the deviation of the product’s actual status from the required status/specification
Testing Today’s Systems

- Component-based
  - Test-components to respect the SUT structure

- Distributed
  - Not only local, but also distributed test setups

- Dynamic in terms of behavior and configuration
  - Testing of static and dynamic aspects; dynamic creation of test components

- Use various type systems to exchange data
  - Open to all type systems

- Service is essential
  - Concentration on service-oriented black-box testing
Design Principles of TTCN-3

- One test technology for different tests
  - Distributed, platform-independent testing
  - Integrated graphical test development, documentation and analysis
  - Adaptable, open test environment

- One test technology for distributed IT and Telco systems
History (1)

- **TTCN (1992)**
  - Published as an ISO standard
  - Tree and Tabular Combined Notation
  - Used for protocol testing (GSM, N-ISDN, B-ISDN)
  - Rather for conformance testing

- **TTCN-2/2++ (1997)**
  - Concurrent tests
  - Modularization
  - Manipulate external data
  - Still for conformance testing
History (2)

  - Testing and Test Control Notation
  - ETSI STFs
  - Proper language (well defined syntax and semantics)
  - Enhanced communication, configuration and control
  - Standard test specification (SIP, SCTP, HiperLan, HiperAccess, IPv6 etc.)

  -> Changing requests, extension proposals
The TTCN-3 Language

Introduction to TTCN-3

Basic Concepts
What is TTCN-3?

- Testing and Test Control Notation
- 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

```plaintext
testcase Hello_Bob () {
    p.send("How do you do?");
    alt {
        []p.receive("Fine!");
        {setverdict( pass )};
        [else]
        {setverdict( inconc )}  //Bob asleep!
    }
}
```
testcase Hello Bob () {
    p.send("How do you do?");
    alt {
        p.receive("Fine!");
        setverdict( pass );
    }
    else {
        setverdict( inconc ) //Bob asleep!
    }
}
Application Areas

- Multiple application areas
  - Software testing
  - Text-based protocols …

- Additional communication paradigm
  - Message-based communication
  - Procedure-based communication

- Different kinds of testing
  - Functional testing
  - Conformance testing
  - Scalability testing …

- Covering larger range within development cycle
  - From unit to integration testing
Main Aspects of TTCN-3

- **Triple C**
  - Configuration: Dynamic concurrent test configurations with test components
  - Communication: Various communication mechanisms (synchronous and asynchronous)
  - Control: Test case execution and selection mechanisms

- **Features**
  - Well-defined syntax, static and operational semantics
  - Different presentation formats
  - Module concept
  - Extendibility via attributes, external function, external data
  - Harmonized with ASN.1
Differences TTCN-2 / TTCN-3

- **Configuration:**
  - Static configuration with configuration tables
  - Dynamic configuration with arbitrary amount of components
  - Differentiation between PCOs and CPs
  - One port concept

- **Communication:**
  - Asynchronous communication only
    - Abstract Service Primitives
    - Protocol Data Unit
  - Procedure- and message-based communication
    - Procedures
    - Messages

- **Control:**
  - Static selection of test cases via selection expression
  - Complete high level control flow mechanisms
Differences TTCN-2 / TTCN-3

- **Externalisation**
  - Test suite operations
  - External function
  - PICS / PIXIT
  - Module parameters

- **Data types, values**
  - TTCN-2 / ASN.1
  - TTCN-3, ASN.1, IDL, XML, ...

- **Modularisation**
  - Possible but seldom used
  - Central concept

- **Extensibility**
  - Not possible
  - Attributes, languages

- **Methodology**
  - Conformance Testing Methodology Framework (CTMF) (ISO 9646)
    - No specific

- **Presentation**
  - Tabular, machine processable
  - Textual, graphical, tabular, ...

- **Implementation**
  - No runtime interfaces
  - TTCN-3 Runtime Interfaces, TTCN-3 Control Interfaces

- **Acronym**
  - Tree and Tabular Combined Notation
  - Testing and Test Control Notation
New: TTCN-3 Edition 3 Standards

- ETSI ES 201 873-1  TTCN-3 Core Language (CL)
- ETSI ES 201 873-2  TTCN-3 Tabular Presentation Format (TFT)
- ETSI ES 201 873-3  TTCN-3 Graphical Presentation Format (GFT)
- ETSI ES 201 873-4  TTCN-3 TTCN-3 Semantics
- ETSI ES 201 873-5  TTCN-3 TTCN-3 Runtime Interface (TRI)
- ETSI ES 201 873-6  TTCN-3 TTCN-3 Control Interfaces (TCI)
- ETSI ES 201 873-7  Integration of ASN.1
- ETSI ES 201 873-8  Integration of IDL
- ETSI ES 201 873-9  Integration of XML

