A TTCN-3-based Web Service Test Framework

Edith Werner, Jens Grabowski, Stefan Troschütz, Benjamin Zeiss
Software Engineering for Distributed Systems Group,
Institute for Computer Science, University of Göttingen,
Lotzestr. 16–18, 37083 Göttingen, Germany.
{ewerner,grabowski,zeiss}@cs.uni-goettingen.de,
stefan.troschuetz@stud.informatik.uni-goettingen.de

Abstract: The increased usage of Web services for critical applications introduces a growing need for efficient testing approaches to assure their quality. The *Testing and Test Control Notation* (TTCN-3) is a standardised testing language that is well suited for black-box testing of distributed systems such as Web services. Also due to its abstract test specification methodology, it allows easy adaptation to different Web service frameworks or platforms. This paper presents a mapping from the *Web Service Description Language* (WSDL) to TTCN-3 and a corresponding automated translator.

1 Introduction

Web services are a standards-based approach to distributed communication that utilises the *Hypertext Transfer Protocol* (HTTP) and the *Extensible Markup Language* (XML) as technological foundation. Usually, a Web service is expected to have an interface described in a machine-processable format, e.g. the *Web Service Description Language* (WSDL), and to communicate to other systems using SOAP messages via HTTP. Due to the progressing adoption of Web services in business-critical applications, the demand for efficient testing approaches increases.

The *Testing and Test Control Notation* (TTCN-3) is a test specification and test implementation language standardised by the *European Telecommunications Standards Institute* (ETSI) and the *International Telecommunication Union* (ITU). Several authors already recognised the suitability of TTCN-3 for Web service testing and also introduced the idea of deriving abstract test interfaces from a Web services' WSDL description.

In this paper, we present a specification of a mapping between WSDL and TTCN-3 *Abstract Test Suite*s (ATSs) that is both unambiguous and modularly structured. The mapping is implemented and has also been integrated into the commercial TTCN-3 *Integrated Development Environment* (IDE) TTworkbench.

This paper is structured as follows: We start by giving an overview on related research in Section 2. In Section 3, we describe the foundations of SOAP, WSDL, and TTCN-3. Our mapping of WSDL to TTCN-3 is presented in Section 4. Afterwards, in Section 5, the implementation of our mapping and its integration into the development environment TTworkbench are described. In Section 6, we conclude with a summary and an outlook.

2 Related work

There are various tools from industry dealing with Web service testing, many of which are language-specific and sometimes even bound to a specific test platform. For example, in [JA] the authors present how to automate Web service testing using technologies such as JUnit, Apache Commons HttpClient, and XMLUnit, whereas in [McC], the authors automate ASP.NET Web service testing using means provided by the .NET Framework. However, these tools and approaches mostly do not sufficiently cope with the language and platform-independency of Web services.

In [XPS05], the advantages of Web service testing based on TTCN-3 are discussed. The ATS is developed using the knowledge of the WSDL descriptions, models, and the Web service source code while the adapter concept of TTCN-3 enables testing using different client-side implementation languages and different Web service client toolkits by simply switching the TTCN-3 execution platforms and adapters. Anyhow, there is no automation of the mapping involved.

In [SS03], the authors discuss the automated testing of SOAP based Web services by use of TTCN-3 and present a mapping between XML data descriptions and TTCN-3 data to enable the automated derivation of test data. The paper is the first proposal of an automated mapping between *Document Type Definition* (DTD) [W3C06] or *XML Schema Definition* (XSD) types and TTCN-3 type definitions. Nonetheless, the automated generation of TTCN-3 types associated with test components as well as generic test implementation and execution are not considered properly. Additional work on importing XSD types into TTCN-3 [Jea04, JDR04] became the foundation of the forthcoming TTCN-3 Standard Part 9 [ETS07], which was finalised as draft in October 2006 and is going to standardise the use of XSD with TTCN-3.

