1

Introduction

The Testing and Test Control Notation Version 3 (TTCN-3) is an internationally standardised language for defining test specifications for a wide range of computer and telecommunication systems. It allows the concise description of test behaviour by unambiguously defining the meaning of a test case pass or fail. The predecessor of TTCN-3, TTCN-2, has been used successfully for over a decade, mostly in testing telecommunications systems. In this third revision of TTCN, the best parts of the previous testing language have been combined and extended with a powerful new textual syntax to create a universal testing language whose application area is no longer restricted to testing telecommunication systems.

Five years after the publication of the first edition of this book a lot of things have happened in the TTCN-3 community. TTCN-3 is celebrating its 10th anniversary and has grown into a 10 part standard with four extension packages to date. It has grown into a *global* testing language used well beyond telecommunication and standardisation. We are witnessing a rapid uptake of the language in Asia – especially India and China – which today provides already more than 40% of all testing services worldwide. TTCN-3 has established itself in the automotive and medical domain, with the banking sector promising to be the next big application area. The continued success and propagation of the language is visible in the annual international TTCN-3 User Conference which reflects these developments. In addition, TTCN-3 has further strengthened its position in standardisation and is used increasingly for certification and acceptance testing: In telecommunication, the third Generation Partnership Program (3GPP) [2] has decided to perform all of their future test suite development including their prestigious test suite for Long Term Evolution (LTE™[1]) [1] /4G terminals using TTCN-3 (see Chapter 16). The Open Mobile Alliance (OMA) [3] is using TTCN-3 for their test suite development in service provision and broadcasting across mobile networks. The WiMAX Forum [4] is using TTCN-3 for certifying the conformance of terminals and the interoperability of terminals and network elements. In the automotive domain, the influential AUTOSAR consortium [5] – composed of all the major car manufacturers, suppliers and service

[1] ™LTE is a trade mark of ETSI.

An Introduction to TTCN-3, Second Edition.
Colin Willcock, Thomas Deiß, Stephan Tobies, Stefan Keil, Federico Engler and Stephan Schulz.
© 2011 John Wiley & Sons, Ltd. Published 2011 by John Wiley & Sons, Ltd.

providers – has adopted TTCN-3 for the specification of its compliance test suites. And in the internet domain, TTCN-3 test suites are used in the context of certification of the IPv6 ready program [6]. Last but not least the European Telecommunication Standardisation Institute (ETSI) is continuing to use TTCN-3 in a wide range of application areas including Next Generation Networks (NGNs), electronic passport, radio technology as well as evaluating test execution traces at their interoperability events [7].

This book provides a solid introduction to the TTCN-3 language and its use. All the important concepts and constructs of the language are explained in a tutorial style, with the emphasis on extensive examples. This book also introduces the larger picture of how the testing language is related to the overall task of test system implementation. By doing so, it becomes the perfect companion to the available TTCN-3 language standards [8–21], filling the gaps like style guide, structuring and application. In addition, this book points out some of the dangers and pitfalls of TTCN-3 on the basis of our personal TTCN-3 experience from language standardisation, tool implementation and applying TTCN-3 for a number of years *in the real world*. The style and level of this book make it suitable for both engineers, learning and applying the language in the real world, and students, learning TTCN-3 as part of their studies. Although this book is intended to be accessible to a wide audience, it does assume that the reader has some basic knowledge of software programming.