New edition 3 released and published
Standard available for download at http://www.etsi.org/ptcc
Testing Tech tools support Edition 3
TTCN-3 By Example

Tester

Local Network Client

Main Test Component

Send fully qualified hostname

Return IP-address

Local Domain Name Server

System Under Test
TTCN-3 By Example

Tester
Client

pass
Main Test Component

DNS

System Under Test

(www.testingtech.de,A)
(www.testingtech.de,217.160.141.54,A)
TTCN-3 Modules

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

```
module DNS {

  // module definitions

  // module control (optional)

}
```
Module Definitions

Contains descriptions for

- What type of data the System Under Test understands
- How the System Under Tests can be accessed and what environment a test component needs
- When to communicate what with the SUT and why
- Dependencies between test cases, if any
Module Definitions (1)

- Module definitions
  - Type definitions
  - Port definitions
  - Component definitions
  - Test case
  - Templates
  - Control part

```
type record DNSQuery {
    charstring hostname,
    AnswerType answer optional,
    QueryType qtype
}
type union AnswerType {
    Byte ipAddress[4],
    charstring hostname
}
type integer Byte (0 .. 255);
type enumeration QueryType {
    A, NS, CNAME, MX
}
```
Module Definitions (2)

- Module definitions
  - Type definitions
  - Port definitions
  - Component definitions
  - Test case
  - Templates
  - Control part

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 (3)

- Module definitions
  - Type definitions
  - Port definitions
  - Component definitions
- Test case
- Templates
- Control part

```plaintext
testcase Testcase1() runs on DNSTester {
    P.send(query);
    P.receive(answer);
    setverdict(pass);
}

// there may be more than one in a module
```
Module Definitions (4)

- Module definitions
  - Type definitions
  - Port definitions
  - Component definitions
  - Test case
  - Templates
  - Control part

```plaintext
// Example of type definitions

type record DNSQuery {
    charstring hostname,
    AnswerType answer optional,
    QueryType qtype
}

type union AnswerType {
    Byte ipAddress[4],
    charstring hostname
}
```

Example: Querying www.testingtech.de

```
query "www.testingtech.de" A

answer "www.testingtech.de" A
```

Result: IP addresses

```
217, 160, 141, 54
```
Module Definitions (5)

- Module definitions
  - Type definitions
  - Port definitions
  - Component definitions
- Test case
- Templates
- Control part
  - Controls the execution of test cases

```plaintext
control {
    execute(Testcase1(), 5.0);
    while ( /* condition */ ) { }

    // more testcases might follow
    // C-like control structures available
}
```
Execution of a Test Case

```c
control {
  execute(Testcase1(), 5.0);
  while( /* condition */ ) {
  }
}
```

testcase Testcase1() runs on DNSTester {
  P.send(query);
  P.receive(answer);
  setverdict(pass);
}

Is this test case definition adequate?
Dealing with Erroneous Behavior (1)

- `P.receive(answer)` blocks until it receives a message that matches `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 (2)

document: Testing Technologies, Germany & Tekstrom Infotech Private Limited, India

```tcc

testcase Testcase2() runs on DNSTester {
    timer t := 4.0
    P.send(query);
    t.start;

    alt {
        [] P.receive(answer) {
            setverdict(pass);
        }
        [] P.receive { // any message
            setverdict(fail);
        }
        [] t.timeout {
            setverdict(inconc);
        }
    } stop;
}
```

Diagram:
- Client
- DNS
- (www.testingtech.de,A)
- (217.160.141.54,A)
- pass
- fail
- inconc
Dealing with Erroneous Behavior (2)

testcase Testcase2() runs on DNSTester {
    /* timer t := 4.0 */
    P.send(query);
    t.start;
    alt {
        [] P.receive(answer) {
            setverdict(pass);
        }
        [] P.receive { // any message
            setverdict(fail);
        }
        [] t.timeout {
            setverdict(inconc);
        }
    }
    stop;
}

type component DNSTester {
    port DNSPort P;
    timer t := 4.0;
}
Code Reusability – Altsteps and Defaults

alt {
    [] P.receive(answer) {
        setverdict(pass);
    }
    [] P.receive { // any message
        setverdict(fail);
    }
    [] t.timeout {
        setverdict(inconc);
    }
}

refactor

altstep RefactoredAltstep() runs on DNSTester {
    [] P.receive { // any message
        setverdict(fail);
    }
    [] t.timeout {
        setverdict(inconc);
    }
}

becomes

var default d := activate(RefactoredAltstep());
P.send(query);
t.start;
P.receive(answer);
setverdict(pass);
Non-Local DNS Query (1)