In [SVR06], the import of WSDL descriptions into TTCN-3 as well as mapping rules between WSDL and TTCN-3 definitions are discussed. It enables the automated derivation of basic TTCN-3 test suites from WSDL descriptions where only specific test data must still be entered manually. Although the approach is evidenced by means of a simple case study, the presented mapping between WSDL and TTCN-3 is incomplete, e.g. it is not considered how TTCN-3 identifiers derived from WSDL can be ensured to be unique.

3 Foundations

In the following, we provide an overview on the Web service technologies SOAP and WSDL as well as key concepts of the test specification language TTCN-3. Language concepts from WSDL, SOAP and TTCN-3 will be highlighted using *slanted type*.

SOAP [W3C03] is a protocol for exchanging XML-based messages in a distributed environment and constitutes the leading standard for the interaction with Web services. The SOAP protocol assumes that every message has a sender, a receiver, and an arbitrary number of so-called intermediaries that process the message and route it to the receiver. The *Body* element carries the core information of a SOAP message and its content must be processed by the receiver. The *Header* element is used to add functionality or additional information to a SOAP message without modifying the core of the message. For example, the header could contain a session identifier or security credentials.

WSDL [W3C01] is an XML-based format for the description of Web services that specifies the exposed interface and location of a Web service as well as how to access it. The overall structure of WSDL documents can be divided into two parts. The abstract part consisting of the *types*, *message*, and *portType* elements describes the interface of the Web service, i.e. the exposed operations and data types. The instantiation part comprising the *service* and *binding* elements describes a concrete implementation of the service interface at a network endpoint, i.e. where the Web service is located and how its operations can be accessed.

TTCN-3 is a multipart standard [ETS07] which consists of a core language (part 1 of [ETS07]), a number of representation formats (parts 2 and 3), the operational semantics (part 4), the *TTCN-3 Runtime Interface* (TRI) for the connection to the *System Under Test* (SUT) (part 5), the *TTCN-3 Control Interface* (TCI) for the connection to the test environment (part 6) as well as several language bindings (parts 7–9). The TTCN-3 core language is similar to typical programming languages, but provides additional features dedicated to testing such as built-in data matching, test verdicts, timer handling, or the concurrent execution of test components. The test behaviour is specified by *test cases*, *functions*, and *altsteps*. Functions and altsteps are used to specify and structure test behaviour. A *default* is an altstep that when activated is implicitly included at the end of alternative behaviour and is typically used for handling errors and timeouts. A *test case* is a special kind of function that always returns a test verdict. The different TTCN-3 elements are structured into *modules*. Besides the test definitions, a module can contain a control part that is the behavioural entry point when a TTCN-3 module is executed. To increase the flexibility of the modularisation, a module can also import definitions from other modules.

4 A mapping of WSDL to TTCN-3

To test a software system using a test specification language like TTCN-3, an ATS has to be created which is then compiled and finally executed on the test system. In the case of a Web service, an ATS skeleton can be derived from the interface definition that is given in WSDL as depicted in Figure 1. For the test execution, a SUT adapter and codec have to be provided to instantiate the ATS for the target system. The specification of abstract test specifications simplifies reusability and portability, but also implies that the mapping is involved at two different levels in a TTCN-3 test system: First, the information by the provided WSDL documents must be translated to TTCN-3 and the TTCN-3 type system to enable the development of Web service test suites in the TTCN-3 core language. Second, the TTCN-3 adaptation layer must implement the reversed mapping to realise compatibility with the original interface definition. Thus, our mapping is a detailed rule set that fulfils the requirement to be unambiguous in both directions. In this paper, we only highlight the interesting aspects of the mapping.

