Ensuring Interoperability between Network Elements in Next Generation Networks

Matthias Bormann

A portfolio comprised of two projects and submitted in partial fulfilment of the requirements of the University of Glamorgan for the degree of Master of Philosophy

Faculty of Advanced Technology
University of Glamorgan
Pontypridd
CF37 1DL

June 29, 2010

Certificate of Research

This is to certify that, except where specific reference is made, the work described in this thesis is the result of the candidate. Neither this thesis, nor any part of it, has been presented, or is currently submitted, in candidature for any degree at any other University.

Signed ....
Candidate

Signed ....
Director of Studies

Date ....

Abstract

Next Generation Networks (NGNs), based on the *Internet Protocol* (IP), implement several services such as IP-based telephony and are beginning to replace the classic telephony systems. Due to the development and implementation of new powerful services these systems are becoming increasingly complex.

Implementing these new services (typically software-based network elements) is often accompanied by unexpected and erratic behaviours which can manifest as interoperability problems. The reason for this caused by insufficient testing at the developing companies. The testing of such products is by nature a costly and time-consuming exercise and therefore cut down to what is considered the maximum acceptable level.

Ensuring the interoperability between network elements is a known challenge. However, there exists no concept of which testing methods should be utilised to achieve an acceptable level of quality. The objective of this thesis was to improve the interoperability between network elements in NGNs by creating a testing scheme comprising of three diverse testing methods: conformance testing, interoperability testing and post-hoc analysis.

In the first project a novel conformance testing methodology for developing sets of conformance test cases for service specifications in NGNs was proposed. This methodology significantly improves the chance of interoperability and provides a considerable enhancement to the currently used interoperability tests. It was evaluated by successfully applying it to the Presence Service.

The second report proposed a post-hoc methodology which enables the identification of the ultimate causes for interoperability problems in a NGN in daily operation. The new methods were implemented in the tool IMPACT (IP-Based Multi Protocol Post-hoc Analyzer and Conformance Tester), which stores all exchanged messages between network elements in a database. Using SQL queries, the causes for errors can be found efficiently.

Overall the presented testing scheme improves significantly the chance that network elements interoperate successfully by providing new methods. Beyond that, the quality of the software product is raised by mapping these methods to phases in a process model and providing well defined steps on which test method is the best suited at a certain stage.

Contents

CHAPTER 1

Introduction

Modern communication systems, based on the *Internet Protocol* (IP), are becoming increasingly complex. Basic services, such as IP telephony and messaging, have already been implemented and are beginning to replace the classic telephony systems (e.g. in companies). In particular the integration with computer-based applications typically used in an office environment gives IP-based telephony an advantage over the classic approach.

In recent years the evolution of these so called *Next Generation Networks* (NGNs) has been further improved by the development and implementation of new services. These new services, often the combination of already existing services, are getting more powerful.

However, from an end user's perspective new products (typically software implemented in network elements) implementing new technologies can often be accompanied by unexpected and erratic behaviours which can manifest as interoperability problems. These errors have to be reported to the product manufacturers and the customers have to wait for updates and fixes.

From the manufacturers point of view new products have to have shorter and shorter development cycles to reduce time to market and thus outpace competing companies. The testing of such products is by nature a costly and time-consuming exercise and therefore cut down to what is considered the maximum acceptable level.

The software developer, situated between end user's and company's interests, is aware of the impact of insufficient testing on the end user. Although testing methodologies and tools are provided, i.e. by standardisation bodies, the developer's hands are usually tied by the instructions of the companies developing the products.

There are several testing methodologies, such as conformance or interoperability testing, each inspecting different aspects of a communication software. Figure 1.1 depicts a selection of such testing methodologies and their current field of application. While interoperability testing is a common and widespread approach, conformance testing is only used for testing protocol entities so far and, due to high costs, applied rarely.

Another important phase in the life cycle of a developed communication network element is the time in operation at the customer. Even when testing thoroughly in a laboratory under controlled conditions it is unlikely that every possible combination of tests could be tested before release. A network element outside of a dedicated test environment is open to a bigger variety of faulty effects, e.g. when reaching its load limits. To fix these errors a quick and reliable detection of their cause is of utmost importance for the customer's satisfaction. For this purpose additional methods – in this thesis called post-hoc analyses – as well as dedicated tools are needed.

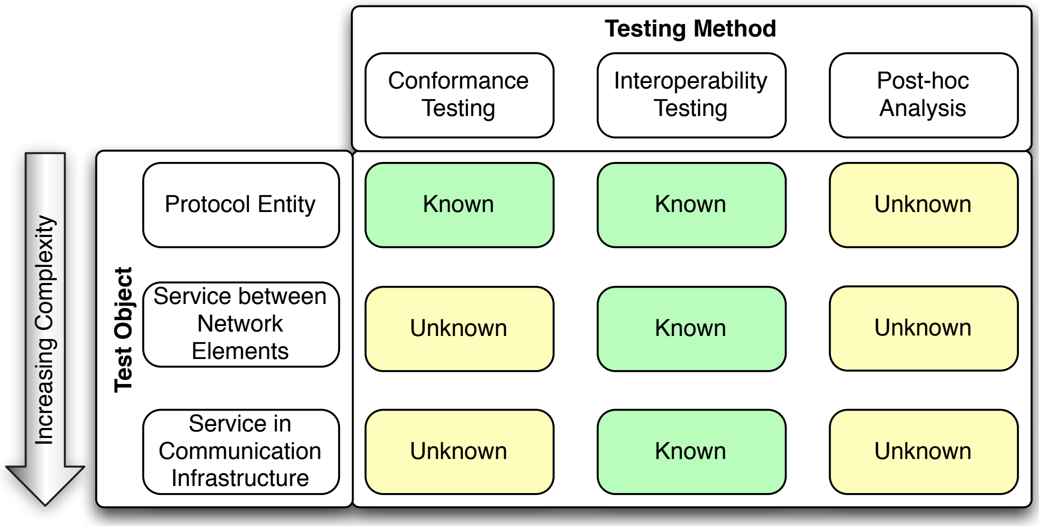

Figure 1.1: Methodologies Currently Existing for Testing of Network Elements in Communication Systems. Conformance testing is done only for protocols but not yet for services. Interoperability testing is a well established methodology and the best current practice for ensuring the interoperability between network elements in a communication system. Methodologies for post-hoc analyses are not yet available and the causes of errors between network elements in operation are typically detected manually.

All in all, ensuring the interoperability between network elements is a known and accepted challenge. However, there exists no concept of which testing methods, and to what extent, should be utilised to achieve an acceptable level of quality.

The overall objective of this thesis is to improve the interoperability of different, interacting network elements in Next Generation Networks (NGNs) by proposing a new test strategy. To this end the testing scheme in figure 1.1 has to be completed by providing the missing test methods. The main objective can be divided into:

- Critical analysis and evaluation of existing test methods
- Creation of a conformance testing methodology for complex protocols and services
- Design of methodologies for efficiently detecting the errors in operation, and their causes
- Evaluation of the new methodologies to selected protocols and services in Next Generation Networks
- Increasing the test coverage – the confidence that network elements work as expected – by combining the three methodologies and providing a concept for their systematic application

CHAPTER 2

Research Objectives and Results

This chapter presents the research objectives and the results of the two conducted projects and describes their relation to each other.

2.1 Report 1

Conformance Testing of Network Elements in Next Generation Networks Implementing Protocols and Services

When testing a network element on conformance to its protocol specification, it is examined whether all the requirements, so called test purposes, of the protocol are met. A test case is a set of inputs and expected results developed to check if a test purpose, a certain requirement, is met by the network element. The actual testing of the network element, the *Implementation under Test* (IUT), is conducted by using a dedicated test system which sends messages to the IUT according to the test case's description. The received messages from the IUT during a test case are evaluated and a *verdict*, which states that an implementation has passed or failed the test, is computed by the test system. If the outcome of a test cannot be determined to be pass or fail, the verdict is set to inconclusive.

The process for developing conformance test cases for protocols has been standardised by the ITU (*International Telecommunication Union*), the ISO (*International Organization for Standardization*), and the ETSI (*European Telecommunications Standards Institute*). It has been applied to protocols such as the Session Initiation Protocol (SIP) and proven reliable and efficient. Although conformance testing can never guarantee an error-free software, it ensures that the implementation is consistent with its specification with a reasonable degree of confidence [1].

With the increasing demand for new and more complex services in communication networks, additional protocols, based on SIP, are being designed. Conformance test cases are necessary for these newly developed protocols as well. While the process for conformance testing has successfully been applied to developing test cases for communication protocols such as SIP, it has not been used so far for testing complex communication services [2].

The objectives of the first project are (figure 2.1):

- Development of a methodology for conformance testing of complex services
- Application of this methodology to the *IP Multimedia Subsystem's* (IMS) *Presence Service* as a prominent example for a service in a Next Generation Network
- Implementation of a conformance test suite for the Presence Service

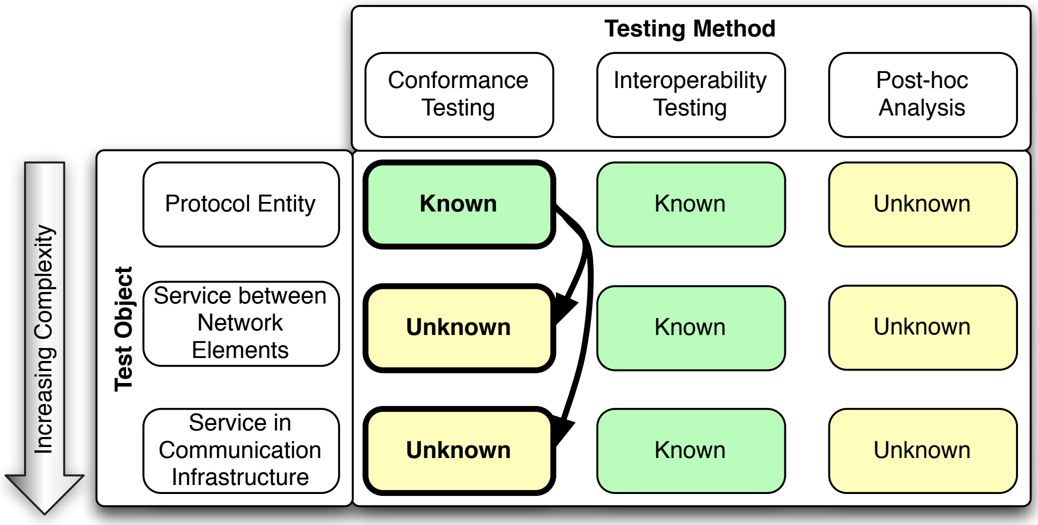

Figure 2.1: Research Objectives for Project 1. Development of new conformance testing methodologies for ensuring interoperability between network elements implementing complex protocols and services.