Local Network

Client

Parallel Test Component 1

Main Test Component

Tester

Send fully qualified hostname

Return IP address

Ask for remote DNS

Get remote address

The internet’s root name service

Parallel Test Component 2

Remote DNS

Parallel Test Component 3

Local Domain Name Server

System Under Test

Parallel Test Component 1

Parallel Test Component 2

Parallel Test Component 3
Non-Local DNS Query (2)

Tester

Client

("www.testingtech.de", A)

SUT

DNS

("testingtech.de", NS)

("testingtech.de", "ns.testingtech.de", NS)

("www.testingtech.de", A)

("www.testingtech.de", 217.160.141.54, A)

Tester

root NS

NS

("www.testingtech.de", 217.160.141.54, A)
From Simple To Complex Test Scenarios

- Test systems need more interfaces
  - Test System Interface has to be extended

- Additional test behavior needed at additional test interfaces
  - Behavior of Local Network Client already covered in Testcase 2
  - Behavior of RootNS and NS required

- Test case that combines all pieces
Parallel Test Components (1)

- Test system interface

```typescript
type component DNSTester {
  port DNSPort P

  // a component may have more than one port
}
```

![Diagram](image)
From Testcase to Test Function

Functions define the behavior of the parallel test components

testcase Testcase2() runs on DNSTester {
    var default d := activate(RefactoredAltstep());
    timer t := 5.0;
    P.send(query); t.start;
    P.receive(answer);
    setverdict(pass);
    stop;
}

becomes

function ClientBehaviour() runs on DNSTester {
    var default d := activate(RefactoredAltstep());
    timer t := 5.0;
    P.send(query); t.start;
    P.receive(answer);
    setverdict(pass);
    stop;
}
Additional Test Behavior

Simple „react-on-request“ behavior

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

```plaintext
function NSBehaviour() runs on DNSTester {
  alt { [] P.receive(nsquery) {
    P.send(nsanswer);
    setverdict(pass);
  }
  [] P.receive {
    setverdict(fail);
  }
}
```
Dynamic Configuration

testcase Testcase3() runs on MTC
system TestSystemInterface {

var DNSTester ClientComp, RootComp, NSComp;

ClientComp := DNSTester.create;
RootComp := DNSTester.create;
NSComp := DNSTester.create;

map(ClientComp:P, system:CLIENT);
map(RootComp:P, system:ROOT);
map(NSComp:P, system:NS);

ClientComp.start(ClientBehaviour());
RootComp.start (RootBehaviour());
NSComp.start (NSBehaviour());

ClientComp.done;
// block until ClientComp is done
stop;
}

Re-configuration during run time is possible
Procedure-based Communication (1)

DNS also allows queries over TCP/IP connections
How can this be adequately tested?

Signature definitions

```plaintext
signature DNSCall(
    inout charstring hostname,
    out AnswerType ans,
    inout QueryType qtype
);
```

Ports for procedure-based communication

```plaintext
type port DNSCallPort procedure {
    out DNSCall
}
```

Component

```plaintext
type component DNSCallComponent {
    port DNSCallPort Q
}
```
Procedure-based Communication (2)

```c
testcase Testcase4() runs on DNSCallComponent {4
    Q.call(DNSCall:{"www.testingtech.de", -, A}, 5.0);
    {
        [] Q.getreply(DNSCall:{"www.testingtech.de",{217.160.141.54},A}) { setverdict(pass)
        }
        [] Q.getreply(DNSCall: ?) { setverdict(fail)
        }
        [] Q.catch(timeout) { setverdict(inconc)
        }
    }
    stop;
}
```
A Little Bit on Syntax

- Case Sensitive
  - 131 (edition 3) keywords, all lower case
  - Identifiers

- Comments
  - Multi line comments: /* */
  - Single line comments: //

- Statements are terminated with: ;

- Statement blocks are enclosed in: { }

- Operators
  - Assignment: :=
  - Comparison: !=, ==, <=, >=
Summary

- TTCN-3 as standardized language for testing
- Easy description of test scenarios
  - Different presentation formats
- Clear specification of test configurations
  - Parallel test components / concurrent behavior
- Wide range of applicability
  - Different communication paradigms
The TTCN-3 Language

An Introduction to TTCN-3

Applications in Fixed and Wireless Communication
Hot Topics from the Testing Perspective

- Wireless communication
  - 3G technology
  - Mobile WiMAX

- Fixed line communication – IMS
  - Interoperability
  - Scalability / Performance
  - Network Infrastructure
Standardization Bodies

- **3GPP**
  - RAN #5 focuses on conformance testing of user equipment
    - Testing the user equipment over the air

- **WiMAX Forum**
  - Focuses on certification program for fixed and mobile WiMAX
    - WiMAX Certification of subscriber and base stations