This second edition of the book has been updated and revised to cover the additions, changes and extensions to the TTCN-3 language since the first version was published. In addition to this extensive new material caused by language evolution the book also contains a new section on testing frameworks and a major new example domain: LTE testing using TTCN-3. This domain is not just covered in the text but also in the form of downloadable tools and a test suite. This book is structured to present concepts in an order that offers the quickest start to the efficient use of the TTCN-3 language. In Sections 1.1 and 1.2, we discuss the advantages of using TTCN-3. Chapter 2 then goes on to introduce a complete example to get a first hands-on impression of the language and lists all the additional parts that are necessary to transform the TTCN-3 code into a working test system. In Chapter 3, we then move on to present the basic language concepts of TTCN-3, including basic types, operators and expressions, as well as the language constructs for control flow. In Chapter 4, the subject of test specification in TTCN-3 is considered in more detail by discussing the language constructs that are most commonly used for non-concurrent testing. Test cases, test verdicts and message-based communication are a few of the topics that are considered. Concurrent TTCN-3 is described in Chapter 5; the key issues considered are the usage and synchronisation aspects of test components. The importance of procedure-based communication is highlighted by providing a separate chapter, Chapter 6, which provides an in-depth discussion of this communication paradigm. Chapter 7 considers the issue of modularity, which is needed to address the issues of code re-usability as well as multi-user development. Chapter 8 provides a thorough introduction to the TTCN-3 type system, leaving more complex type topics such as external type systems to Chapter 9. Templates and advanced aspects of their use are brought up in Chapters 10 and 11. Chapter 12 considers TTCN-3 extension packages in general and the extension package for real-time testing specifically. With all the language parts described in detail, Chapter 13 then provides a detailed description of how TTCN-3 test systems work in practice. In Chapter 14 we consider frameworks for testing and Chapter 15 can be seen as a utility chapter that provides a collection of code examples and common sense advice. Chapter 16 rounds off the book by introducing LTE testing using TTCN-3. This provides the link to the tools and test suite available on the companion website which will enable you not just to read about TTCN-3, but actually experiment with it.

1.1 TTCN-3 as a Language

TTCN-3 is a language designed specifically for testing. Many constructs are similar to those in other programming languages but are extended with additional concepts not available elsewhere. These concepts include built-in data matching, distributed test system architecture and concurrent execution of test components. TTCN-3 has a larger type system than normal programming languages and includes native types for lists, test verdicts and test system components. In addition, TTCN-3 provides direct support for timers as well as for message-based and procedure-based communication mechanisms.

TTCN-3 is an internationally standardised test language [8–21]. Within these documents, the meaning of each and every language element is clearly and precisely specified. This means that a test script written in TTCN-3 is unambiguous. This precise definition of the language also leads to tool vendor independence, since every tool should execute a given test case in exactly the same way. Tool vendor independence facilitates easy moving from one TTCN-3 toolset to another and greatly helps in testing projects where test tools from different vendors are used in parallel. The language is designed to provide a single general-purpose testing language suitable for a wide range of testing applications. It can be used across the whole product development cycle. In this way, TTCN-3 can provide major benefits in terms of return on investment in testing tools, training and, naturally, product quality.

At its heart, TTCN-3 has a powerful, intuitive textual format for defining test scenarios, that is similar to conventional procedural programming languages. This textual format is referred to as the *TTCN-3 core notation* [8]. This book concentrates on this core notation, with the following chapters describing in detail its syntax and use.

In addition to the core notation, TTCN-3 also supports the specification of test scenarios using other presentation formats. A TTCN-3 *presentation format* provides an alternative way of specifying test scenarios visually or in a context-specific manner. All presentation formats can be converted into the core notation while preserving their meaning as shown in Figure 1.1. Two presentation formats have been standardised initially. The standardised tabular presentation format [9] was designed to give the test developers the 'look and feel' of the existing TTCN-2 tabular format. This format was introduced to provide an easy migration path for existing TTCN-2 users into the TTCN-3 world. An example is shown in Figure 1.2. The graphical presentation format [10] uses an extended version of an MSC-like Message Sequence Charts [22] notation for specifying test scenario behaviour as shown in Figure 1.3. Neither of the two presentation formats has gained acceptance by the TTCN-3 community. Therefore they are not considered further in this book.