The challenges of the first project are:

- Complexity of the standardisation process of the IMS and the Presence Service
- Multiple protocol specifications are needed for defining a service

- Reference implementations for the evaluation of developed test cases are typically not available
- Requirement to test network elements with multiple interfaces implementing different protocols

The result of the first project is a new conformance testing methodology, that extends the currently existing testing methodology defined by the European Telecommunications Standards Institute [3]. The approach takes into account the above mentioned challenges and defines rules and guidelines to develop conformance test cases for complex protocols and services in NGNs.

The newly developed methodology was scrutinised by applying it to the IMS' Presence Service, leading to an executable set of conformance test cases which was designed and implemented. The developed test suite is commercially available [4].

2.2 Report 2

Evaluation of Impact of Test Methods for Ensuring Interoperability between Network Elements in Next Generation Networks

Ensuring the interworking between all network elements in Next Generation Networks is an important aspect. For this purpose, conformance and interoperability testing are common approaches. Both testing methods are typically conducted in a lab under controlled conditions. However, several errors of network elements in communication networks can only be detected while being in operation.

When users experience an erroneous or unexpected behaviour during production usage (e.g. cancelled calls in a communication network) they typically report it to the network operator or network element manufacturer, who can then reproduce the scenario the error was encountered in and record all exchanged messages. In a next step the trace of the exchanged messages, so called *protocol data units* (PDUs), is analysed manually by defining filters. However, the number of exchanged messages can be very large and searching for errors – and their cause – is a tedious process.

The objectives of the second project are (figure 2.2):

- Comparison of conformance and interoperability testing methods that ensure the interoperability in a communication network
- Development of post-hoc analysis methodologies for the efficient detection of faulty

message sequences

- Improvement of the interoperability between the interaction of network elements by systematically combining the three different testing methodologies

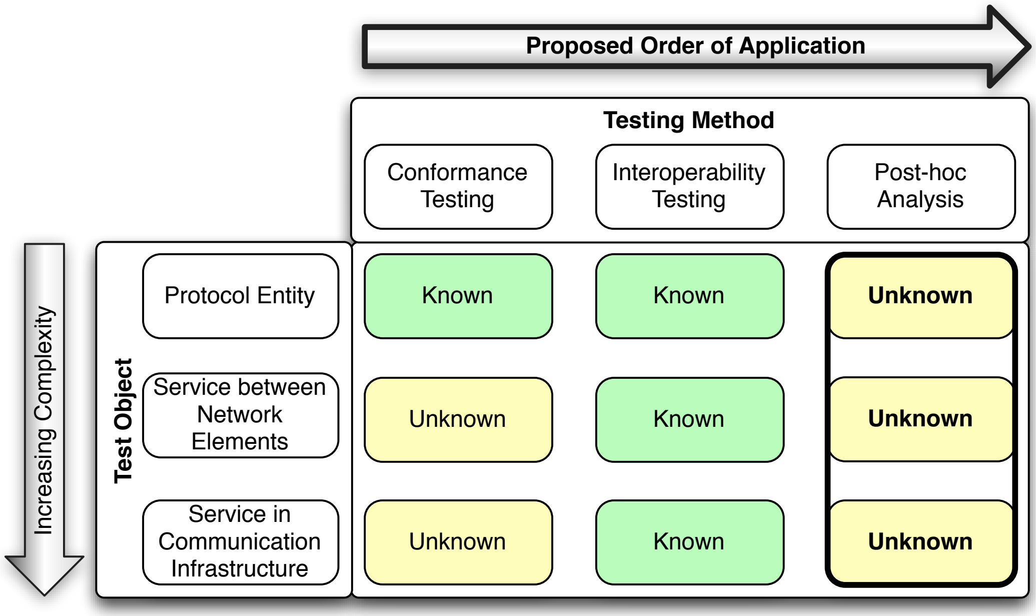

Figure 2.2: Research Objectives for Project 2. Development of methodologies for finding errors and their causes between network elements in operation. Combined consecutive application of the three presented testing methodologies should lead to a further improvement of interoperability.

Challenges of the second project are:

- Development of a method for logging of all exchanged protocol data units (PDUs) between the network elements
- Design of methodologies for selecting faulty message sequences out of traces with many PDUs
- Comparison of the three presented testing methodologies
- Selection of a concept or model to order the different methodologies to increase the interoperability between the network elements

The result of the second project is a set of methodologies for analysing the interaction between network elements during operation. PDUs are recorded and stored in a relational database. Using the *Structured Query Language* (SQL) any combination of message sequences in a trace can efficiently be selected. Furthermore, conformance and interoperability analyses are formulated as SQL queries.

For a further improvement of the interoperability between network elements in NGNs the three testing methodologies were ordered by mapping each of them to the V-Model, a system development model which was designed to simplify the understanding of the development of complex systems.

CHAPTER 3

Conclusions

In this thesis testing methodologies currently used for ensuring the interoperability between network elements in *Next Generation Networks* were examined. A scheme has been set up (figure 1.1) which identifies several areas where testing methodologies are absent. The missing gaps have been filled with the two research projects presented here.

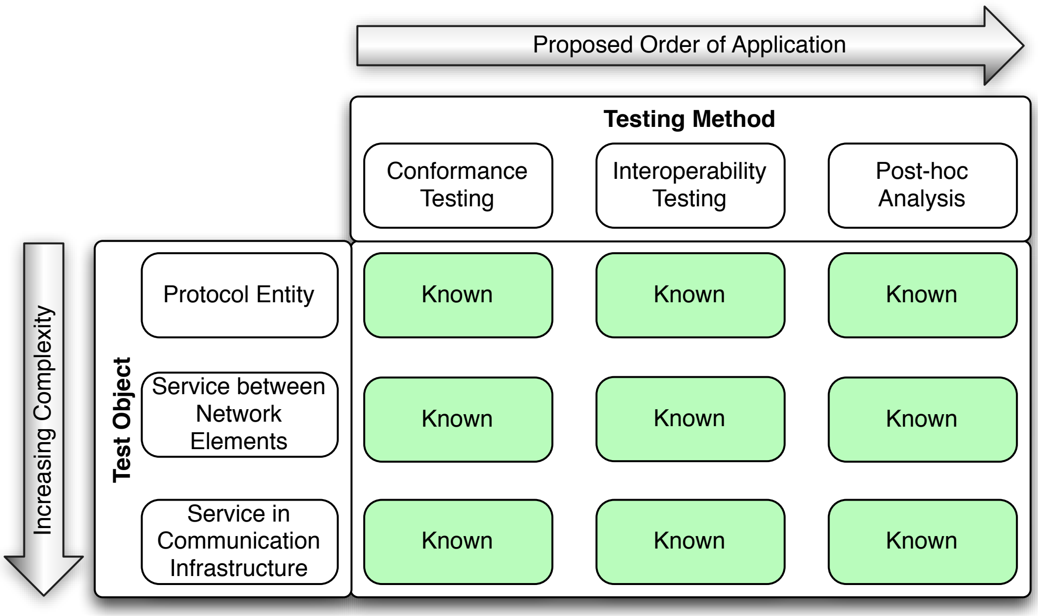

Figure 3.1: Achievements of Project 1 and Project 2. The missing methodologies as indicated in figure 1.1 have been developed in this thesis. As an additional result a systematic order of their application is proposed.

The goal of the first project was to develop a conformance testing methodology for complex protocols and services in NGNs. This was achieved by extending ETSI's process on conformance testing for protocols. As an example a set of conformance test cases for the *IP Multimedia Subsystem's* (IMS) *Presence Service* has been successfully implemented.

In the second project a concept for recording and analysing exchanged *protocol data units* (PDUs) between network elements in a NGN was presented by using the tool IMPACT (IP-based Multi Protocol Post-hoc Analyzer and Conformance Tester). IMPACT stores the PDUs in relational database. Analyse functions using SQL have been implemented for an efficient detection of errors in traces with many PDUs. Furthermore, a systematic combination of the three presented testing methodologies was proposed mapping them to phases in the V-Model.

The two projects were successfully completed and the objectives associated with each project were achieved. With the complete testing scheme (depicted in figure 3.1) a novel concept offering distinct advantages over currently applied methods by the industry ensuring the interoperability between network element in NGNs has been demonstrated. The overall testing process for network elements in NGNs has been improved considerably. The confidence that two independently developed products interoperate as expected when tested according to this scheme has increased significantly. The newly developed set of conformance test cases for the Presence Service as well as IMPACT's analysis functions have already been a great asset for improving the interoperability testing.

3.1 Contributions

This thesis constitutes a significant contribution to knowledge in the field of software testing in general and in particular in the field of protocol testing. The contributions are:

1. Improvement of the interoperability between network elements in Next Generation Networks by providing a novel testing scheme comprising of several distinct testing methods. The missing test methods were designed:

    a) Creation of a conformance testing methodology for testing complex services in NGNs.

    b) Creation of a post-hoc method for the efficient detection of interoperability problems in NGNs in operation.

c) Proposition of an approach to systematically combine the presented testing methods taking advantage of their differences.

2. Contribution to software testing by applying the new conformance testing methods to the Presence Service's specifications. As a result a Presence Test Suite was developed which is commercially available (see page 15).

3. The post-hoc testing concept was implemented in the tool IMPACT which is currently used for further research projects at the University of Applied Sciences Ostfalia (see `http://ostfalia.de/cms/de/pws/wermser/fue/`).

4. Contribution to literature in the field of protocol testing in the proceedings of several conferences (see page 14 ff.).

3.2 Further Work

The presented testing scheme is flexible and extensible. In the future, further relevant testing methodologies such as stress testing can be included. The scheme can be further extended to provide for the testing of more complex systems (i.e. interaction between complete communication systems) (figure 3.2). Furthermore, the newly developed methodologies – although applied so far in the field of communication systems have the potential to be used in other fields of application such as in-vehicle or smart metering networks.

Another important aspect which has to be scrutinised in the future is, how interoperability is related to performance and security as well as the impact of these two aspects on the presented testing scheme. Performance and security are very important but opposing aspects in Next Generation Networks. Often security is only related to the software's functionality. However, this is a common failure and leads to systems with an insufficient performance and is experienced as poor usability [5].