- **ETSI**
  - IMS Benchmarking test specification
  - IPv6 conformance testing
  - Interworking test specification between SIP and ISUP

- They use TTCN-3 as standardized test specification language
Examples of TTCN-3 Based Products

- Test standards implementations
  - WiMAX conformance tester
  - IMS benchmarking test suite
  - …

- Test devices with TTCN-3
  - IMS Interoperability test suite
  - …
WiMAX Testing Requirements

- Protocol Stack Initial Dev
- Hardware Dev
- DSP Development

Conformance Testing
Pre-Validation & Regression Testing
Test House
A WiMAX Test System

- WiMAX Forum approved protocol conformance test equipment
  - IEEE 802.16e mobile and portable WiMAX standard
- Base station and subscriber station testing
- Supporting interfaces
  - Analogue I/Q base band
  - RF (over cable and over air)
- Development and conformance test cases including scripting assistance
Protocol Conformance Test System

- TTCN-3 for scripting and conformance
  - Integrated TTCN-3 Development and Execution environment
  - User friendly, intuitive programming interface

- Result Analysis
  - Manage + organise logs
  - Report generator and HTML log exporter
  - Layer 1 and 2 logging and filtering
  - Advanced debug and message decode

- Ready to use product by Aeroflex
  - TTworkbench as TTCN-3 environment
  - MiNT Log Manager and MiNT Log Analyzer
IMS Benchmark Testing

User Equipment Emulations

Benchmark Test System

Control, coordination and measurement

I-CSCF
P-CSCF
HSS
S-CSCF

...
Why IMS Benchmarking? (1)

Goal - performance benchmark for IMS components
  ▶ Performance and scalability testing of all IMS and related components with simulated real-world traffic
  ▶ Measurement and analysis of important QoS parameters

Why
  ▶ Creation of objective means to compare overall IMS of different systems by performance (and price)
  ▶ Check ability of hardware/software to run the IMS; Capacity planning
Why IMS Benchmarking? (2)

How
- Define standard scenarios and traffic models for the work load
- Define the metrics to be measured
- Standardize the test procedure, test parameters and the Benchmark test report

Where
- Standardization of IMS benchmarking at ETSI TISPAN WG6
- Version 1.0 of IMS benchmarking standard to be adopted in February 2007
An IMS Benchmarking Test System (1)

Scenario description in TTCN-3
- Standardized implementation-independent test notation
- Allows full control and easy extension of scenarios
- Implemented scenarios
  - are flexible due to comprehensive test parameterization
  - provide full state machine for each user

Distributed test execution platform
- Deploys, creates and coordinates distributed parallel test components emulating user equipment on several hosts
- Synchronized traffic load, measurement and logging of results
- Good scalability
An IMS Benchmarking Test System (2)

- Benchmark result analysis, reporting and visualization
  - Collects the log files from distributed test servers and generates the test report
  - Visualization of several metrics and DOC (depending on scenarios)

- Ready to use IMS Benchmarking test suite by Fraunhofer FOKUS
  - TTworkbench Enterprise as TTCN-3 environment
  - TraVis tool with database support
  - Standard server hardware as hardware platform
The IMS World

**IMS Network**

- **P-CSCF**
- **I-CSCF**
- **S-CSCF**
- **MGW (IMS)**
- **MGCF**

**Applications**

- **PoC**
- **Enabling Services**
- **Presence**

**Foreign Domains**

- **PDF**

**Legacy Networks**

- **MGW (IMS)**

**Client**

- **UNI**
- **Gm**

- **PDF**
- **Gm**

- **NNI**
- **Gg**

- **Mw**
- **Cx**

- **Mw**
- **I-CSCF**

- **AS**
- **HSS**

- **Mw**
- **Cx**

- **Mg**
- **Mn/Ms**

Sponsored by: Testing Technologies, Germany & Tekstrom Infotech Private Limited, India
Testing An IMS Network

[Diagram showing the components of an IMS network with arrows indicating connections between P-CSCF, I-CSCF, S-CSCF, HSS, AS, and PoC Presence.]
IMS Interoperability Test Suite

Over 100 test cases focus on
- Registration
- Peer to peer communication
- Push to Talk over Cellular (PoC)
- RTP evaluation
- UNI
- NNI

Ready to use product by Tektronix
- TTworkbench as TTCN-3 environment
- Tektronix K1297-G35
Outlook

- **Standard Bodies**
  - Expansion of test suite specifications in TTCN-3
  - New bodies are likely to join

- **Test device vendors**
  - TTCN-3 based conformance testing products
  - More test device vendors to support natively TTCN-3

- **Companies**
  - Expansion of TTCN-3

- Join now the TTCN-3 movement