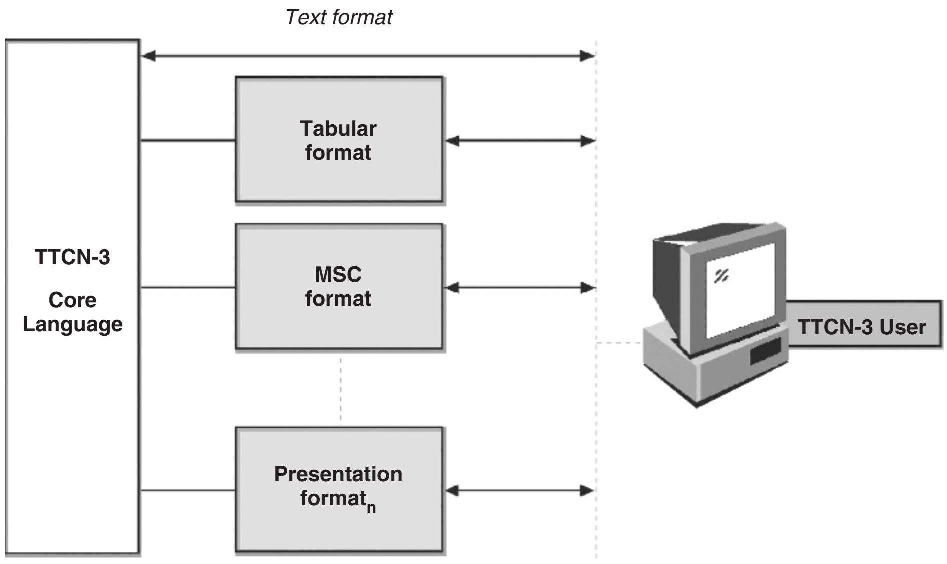
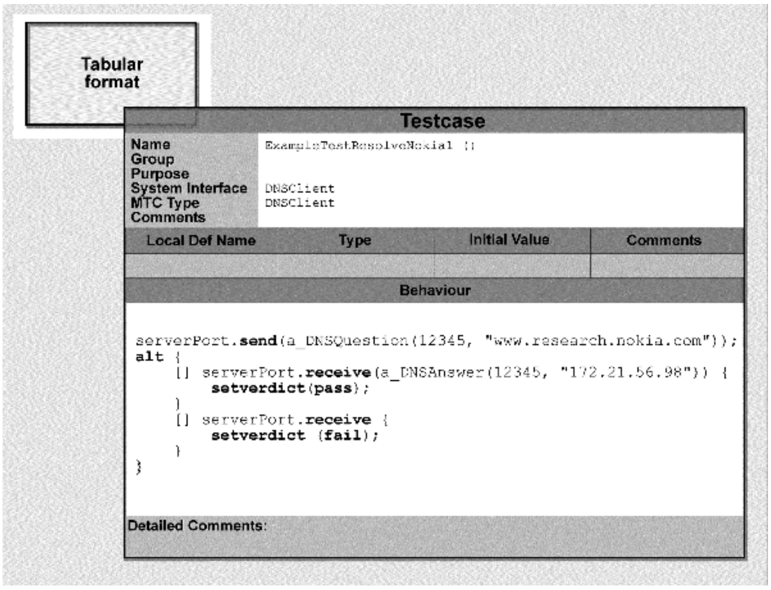


Figure 1.1 TTCN-3 presentation formats.



Testcase

Name	ExampleTestResolveNokia1 ()
Group	
Purpose	
System Interface	DNSClient
MTC Type	DNSClient
Comments	

Local Def Name	Type	Initial Value	Comments

Behaviour

```
serverPort.send(a_DNSQuestion(12345, "www.research.nokia.com"));
alt {
    [] serverPort.receive(a_DNSAnswer(12345, "172.21.56.98")) {
        setverdict(pass);
    }
    [] serverPort.receive {
        setverdict (fail);
    }
}
```

Detailed Comments:

Figure 1.2 An example of the tabular presentation format.