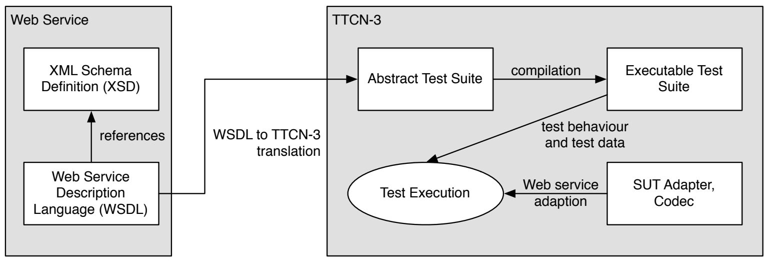

Figure 1: The Web service test framework

4.1 Communication paradigm

An essential choice that influences the mapping rules is whether the TTCN-3 ATS should use message-based or procedure-based communication for the interaction with Web services. In the related research work, both communication ways are used almost equally often. The mapping presented in this paper generates a TTCN-3 ATS that uses message-based communication. This decision is based on the practical observation that Web services typically use message-based communication themselves. Also, the input, output, and fault messages described by the WSDL document can be mapped directly to a message type definition in TTCN-3. Lastly, the *call* statement, a procedure-based communication operation, is not allowed to reference *altsteps* and during its evaluation, all active *defaults* are ignored. Therefore, common behaviour such as the reaction on inactivity of the system under test must be specified repeatedly.

4.2 Identifier conversion

Each of the two languages, WSDL and TTCN-3, define a set of rules and naming conventions for identifiers to assure their uniqueness and readability. In our setting, the WSDL identifiers are translated into TTCN-3 identifiers for the generation of the ATS and have to be reverted when the test suite is instantiated and executed. To allow this, TTCN-3 identifiers are derived from the name attribute of WSDL elements. However, as WSDL allows a wider range of characters in identifiers than TTCN-3, these additional characters are replaced by reversible escape sequences to obtain a TTCN-3 compatible name.

In WSDL, scoping and overloading of identifiers is widely used, e.g. it is possible to import elements with equivalent names from another document or from another XSD. To generate unique TTCN-3 identifiers, the WSDL identifiers are prefixed with their type and scope. The prefixes employed are partly suggestions made by [WDT$^+$05] which are supplemented with a set of prefixes closely bound to the domain of Web service testing.

4.3 Mapping of WSDL language constructs

Our mapping rules make wide use of modularisation, which enhances the readability, reusability and maintainability of the resulting test suite. In addition, the modularisation reduces the number of rules that target the elimination of name ambiguities.

The WSDL *types* element encloses data definitions, usually given using the XSD, that are relevant for the exchanged messages. The mapping of XSD data type definitions to TTCN-3 is defined in the forthcoming TTCN-3 standard part 9 [ETS07], which is adopted with some minor changes and additions. For example, rather than coding a unique qualified form for each identifier, a module is used to represent a target namespace of any enclosed XSD instances. This as well improves the overall test suite structure. The components defined by each XSD are mapped to TTCN-3 type definitions, which are located in the module that has been created for each corresponding target namespace of the schema.

The WSDL *portType* element is also mapped to a module. Within the module, a TTCN-3 *group* is defined for each operation element of the respective WSDL port type. Inside of each group, a TTCN-3 *record* type is defined for every input, output, and fault element of the respective WSDL operation. At each representation of an input element, the first field of the according TTCN-3 *record* type is named soapbinding and is of the TTCN-3 SoapBinding type, so that each input message can be accompanied with the proper SOAP binding information upon test execution. The SoapBinding type is predefined inside a separate module containing base definitions. The other fields of the TTCN-3 *record* types are derived from the WSDL message that is referenced by the input, output, or fault element to which they correspond. Finally, each TTCN-3 module representing a WSDL port type contains a TTCN-3 *port* type which uses message-based communication and allows sending and receiving of the defined *record* types.

For each WSDL *port* and its associated *binding* element, a TTCN-3 module is created which contains TTCN-3 constants with SOAP binding information for every described operation. Furthermore, additional local constants are created inside this module that hold information common to all operations defined by the respective WSDL port, e.g. SOAP version and *Uniform Resource Identifier* (URI). For each operation, a record of type SoapBinding is declared which is composed of references to the constants holding common SOAP binding information and the invocation details specified by the attributes of the SOAP binding elements.