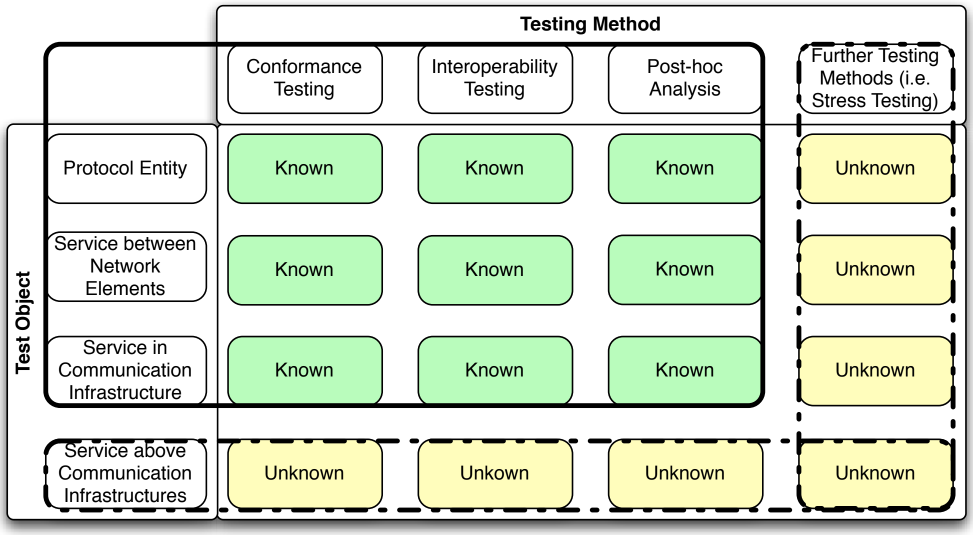

Figure 3.2: Extensions of the Proposed Testing Scheme. In the future further relevant testing methodologies such as stress testing could be included by adding an extra column to the scheme. Testing of more complex systems (i.e. interaction between complete communication systems) is represented as an additional row. For both extensions the missing methodologies have to be developed.

In an IP-based network for example it is very important that personal data (e.g. VoIP calls) is securely encrypted. Designing and implementing a secure VoIP infrastructure is a challenging task. This is due to the trade-off between security and performance. This issue exists for data networks in general and in particular in voice networks, where quality of service is closely tied to performance.

Transmission errors and delays can distort or even lead to the cancellation of VoIP calls. Such errors may reduce the quality of the voice sound on one or both ends, create echo effects or drop calls entirely, which is not acceptable for organisations that depend on phone calls to conduct business. Security mechanisms on an IP-based network almost always cause some overhead that affects the performance. When data is being transmitted this may not necessarily be noticed. However, the accumulated delays caused by encrypting and decrypting packets can adversely affect the quality of the call.

A limited amount of work has been done on the usability of security systems and in particular on the close relationship that exists between usability and security. The design of secure systems with a good performance raises crucial questions concerning how to solve conflicts between security and performance aspects. The fundamental question is how to ensure performance without compromising security and vice-versa.

Considering this trade-off when designing new systems, dedicated methods are needed. For this purpose some methods are already available such as the *Architecture Tradeoff Analysis Method* (ATAM) [6] or the *Security Usability Symmetry* (SUS) [5]. However, in the long run additional testing methods – or at least modifications to the existing ones – are needed to measure the designed systems in terms of their performance and security, which is a challenging objective for the near future.

Contributions

Contributions to Literature in the Field of Protocol Testing

1. M. Bormann, D. Hartmann, and D. Wermser, "Sicherstellung der Interoperabilität von Netzelementen in IKT-Infrastrukturen im operativen Betrieb," *Science Days 2009 Nachrichtentechnik im Informationszeitalter an der HfTL*, November 2009.
2. D. Wermser, M. Bormann, and D. Hartmann, "Ensuring Interoperabilty between Network Elements in Next Generation Networks," *ZVEI-Elektronik Wireless Congress 2009*, October 2009.
3. M. Bormann, D. Wermser, and R. Patz, "Conformance Testing of Complex Services Exemplified with the IMS' Presence Service," in *Proc. IEEE NGMAST 2009*, September 2009.
4. M. Bormann, D. Hartmann, and D. Wermser, "Gegenüberstellung und Anwendung verschiedener Testverfahren zur Sicherstellung der Interoperabilität von Netzelementen in Next Generation Networks," *VDE ITG-Fachtagung Mobilkommunikation*, May 2009.
5. M. Bormann, V. Sandhaus, and D. Wermser, "IMS und Presence Service als Plattform für die Entwicklung mobiler Dienste - Protokolle und Testverfahren," *Science Days 2008 Verteilte Anwendungen an der HfTL*, November 2008.
6. M. Bormann, V. Sandhaus, and J. Fricke, "Conformance Testing for SIP-based Communication Services and IMS," *Conference for Model-Based Validation of in-Vehicle Networks; aGiP Research project*, June 2008.
7. M. Bormann, V. Sandhuas, and D. Wermser, "Presence Service im IMS - Architektur, Protokolle und Testverfahren," *VDE ITG-Fachtagung Mobilfunk*, May 2008.

8. D. Wermser, M. Bormann, and V. Sandhaus, “IMS: Protocols, Services and Assuring Conformance,” *ZVEI-Elektronik Wireless Congress 2007*, October 2007.

9. V. Sandhaus, M. Bormann, and D. Wermser, “Spezifizierung und Realisierung von Konformitätstests für anwendungsorientierte Kommunikationsprotokolle in Mobilfunknetzen mittels der Testsprache TTCN-3,” *VDE ITG-Fachtagung Mobilfunk*, May 2007.

Contributions to Software Testing

- Testing Technologies IST GmbH, “TTsuite-Presence,” 2008. [Online]. Available: http://www.testingtech.com/solutions/ttsuite-presence_w.php

Bibliography

[1] S. Moseley, S. Randall, and A. Wiles, "Experience within ETSI of the combined roles of conformance testing and interoperability testing," in *The 3rd Conference on Standardization and Innovation in Information Technology*, October 2003, pp. 177–189.

[2] M. Schmidt, A. Wilde, A. Schülke, and H. Costa, "IMS Interoperability and Conformance Aspects," *IEEE Communications Magazine*, vol. 45, pp. 138–142, March 2007.

[3] European Telecommunications Standards Institute, "Methods for Testing and Specification (MTS); Protocol and profile conformance testing specifications; Standardization methodology," European Telecommunications Standards Institute, ETS 300 406, April 1995.

[4] Testing Technologies IST GmbH, "TTsuite-Presence," 2008. [Online]. Available: http://www.testingtech.com/solutions/ttsuite-presence_w.php

[5] C. Braz, A. Seffah, and D. M'Raihi, "Designing a trade-off between usability and security: A metrics based-model," in *Human-Computer Interaction - INTERACT 2007*, September 2007, pp. 114–126.

[6] Kazman, R. and Klein, M. and Barbacci, M. and Longstaff, T. and Lipson, H. and Carriere, J. , "The architecture tradeoff analysis method," in *Proc. Fourth IEEE Int. Conf. Engineering of Complex Computer Systems ICECCS '98*, 1998, pp. 68–78.

List of Figures

Glossary

Term	*Definition*
Conformance Test Suite	A complete set of test cases, possibly combined into nested test groups, that is needed to perform dynamic conformance testing for one or more OSI protocols.
Conformance Testing	Testing the extent to which an IUT is a conforming implementation.
Executable Test Case	A realisation of an abstract test case.
Implementation Under Test (IUT)	An implementation of one or more OSI protocols in an adjacent user/provider relationship, being that part of a real open system which is to be studied by testing.
Multi-protocol Testing	Testing of more than one protocol within the IUT by means of test cases which have test purposes which cover conformance requirements that relate to more than one protocol.
System Under Test (SUT)	The real open system in which the IUT resides.
Test Case	An abstract or executable test case.
Test Verdict	A statement of pass, fail or inconclusive, as specified in an abstract test case, concerning conformance of an IUT with respect to that test case when it is executed.

List of Acronyms

Acronym	*Meaning*
ETSI	European Telecommunications Standards Institute
GUI	Graphical User Interface
IMS	IP Multimedia Subsystem
ISO	International Organization for Standardization
ITU	International Organization for Standardization
IUT	Implementation under Test
NGN	Next Generation Network
PDU	Protocol Data Unit
RTP	Real-Time Transport Protocol
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SQL	Standardised Query Language
SUT	System under Test
TTCN-3	Testing and Test Control Notation Version 3
VoIP	Voice over IP

Conformance Testing of Network Elements in Next Generation Networks Implementing Protocols and Services

Matthias Bormann

First report is part of a portfolio and submitted in partial fulfilment of the requirements of the University of Glamorgan for the degree of Master of Philosophy

Faculty of Advanced Technology
University of Glamorgan
Pontypridd
CF37 1DL

June 29, 2010

Certificate of Research

This is to certify that, except where specific reference is made, the work described in this thesis is the result of the candidate. Neither this thesis, nor any part of it, has been presented, or is currently submitted, in candidature for any degree at any other University.

Signed ....
Candidate

Signed ....
Director of Studies

Date ....

Abstract

Testing network elements on conformance to their respective protocol specification has become a common phase during the development process. For this purpose special programming languages, such as the Testing and Test Control Notation (TTCN-3), are available.

In the context of testing complex communication architectures on conformity, further difficulties were revealed. When testing hybrid network elements with multiple interfaces and different protocols, new testing methods are needed. Systems under test have to be stimulated on all interfaces with emulated network elements, which have to be controlled and synchronised.

This report considers the requirements, design, implementation and application of testing methods for complex communication architectures. For this report an executable set of test cases for IP Multimedia Subsystem's Presence Service, as prominent example for a complex communication architecture was developed.

It is planned to use these developed test cases as an input for ETSI's work on the conformance and interoperability standardisation process for IMS [1].

Contents

CHAPTER 1

Introduction

1.1 The Concept of Communication Networks

A communication network is the infrastructure that allows two or more network elements, such as terminals, processors, computers and control software, to communicate with each other. The network achieves this by providing a set of rules for communication, called protocols, which should be observed by all participating network elements. The use of protocols allows different network elements from different vendors and with different characteristics to operate with each other [2].

1.2 Testing of Communication Software

Testing is an important phase in the life cycle of the development of communication software. It ensures that the end product fulfils the needs of the user. To this end correctness, reliability, robustness, usability, and compatibility are important aspects to be tested. There are specific requirements that have to be met by the software with respect to its field of application.

