1

Introduction

The Testing and Test Control Notation Version 3 (TTCN-3) is an internationally standardized 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.

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 [1–6], filling the gaps like style guide, structuring, and application. In addition, this book points out the dangers and pitfalls of TTCN-3 on the basis of our personal TTCN-3 experience from language standardization, 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 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 considering 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 synchronization 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. With all the language parts described in detail, Chapter 12 rounds off the book by providing a detailed description of how TTCN-3 test systems work in practice. Finally, Chapter 13 can be seen as a utility chapter that provides a collection of code examples and common sense advice. Visit the accompanying web site http://www.wiley.com/go/ttcn-3/ for code samples from the book and links to relevant standards documents.

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 standardized test language [1–6]. 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 [1]. 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.

1.1.1 TTCN-3 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
1.1 TTCN-3 AS A LANGUAGE

1.1 TTCN-3 AS A LANGUAGE

into the core notation while preserving their meaning. This allows the same compiler and run-time execution environment to be used regardless of which presentation format the different tests where specified in. The relationship between presentation formats and the core language is shown in Figure 1.1.

Presentation formats can be standardized formats, as in the cases of the tabular presentation format [2] shown in Figure 1.2 and the graphical presentation format [3]
shown in Figure 1.3, or they can be purely tool-, vendor-, or application area-specific. The standardized tabular presentation format 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.

The graphical presentation format uses an extended version of an MSC-like [9] notation for specifying test scenario behaviour.

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 [10]. TTCN-2 was first standardized in the late 80s by ITU-T [7] and ISO [8]. 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 SIP [11]. Another major extension provides support for testing of procedure-based systems, something that opens the door to software module testing and the testing of CORBA-based systems.

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. Examples of these extension mechanisms can be found in the standardization work involving the IDL [12] and XML [13] integrations into the TTCN-3 language.

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 standardized 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 and performance testing.

TTCN-3 is actively maintained though 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.

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 standardized 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 communicating systems.