The WSDL *service* element is mapped to a collection of modules. Each module contains basic test behaviour and execution control for testing a WSDL port type referenced by the service via port and binding and a component type definition that specifies the associated WSDL port type. In addition, test cases and necessary templates for testing the operations defined by the WSDL port type are created. The module control part executes the defined test cases for each WSDL port of the service which references the WSDL port type.

The complete mapping including all technical details and an extensive example can be found in the master's thesis of Stefan Troschütz [Tro07].

5 Implementation

In order to execute our generated test suites, we used the commercial tool TTworkbench [TT] as development environment and test system. In addition, we implemented an integration into the Eclipse-based TTworkbench making our translation tool easy to use. As of October 2007, the TTworkbench tool provides WSDL support via an own plug-in, which was developed concurrently to the one presented here. As far as we know, the main difference in the mapping is the used communication paradigm as well as the modular organisation of our mapping.

The automated mapping starts with the mapping of the WSDL *types* elements. The XSDs contained in those elements are retrieved and mapped to TTCN-3 modules. To ease dealing with the possible interdependencies of the XSD components, all schemas are mapped at once. Next, the WSDL *portType* elements are processed. For every WSDL port type defined in the currently mapped WSDL document, a corresponding TTCN-3 module is generated. Finally, the WSDL *service* and *port* elements are mapped. For each *service* element, the enclosed port elements are associated with the corresponding port type definitions. During the mapping procedure, TTCN-3 identifiers are generated mainly by applying rewriting rules on patterns matched by regular expressions.

5.1 Test adapter and codec

In order to run the *Executable Test Suite* (ETS) generated by TTworkbench's TTthree compiler against an actual SUT, a specific SUT adapter and codec have to be implemented. Both SUT adapter and codec inherit from the abstract implementations provided by TTworkbench and override the methods of the TRI and the TCI that are invoked by the test system when executing a test suite. In order to deal with SOAP messages and communicate with a Web service under test, the SUT adapter and codec make use of the *Application Programming Interface*s (APIs) provided by the Apache Axis distribution [Axi]. The codec uses the implementation of the identifier mappings for proper encoding and decoding of SOAP messages and for the recovery of element and attribute names from TTCN-3 identifiers.

5.2 Test composer

Since a WSDL document specifies neither semantics nor behavioural context for the described operations, complex test scenarios cannot be derived from it. Therefore, the automatically generated ATS contains only simple test cases for testing the Web service operations individually. To ease the specification of new and more complex test cases, we provide a dialogue-based means to freely combine the generated operation calls with user-defined data templates and adequate error- and verdict-handling.

First, the user has to select a TTCN-3 module representing a WSDL service where the new test scenario should be inserted. The test component on which the behaviour for testing the corresponding WSDL port type is executed is determined and analysed and the associated operations are presented grouped per port. From this, the user can specify the sequence of Web service operations that should be invoked in the course of the test scenario. In the second step, the user has to specify identifiers for the TTCN-3 definitions used by the new test scenario, including a test case, templates for all input, output, and fault messages of the selected Web service operations, and a group that encloses the former test scenario definitions. All specified identifiers have to be unique within the context of the target module and are required to adhere to the naming conventions to enable their translation into WSDL. Finally, the new test scenario is created and inserted into the target module. For that purpose, a syntax tree representing the test scenario group and all enclosed TTCN-3 definitions is constructed, transformed into TTCN-3 core notation, and inserted into the target module at the place initially specified by the user.

The behaviour of composed test cases is determined as follows. For each operation, first the template for the input message is transmitted via the corresponding port. If the operation is a one-way operation, i.e. the Web service under test will not respond, the test verdict is set to *pass*. Otherwise, the timer owned by the test component is started to safeguard the test behaviour against inactivity of the Web service under test. After starting the timer, an *alt* statement specifies several alternatives. The first alternative is executed if the template defined for the output message of the Web service operation is duly received. It will stop the timer and set the test verdict to *pass*. If the operation specifies fault messages, further alternatives are defined for the receipt of their corresponding templates. Each of those alternatives will also stop the running timer, but set the test verdict to *fail*. At last, a *default* is activated in order to catch the receipt of unexpected messages or a timeout.