Exchange of user and control data between network elements in modern communication systems is a prominent field for the application of complex software. Seamless communication between network elements relies on so called protocols, which are rules that ensure the system works properly. Communication protocols are essential for the power and success of any Internet application. Any communication software must comply with the specifications of the implemented protocols. Therefore, conformance testing is of utmost importance to make sure that different implementations of the same protocol will interoperate.

When testing a software on conformance to its protocol specification, it is examined whether all the requirements, so called test purposes, of the protocol are met. A test case, an executable version of the test purpose, is a set of inputs and expected results developed to check if a certain requirement is met by the software. The actual testing of the software, the *Implementation under Test* (IUT), is conducted according to the definitions of the test cases. The result of a test case is a *verdict*, which states that an implementation has passed or failed the test [3]. If the outcome of a test cannot be determined to be pass or fail, the verdict is set to inconclusive.

The process for the development of sets of conformance test cases was standardised by several international standardisation bodies, such as the ITU (*International Telecommunication Union*) [4–10], the ISO (*International Organization for Standardization*) [11–17], and the ETSI (*European Telecommunications Standards Institute*) [18]. For the specification and implementation of conformance test cases, ETSI developed the programming language TTCN-3 (*Testing and Test Control Notation Version 3*).

1.3 Testing of Next Generation Communication Services

A fundamental protocol for communication networks is SIP, the Session Initiation Protocol. It is used for creating, modifying, and terminating communication sessions of voice and video calls over the Internet between two or more parties [19]. SIP was standardised by the *Internet Engineering Task Force* (IETF) [20].

For SIP-based applications the currently used test procedures [21] have proven to be reliable and efficient. Although conformance testing can never guarantee an error-free software, it ensures that the implementation is consistent with its specification with a reasonable degree of confidence [22].

With the increasing demand for new and more complex services in communication networks, additional protocols, based on SIP, are being designed. Conformance test cases are necessary for these newly developed protocols as well. In addition, any software, implementing such a complex service, must be tested to conform to the service's specification. While the process for conformance testing has successfully been applied to developing test cases for communication protocols such as SIP, it has not been used so far for testing complex communication services [1].

Until now the currently applied method to ensure the correct interoperation between network elements providing complex services is interoperability testing [23–25]. The tests are typically conducted by the software developer by comparing it to other products of the company with the same functions that errors can be detected and fixed. However, this method does not ensure the interoperability with products from other companies. With a conformance testing method for complex services it would be more likely that two independently developed and tested products could interoperate successfully.

Therefore, it is important to examine if the existing conformance test methods are still applicable for complex services or if they have to be adapted for the new challenges of modern communication architectures.

1.4 Research Objectives

The aim of the research is to critically analyse existent academic and other technical literature in the field of IP-based communication system testing and to propose a new concept for the development of conformance test cases for complex services. For the evaluation of this new concept it is applied to the IMS' (*IP Multimedia Subsystem*) Presence Service, a prominent example for a service in a complex communication architecture.

Presence is a basic service based on SIP which is able to provide an extensive amount of information to a set of users (figure 1.1). Moreover, it enables third-party services to read and understand presence information, so that the service provided to the user is modified according to the user's needs.

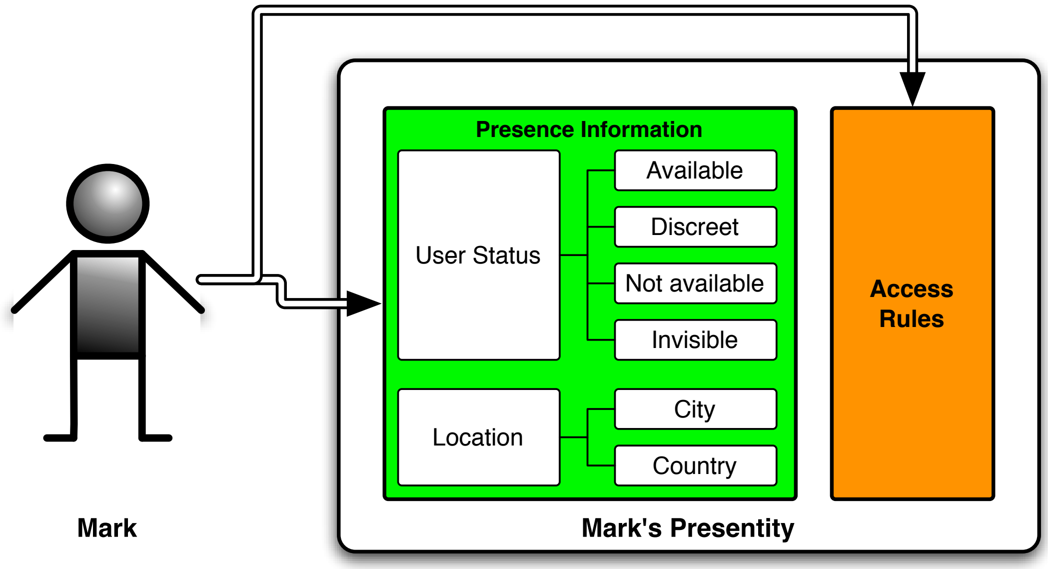

Figure 1.1: Basic Concept of the Presence Service. Mark's ability and willingness to be reached for communication is defined by a set of information known as presence information. Additionally, Mark defines a set of access rules to control access to his presence information. Modified from [26].

One modern communication architecture, supporting the Presence Service, is the *IP Multimedia Subsystem* (IMS). The IMS is the technology merging the Internet with the cellular world, making Internet technologies, such as web, email, instant messaging, and video conferencing, available nearly everywhere.

The research objectives of this report are:

- Critical analysis of existing conformance testing methods in the IMS
- Proposition of a novel strategy for conformance testing of complex services
- Analysis and evaluation of the new concept by applying it on the IMS' Presence Service

This report is structured to reflect these objectives. The next chapter presents the literature review and is followed by a chapter covering the methods on how test cases are being developed for the IMS' Presence Service. The implementation of the developed set of test cases for the Presence Service is shown in the next chapter. The thesis continues with a discussion of the results and ends with a conclusion and the future work.

CHAPTER 2

Literature Review

This chapter gives an overview and a critical analysis of *Next Generation Networks* (NGNs) and their services with special emphasis on the IP Multimedia Subsystem and its Presence Service. Herein, the procedures currently used for the conformance testing of network elements will be scrutinised.

First, the architecture, mode of operation, and the advanced features of the *IP Multimedia Subsystem* (IMS) are described. Next, one of IMS' most important services, the Presence Service, is evaluated. The Presence Service's protocol specifications are examined, which were developed by different standardisation bodies. Finally, conformance testing and its dedicated programming language, the *Testing and Test Control Notation* (TTCN), are inspected.

2.1 The IP Multimedia Subsystem

NGNs are an architectural concept that uses one single network to transport all information and services. Information, such as media and data, is encapsulated into packets, like it is on the Internet. The characteristics of NGNs are specified by the *International Telecommunication Union* (ITU) with the following attributes [27]:

- Packet-based transfer
- Separation of control functions among bearer capabilities, call/session, and application/service
- Decoupling of service provision from network, and provision of open interfaces

- Support for a wide range of services, applications and mechanisms based on service building blocks (including real time/streaming/non-real time services and multimedia)
- Broadband capabilities with end-to-end *Quality of Service* (QoS) and transparency
- Interworking with legacy networks via open interfaces
- Generalised mobility
- Unrestricted access by users to different service providers
- A variety of identification schemes which can be resolved to IP addresses for the purposes of routing in IP networks
- Unified service characteristics for the same service as perceived by the user
- Converged services between fixed/mobile
- Independence of service-related functions from underlying transport technologies
- Compliant with all regulatory requirements, for example concerning emergency communications and security/privacy, etc.

One implementation of a NGN is the IMS [28]. It offers a generic architecture for *Voice over IP* (VoIP) as well as multimedia services based on a *Session Initiation Protocol* (SIP) infrastructure. The idea of the IMS is to offer Internet services everywhere and at every time using cellular technologies. While the former Second Generation networks were circuit-switched, today's Third Generation networks are packet-switched. Using a packet-switched technology, network resources can be utilised more efficiently. A different QoS is needed when using different services (e.g. voice, chat). The most important aspect of the IMS, compared to circuit-switched networks, is that the IMS creates a service environment where all services are independent of one another.

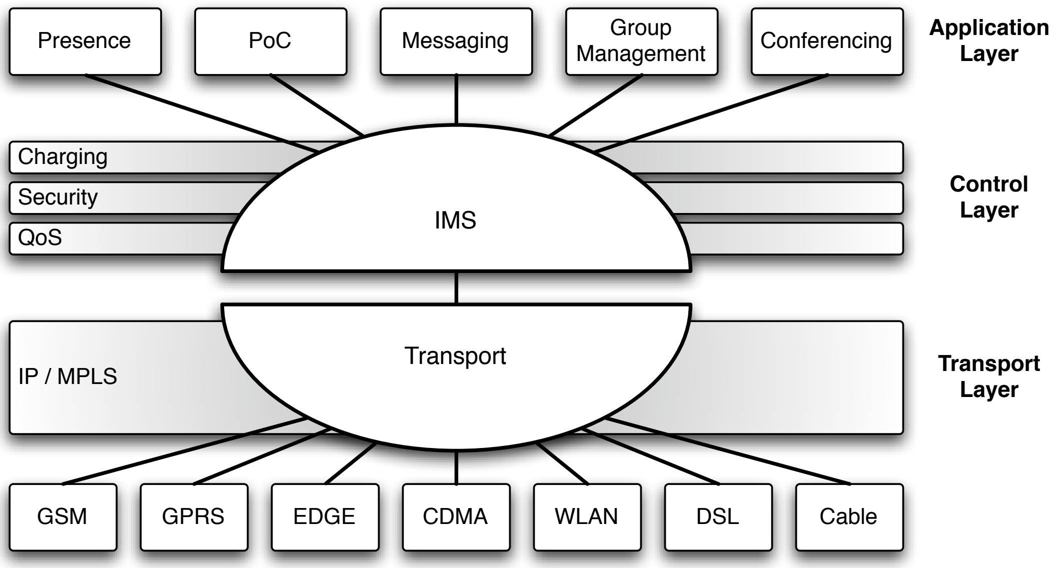

Figure 2.1: IP Multimedia Subsystem's Architecture. IMS is a an access-independent and IP-based service control architecture that provides various types of multimedia services to end-users using IP-based protocols. Modified from [29].