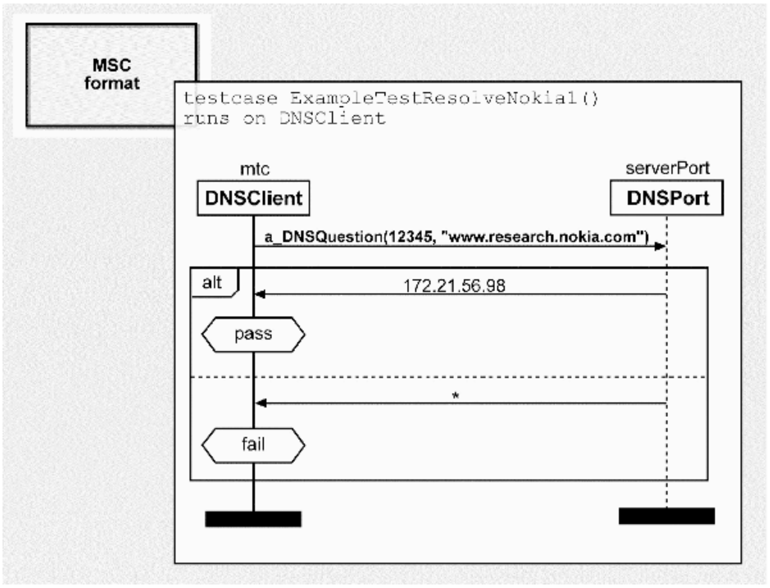

MSC format
testcase ExampleTestResolveNokia1()
runs on DNSClient
mtc
DNSClient
serverPort
DNSPort
a_DNSQuestion(12345, "www.research.nokia.com")
alt
172.21.56.98
pass
*
fail

Figure 1.3 An example of the graphical presentation format.

1.2 The Development of TTCN-3

The direct predecessor of TTCN-3, TTCN-2, was developed by ISO as part of the overall methodology for testing protocol layers in the Open Systems Interconnection (OSI) seven-layer architecture [23]. TTCN-2 was first standardised in the late 1980s by ITU-T [24] and ISO [25]. It has been successfully applied within the area of conformance testing for telecommunications protocols. Nevertheless, a number of problems and shortcomings were limiting its possibilities to be used as a more general purpose testing language. In 1998, ETSI, the European Telecommunication Standardisation Institute, set up the Specialist Task Force (STF) 133 to develop a new improved version of TTCN, taking the known issues into account. This action resulted in the birth of TTCN-3. Over the following two years, the TTCN-3 language was developed with the involvement and input of most of the major tools and telecommunication companies. The official launch of the TTCN-3 language took place in October 2000 at Sophia Antipolis, France.

When developing TTCN-3, four major areas of improvement needed to be considered and addressed in relation to TTCN-2. These areas were productivity, expressive power, flexibility and extensibility.

The aspect of productivity was simply addressed by developing the core language to resemble other well-known, modern programming languages. By making TTCN-3 a textual language, it made it easier for users to edit and learn the new concepts. TTCN-3 also provides significantly extended functionality that makes the language powerful and suitable for a wider range of testing applications. Some of these extensions include better support of new types of testing such as special constructs and features for the testing of IP-based systems and text-based protocols like Session Initiation Protocol SIP [26].

Another major extension provides support for testing systems based on remote procedure calls, using, for example CORBA or web services.

Finally, TTCN-3 is extensible: TTCN-3 has explicit hooks and mechanisms built-in to the language that allow new features and notations to be easily integrated. Some new features are self-contained, for example the integration of IDL [27] and XML [28] type definitions as well as the definition of a common set of documentation tags. New parts in the set of TTCN-3 standards have been defined for these features, see [14–16]. Other new features are more multi-faceted and require the extension of the notation, the operational semantics, as well as other parts of the standard. Behaviour types, type parameterisation, as well as test deployment support are examples of such multi-faceted extensions. These extensions have been defined by separate *extension packages*, including the relevant modifications to the core language, operational semantics and the other parts of the TTCN-3 standards, see [17–19].

Both the introduction of new parts to the standard as well as the definition of extension packages, allow new features to be introduced while keeping the core language stable.

1.2.1 Future Development

TTCN-3 was designed to be a general-purpose testing language that could be used in many application areas. With time, this has spread its usage into many new fields where standardised testing languages have not been used before. Using TTCN-3 in these new areas has generated requests for additions to the language that can provide better support for testing requirements from domains such as real-time testing or performance testing.

TTCN-3 is actively maintained through a well-defined change request process handled by ETSI. The change request process provides a mechanism to balance the needs for stability and backwards compatibility with the calls for extended functionality from new users. Additionally, changes to TTCN-3 are well-documented and the reasoning behind the changes can be followed.

1.3 Summary

In this introduction, we have briefly presented the background and most important concepts of TTCN-3. We have seen that the language has a long history, which originates from the world of telecommunications. In the past decade, the worlds of telecommunications and the Internet have moved much closer together and the systems to be tested are constantly becoming more dynamic and complex in their nature. To meet these new challenges, the existing standardised test language, TTCN-2, was re-designed and extended to result in TTCN-3. The following chapter introduces an example that, even though simple in nature, contains many of the testing issues found in modern communication systems.