6 Summary and outlook

By utilising the WSDL description of a Web service, it is possible to derive an ATS in TTCN-3 that provides the necessary kind of abstraction from the actual implementation platform and the communication details. With the comprehensive mapping specification that we have created, Web service testing in TTCN-3 becomes more reliable, comprehensible, consistent and efficient. The automated translation enables the test developer to concentrate on actual Web service test development from the start and avoids typical humanly errors. Our automated WSDL to TTCN-3 translator is integrated into the TTCN-3 development environment TTworkbench as an Eclipse plug-in.

The presented mapping for Web service testing is one ingredient of a fully-fledged Web service development cycle that we are currently developing. It will support Web service development from design to deployment. We also plan the attempt to couple the existing test case composer with a process description like the *Business Process Execution Language* (BPEL) to automatically generate more advanced test cases.

References

[Axi] Web Services - Axis. Apache <Web Services/> Project, http://ws.apache.org/axis/. [January 11th 2008].

[ETS07] ETSI Standard (ES) 201 873 V3.2.1: The Testing and Test Control Notation version 3; Parts 1-9. European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, also published as ITU-T Recommendation series Z.140, 2007.

[JA] M. Jadhav and M. Ahmed. Automate Web service testing, Part 2: Test a Web service with XMLUnit. IBM developerWorks, http://www-128.ibm.com/developerworks/edu/ws-dw-ws-soa-autotest2.html [January 11th 2008].

[JDR04] D.-M. Jeaca, G. Din, and A. Rennoch. Importing XMLSchema datatypes into TTCN-3. In *Proceedings of the 3rd Workshop on System Testing and Validation (SV04)*, 2004.

[Jea04] D.-M. Jeaca. XML Schema to TTCN-3 Mapping: Importing XMLSchema datatypes into TTCN-3. Master's thesis, University Politechnica Bucharest, Department of Computer Science, Bucharest, Romania, 2004.

[McC] J. McCaffrey. Test Run: Automate Your ASP.NET Web Services Testing. MSDN Magazine, http://msdn.microsoft.com/msdnmag/issues/05/03/TestRun/ [January 11th 2008].

[SS03] I. Schieferdecker and B. Stepien. Automated Testing of XML/SOAP based Web Services. In *Proceedings of the 13th Fachkonferenz der Gesellschaft für Informatik (GI) Fachgruppe "Kommunikation in Verteilten Systemen" (KiVS)*, 2003.

[SVR06] Ina Schieferdecker, Diana Vega, and Cosmin Rentea. Import of WSDL Definitions in TTCN-3 Targeting Testing of Web Services. In *Proceedings of the 9th International Conference on Integrated Design and Process Technology (IDPT)*, 2006.

[Tro07] Stefan Troschütz. Web Service Test Framework with TTCN-3. Master's thesis, ZFI-BM-2007-14, University of Göttingen, Institute for Computer Science, Göttingen, Germany, 2007.

[TT] Testing Technologies TTworkbench. http://www.testingtech.de/products/ttwb_intro.php. [January 11th 2008].

[W3C01] Web Services Description Language (WSDL) 1.1. World Wide Web Consortium (W3C) Note, http://www.w3.org/TR/2001/NOTE-wsdl-20010315, 2001.

[W3C03] SOAP Version 1.2 Part 0: Primer. World Wide Web Consortium (W3C) Recommendation, http://www.w3.org/TR/2003/REC-soap12-part0-20030624/, 2003.

[W3C06] Extensible Markup Language (XML) 1.0 (Fourth Edition). World Wide Web Consortium (W3C) Recommendation, http://www.w3.org/TR/2006/REC-xml-20060816/, 2006.

[WDT$^+$05] C. Willcock, T. Deiß, S. Tobies, S. Keil, F. Engler, and S. Schulz. *An Introduction to TTCN-3*. Wiley, New York, 2005.

[XPS05] P. Xiong, R. L. Probert, and B. Stepien. An Efficient Formal Testing Approach for Web Service with TTCN-3. In *Proceedings of the 13th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2005.