The architecture of the IMS is divided into three layers as shown in figure 2.1. These are [29]:

- The **Application Layer** is where the application server (AS) resides. This is where all of the services are delivered through the IMS interface to the control layer using standardised protocols, primarily SIP.
- The **Control Layer** is the functional area in IMS that provides all of the session and call control. The call session control function (CSCF) is the central routing engine and policy enforcement point for the network and uses SIP for call control. The home subscriber server (HSS) is also found in the control layer. The HSS is a centralised database that contains all the pertinent user information such as home network location, security information, and user profile information (including the services for which the user has subscribed and may therefore participate in).
- The **Transport Layer** comprises many types of access networks. Some examples of packet-based networks are general packet radio service (GPRS), Universal Mobile Telecommunications System (UMTS), code division multiple access (CDMA), wireless local-area networks (WLANs), PacketCable, and asymmetric digital subscriber line (ADSL). The traditional public switched telephone network (PSTN)

is an example of a circuit-switched network.

Users are connected to the IMS infrastructure through the transport layer, either directly through an IMS terminal (such as a 3G wireless handset) or most likely (at least for the near future) through a non-IMS device that interfaces the IMS infrastructure through a gateway. There are several gateway functions found in the transport layer that are primarily in place to provide interworking between legacy networking functions and IMS.

IMS provides an easy and flexible way to deliver services in a standardised manner, which accelerates the creation of services [30]. The following services have already been defined for the IMS [29]:

- Presence Service
- Messaging
- Push to talk over Cellular
- Conferencing
- Group management

Experience shows that there is a demand for new services [31]. Especially the number of applications that provide multimedia services are growing rapidly [32]. Another aspect, and advantage, of the IMS is the modular architecture, which enables the integration of a variety of services from different service providers into the same system. To guarantee the interworking between all those services, conformance testing is indispensable.

2.2 Presence Service

The Presence Service provides a customisable set of information on the status of a user to a group of other users [33]. Furthermore, Presence allows a user to be informed about status changes of other users. Presence information include:

- reachability,
- availability and
- willingness to communicate

On top of this, the Presence Service allows users to give details of devices used and their respective capabilities.

Presence basically defines three functional entities [34], depicted in figure 2.2. These are:

- Presence Source (Presentity)
  - provides user information
- Watcher
  - subscribes to the status of a set of presentities
- Presence Server
  - stores user information
  - controls the access on the information according to defined rules

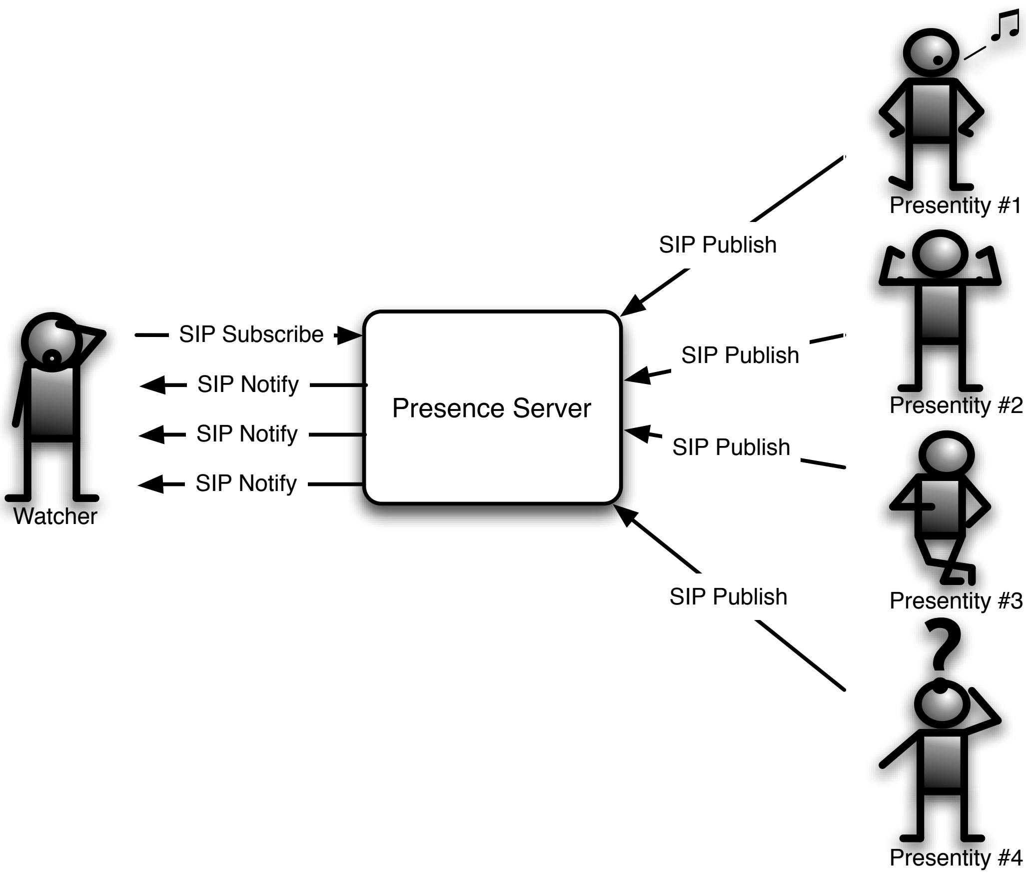

Figure 2.2: Subscription and Publishing - Presence Service's Basic Functions. Users (presentities) are publishing their status on the presence server. The information is stored and can be accessed by other users, so called watchers. Modified from [35].

Based on the presence user information in the presence server, further services, such as advanced call control or multiparty multimedia conferencing, can be implemented.

2.2.1 Presence Service in the IMS

The mapping of the Presence Service' functional entities into the IMS is illustrated in figure 2.3. While the presence server is an application server and connected to the control layer, watcher and presentity both are user agents, typically located in the terminals.

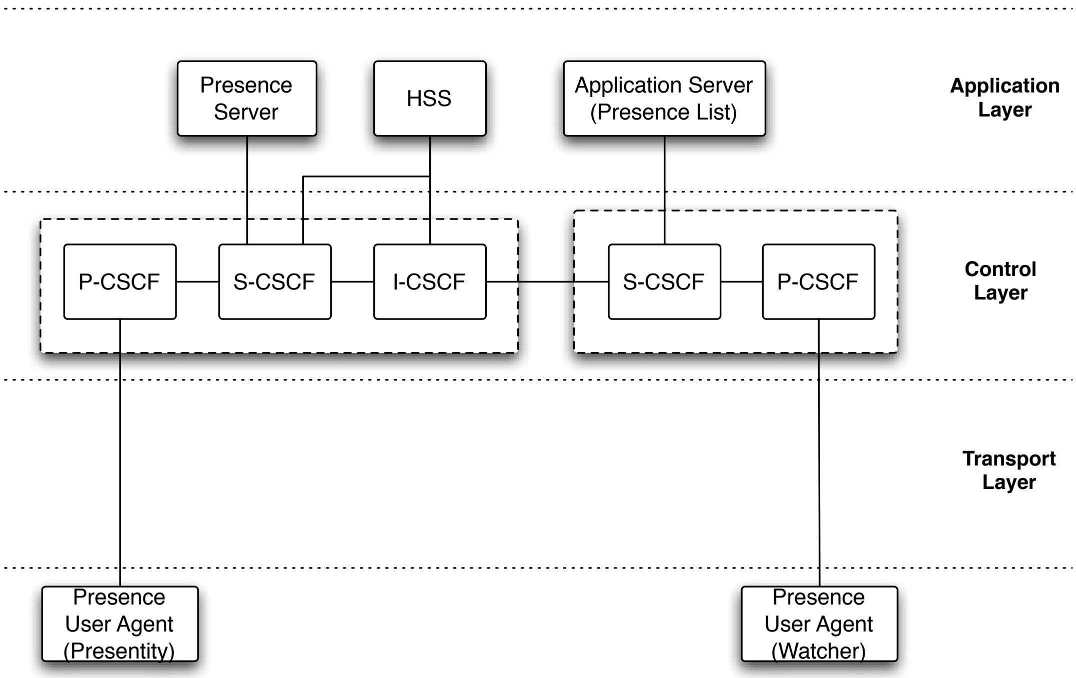

Figure 2.3: Architecture Supporting a Presence Service in the IMS. While the presentity and the watcher are user agents located in the terminals, the Presence Server is an application server. Modified from [36].

The message flow for a subscription of a watcher to the status of a presentity using the IMS is exemplified in figure 2.4. The request is sent from the watcher via his proxy to the presence server of the presentity. The presence server processes the request and checks if the watcher is authorised to watch the presentity's status. In the example above, the access is granted and from now on the watcher is notified on every change of the presentity's status [36].

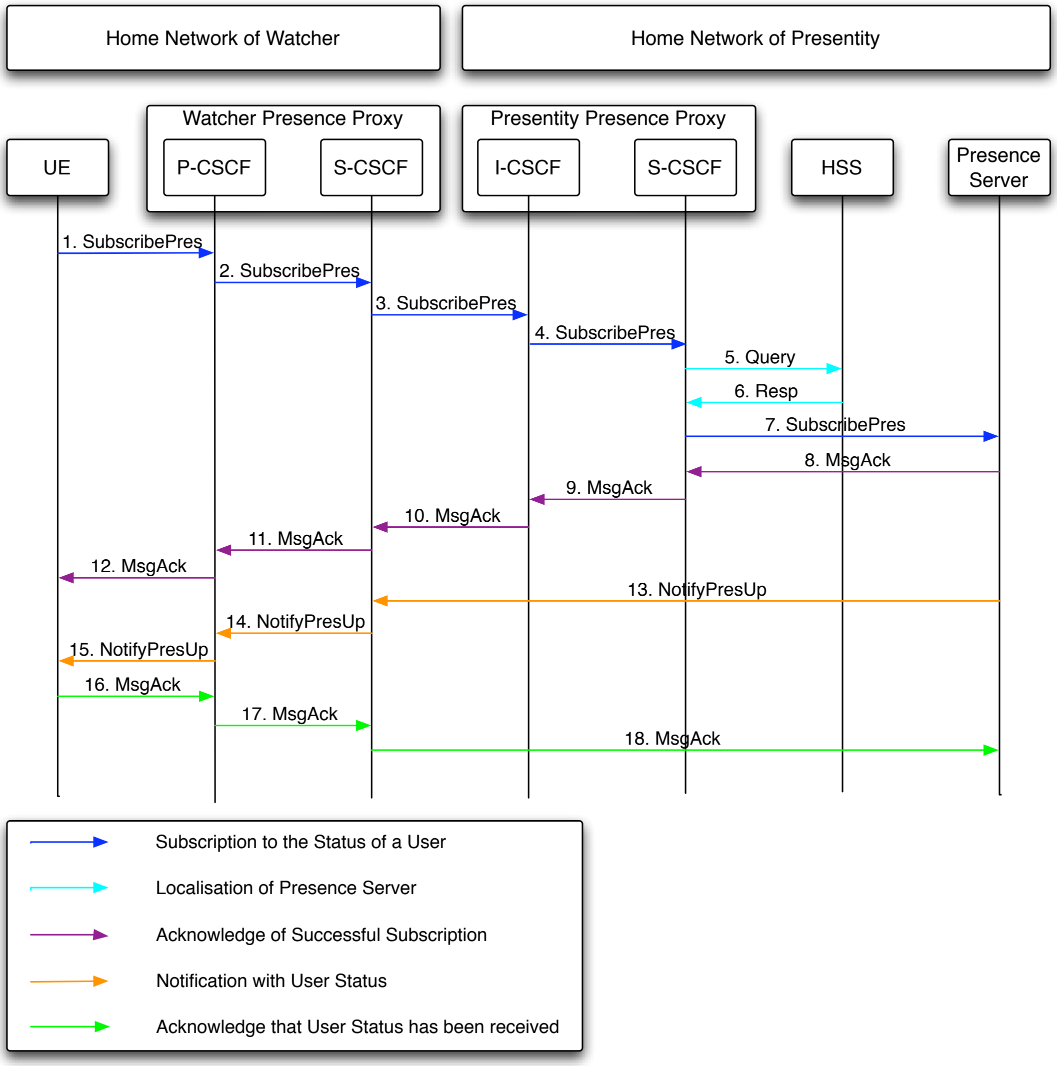

Figure 2.4: Subscription to the Status of an User in the IP Multimedia Subsystem. A client (the watcher) sends a request to another client's (the presentity) presence server. Modified from [36].

2.2.2 Protocol Specifications Used by the Presence Service

Figures 2.5 and 2.6 show typical protocol specifications utilised by the Presence Service and their relations between them. All of them were developed by the *Internet Engineering Task Force* (IETF) (see for example [37, 38]). The specifications can be divided into two groups. One group of the specifications is based on SIP, while the other group of specifications uses HTTP.

Starting from the base specification, e.g. SIP or HTTP, each specification built on top of it, is an extension to the previous one. While for example RFC 3265 extends RFC 3261 by new method types (Subscribe and Notify) and new SIP headers, RFC 3856 adds further functionality to the SIP headers of RFC 3265.

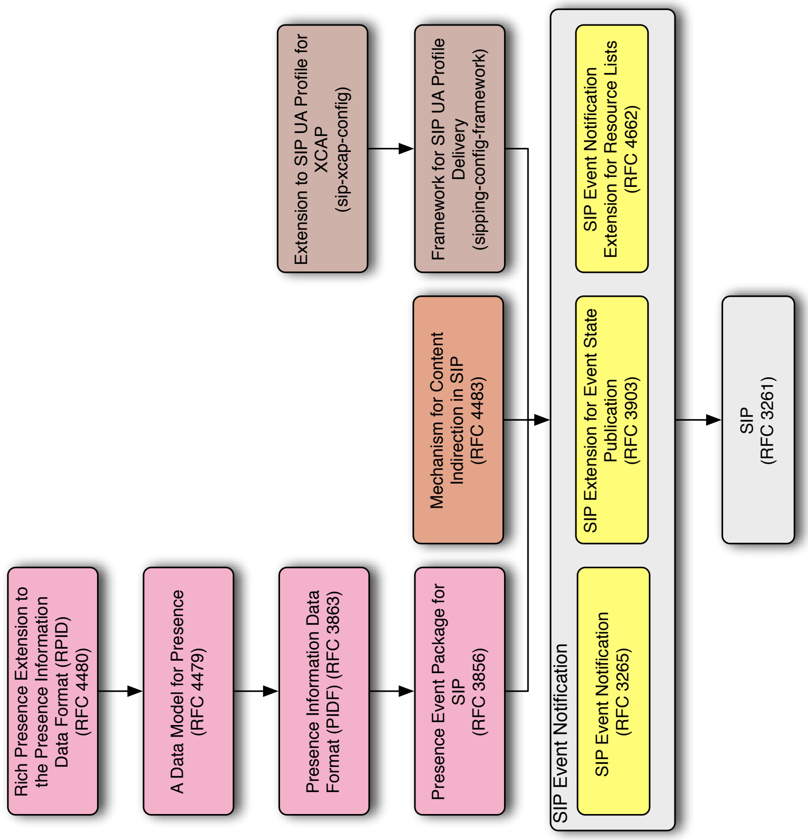

Figure 2.5: SIP-based Protocol Specifications Used by the Presence Service. Protocols are divided into functional groups (colours).

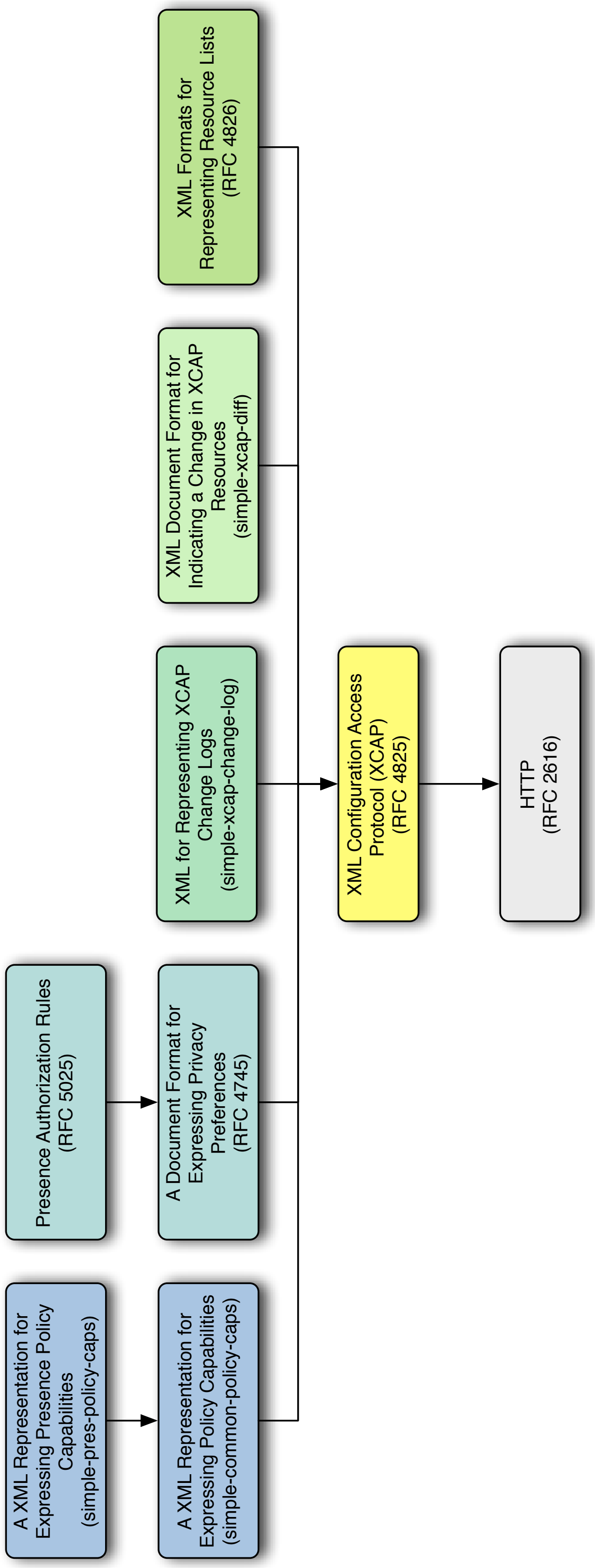

Figure 2.6: HTTP-based Protocol Specifications Used by the Presence Service. Protocols are divided into functional groups (colours).

2.3 Standardisation of the IMS

The IMS, as mentioned in section 2.1, uses Internet protocols. When the IMS needs a protocol for a particular functionality, the IMS standardising bodies specify how to use this protocol. The IMS is standardised by three different organisations [39]. These are:

- Open Mobile Alliance (OMA)
- Third Generation Partnership Project (3GPP)
- Internet Engineering Task Force (IETF)

Each of these organisations has its own specifications for the Presence Service according to its view on the architecture. While the IETF describes the technical protocols for the Presence Service, 3GPP defines how network operators have to parameterise these protocols that they can be used in a 3GPP network. Finally, OMA sets the specifications how the Presence Service is deployed as an application.

A brief review of these three organisations is presented here.

2.3.1 Internet Engineering Task Force

The IETF develops the architecture, protocols and operation of the public Internet. It is a standardisation body and has developed most of the protocols currently used on the Internet. The IETF does not standardise networks or architectures combining different protocols. The focus is on the development and standardisation of IP-related protocols [40].

Structure of the IETF

The IETF's work is distributed among working groups. Each of these working groups has its own task and a limited lifetime. Once the task is completed, e.g. by the delivery of a set of documents, the working group is assigned a new task or it ceases to exists. At the moment there are more than 100 active working groups in the IETF.

Working groups get an acronym name that identifies the assigned task. For instance, *SIPPING* is the acronym of "Session Initiation Protocol Investigation" and *SIMPLE* is the acronym of "SIP for Instant Messaging and Presence Leveraging Extensions" [33].

Types of RFCs

Documents, produced by the IETF, are called *Request for Comments* (RFC). Depending to the contents of the document there are three main types of RFCs [41]:

- Standards-track RFC
- Non-standards-track RFC
- Best Current Practise RFC

Figure 2.7 shows these three types. In the standards-track RFCs, protocols and extensions to protocols are divided into three levels according to their maturity: proposed standard, draft standard, and Internet standard. The evolution of a specification begins with a draft and is supposed to advance to proposed standard, then to draft standard and finally, to Internet standard.

A specification in the proposed standard level has to be stable and unambiguous. Advancing to the draft standard level, at least two independently developed implementations have to interoperate successfully. Nevertheless, in practice the most RFCs only reach the proposed standard level. This is due to the lack of reference implementations which causes challenges when evaluating the developed conformance test cases for this protocol specification.

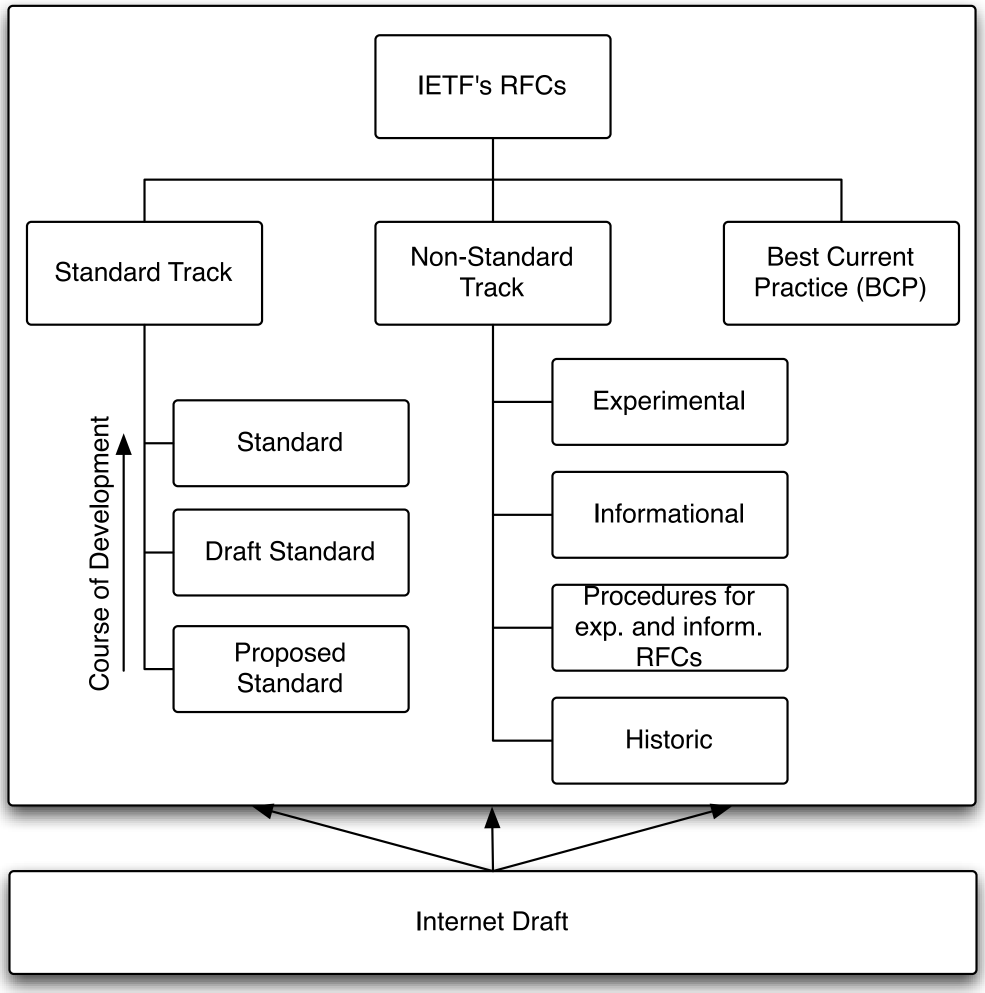

Figure 2.7: RFC types (based on [41])

2.3.2 Third Generation Partnership Project

The Third Generation Partnership Project is a collaboration between a number of telecommunication standard bodies [42]. 3GPP's aim is to develop globally applicable specifications for third-generation mobile systems based on Global System for Mobile communication (GSM).

3GPP prepares, approves and maintains the necessary technical specifications such mobile communication systems including:

- 3GPP core network (originally evolving from GSM) and capabilities such as mobility management, global roaming and utilisation of relevant Internet protocols

- Terminals for accessing the 3GPP core network
- An IMS developed in an access independent manner
- System and service aspects

2.3.3 Open Mobile Alliance

The Open Mobile Alliance was created to provide interoperable mobile data services. OMA's main focus is the usability. Services need to be easy to use [43]. The objectives of the OMA are to:

- Deliver high quality, open technical specifications based upon market requirements that drive modularity, extensibility, and consistency amongst enablers to reduce industry implementation efforts.
- Ensure OMA service enabler specifications provide interoperability across different devices, geographies, service providers, operators, and networks; facilitate interoperability of the resulting product implementations.
- Be the catalyst for the consolidation of standards activity within the mobile data service industry; working in conjunction with other existing standards organisations and industry fora to improve interoperability and decrease operational costs for all involved.
- Provide value and benefits to members in OMA from all parts of the value chain including content and service providers, information technology providers, mobile operators and wireless vendors such that they elect to actively participate in the organisation.

Like the IETF for its RFCs, OMA defines maturity levels for its specifications as well [44, 45]. OMA has three levels:

- Phase 1: Candidate Enabler Release
- Phase 2: Approved Enabler Release
- Phase 3: OMA Interoperability Release

2.4 Software Testing

Software testing in general is an approach for evaluating software in terms of correctness, robustness, efficiency, functionality, and ease of use [46, 47]. The two crucial questions about software correctness are:

- What exactly is the software supposed to do? (functional specification)
- Is it doing exactly what it is supposed to do? (non-functional specification)

The former is performed by the verification step of testing and the latter is performed by the validation step of testing. Correctness is relative to a specification. Specification based testing performs the task of checking the code for correctness. The concept is to develop test cases to check if the responsibilities stated in the specification, the preconditions and the postconditions, are being fulfilled by the designed code.

Verification, validation, and testing are closely tied to software quality. There have been many studies directed toward determining appropriate factors for software quality. Although good quality may be difficult to define and measure, poor quality is readily identifiable. For example, software that is filled with errors or does not work obviously lacks quality. Testing the software, by executing it using representative data samples and comparing the actual results with the expected results, has been the fundamental technique used to determine errors. However, testing is difficult, time consuming, and often inadequate. Consequently, increased emphasis has been placed upon ensuring quality throughout the entire development process, rather than trying to do so after the process is finished [48].

The traditional development life cycle confines testing to a stage immediately prior to operation and maintenance. All too often, testing is the only verification technique used to determine the adequacy of the software. When verification is constrained to a single technique and confined to the latter stages of development, severe consequences can result, since the later in the life cycle that an error is found, the higher is the cost of its correction.

Consequently, if lower cost and higher quality are the goal, verification should not be isolated to a single stage in the development process but should be incorporated into each phase of development. Barry Boehm has stated that one of the most costly mistakes made in software projects is neglecting the detection and correction of software problems until late in the project [49]. The primary reason for early investment in verification activity is to catch potentially expensive errors early before the cost of their correction escalates.

The success of performing verification throughout the development cycle depends upon the existence of a clearly defined and stated product at each development stage e.g., a requirement specification at the requirements stage). The more formal and precise the statement of the development product, the more amenable it is to the analysis required to support verification. Many of the new software development methodologies encourage a visible, analysable product in the early development stages.

The two major categories of testing techniques are referred to as black box and white box testing (see figure 2.8). Black box testing is the verification of an item by applying test data derived from the specified functional requirements without considering the underlying software architecture or composition. Test case selection is based on an analysis of the component's specification without reference to its internal working. Black box testing is also called functional testing. White box testing is the verification of an item by applying test data derived from analysis of the item's underlying software architecture and composition. Test case selection is based on an analysis of the internal structure of the component. White box testing is also known as structural testing. Testing is done using knowledge of the internal working of the system, namely the program logic [50].

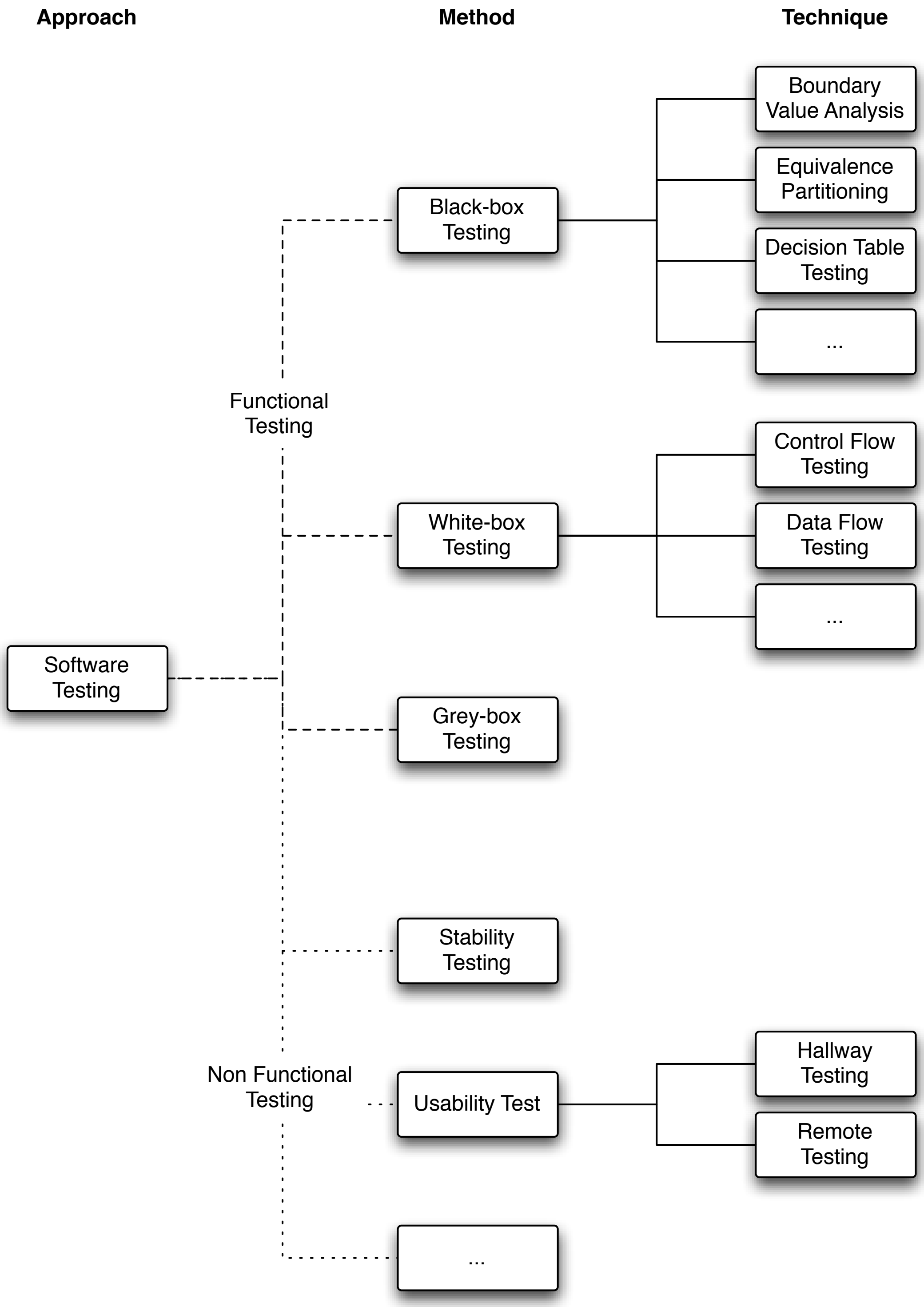

Figure 2.8: Software Testing. Selected methods and techniques for testing software quality aspects.

2.5 Conformance Testing

In the telecommunication system's domain software development is mainly concerned with the correct implementation of protocol specifications. These specifications are – in contrary to software specifications in other domains – standardised and openly available. The major advantage of standardisation enables the possibility that implementations of the same standard but from different developers can successfully interoperate.

However, to ensure the interoperability between the products of different developers the implementations have to be tested as well. In order to be able to compare the quality of implementations from different developers, independently designed methods and techniques are needed. In the telecommunication domain there are several testing methods and techniques available such as conformance and interoperability testing (see figure 2.9).

Conformance testing is used for verifying that an implementation works according to the relevant protocol specifications [51]. The external behaviour of an implementation, the *system under test* (SUT), is tested in order to find logical errors. It is the goal to ensure the interoperability with further systems, which have been tested to be conform as well.

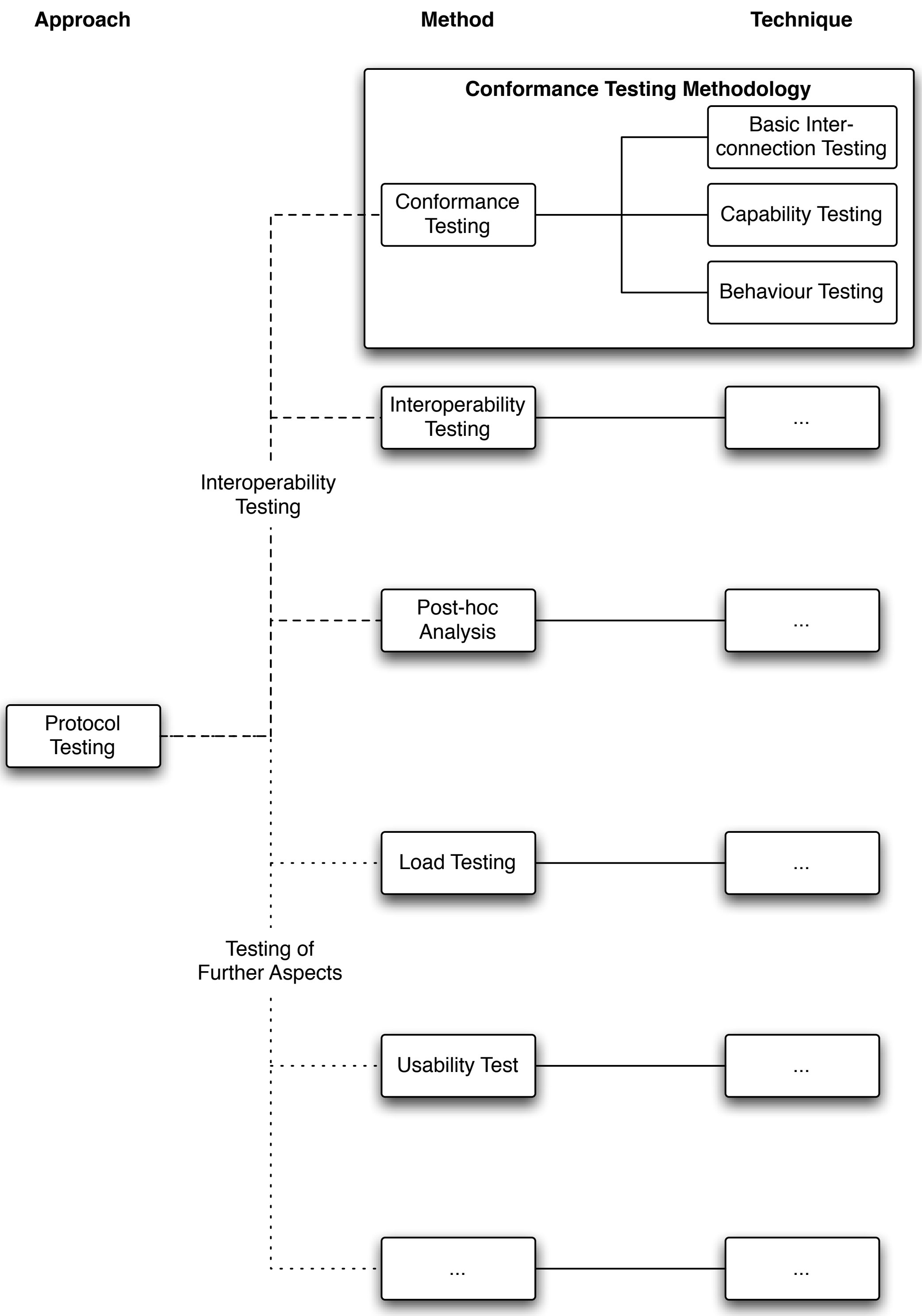

Figure 2.9: Protocol Testing. Selected methods and techniques for testing protocol quality aspects. For conformance testing a standardised methodology exists, which defines a set of methods and the order of their application.

The *International Telecommunication Union* (ITU) and the *International Organization for Standardization* (ISO) have defined a methodology for testing *Open Systems Interconnection* (OSI) protocols. Both have published these methods in [4–10] and [11–17]. ITU's and ISO's aim is to provide general principles for conformance testing which can be applied to a wide range of OSI protocols.

Conformance testing is divided into four types of testing, according to the extent to which they provide an indication of conformance [4]. These four types are [4, 11]:

- Basic interconnection tests: these tests are for detecting any severe cases of non-conformance, i.e., the SUT is not capable of even interconnecting with the tester or it has not implemented correctly the main features of the protocol.
- Capability tests: protocol standards contain a set of static conformance requirements which defines the basic set of capabilities required of an implementation such as the supported options. The capability tests provide limited testing of the static conformance requirements to ascertain what capabilities of the SUT can be observed and to check that those observable capabilities are valid with respect to the static conformance requirements.
- Behaviour tests: these provide as thorough testing of an implementation as is practical. They have the aim of establishing the dynamic conformance of an implementation.
- Conformance resolution tests: these tests provide diagnostic answers, as near to definitive as possible, to the resolution of whether an implementation satisfies particular requirements.

The *European Telecommunications Standards Institute* (ETSI) clarified the principles of ITU's and ISO's testing framework and methodology in [18] and added further criteria for ETSI's Technical Specifications (TSs). In [18] ETSI extends [4–17] in the field of combining test purposes.

2.5.1 Conformance Test Process

Figure 2.10 presents a process on how conformance tests can be derived from the respective protocol specification. These test cases have to be unambiguous and should be reusable. Therefore, a formal testing language like TTCN-3 is utilised.

To test the implementation, an executable test environment is necessary. This task is done by a workbench which compiles the TTCN-3 code into an executable format and manages the messages exchanged. The reaction of the SUT is checked by a pattern matcher on validity. Additionally the workbench does the logging of all messages exchanged and generates a verdict at the end of every test case on the basis of all reactions got from the SUT. All results in a test campaign are summarised in a test report.

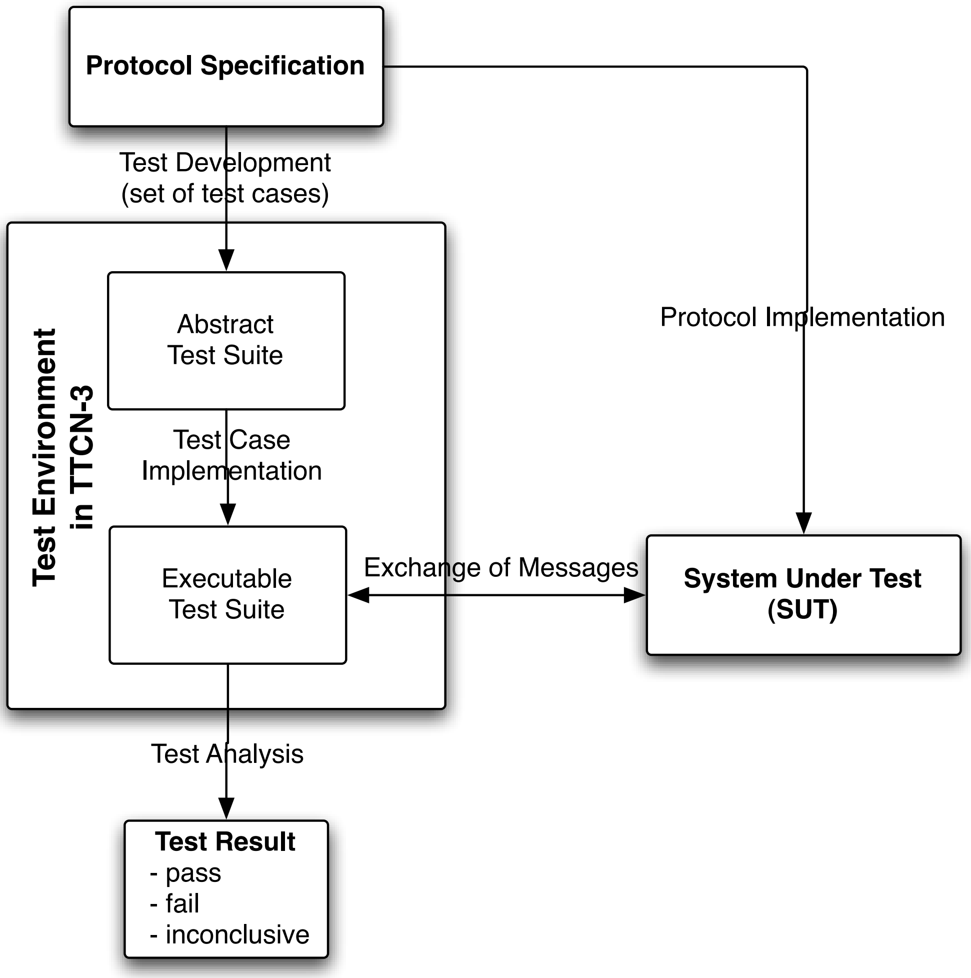

Figure 2.10: Conformance Test Process

2.5.2 Test and Test Control Notation

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. TTCN-3 was developed by the ETSI [52–61].

To test the implementation, an executable test environment is necessary. This task is done by a workbench, which compiles the TTCN-3 code into an executable format and manages the messages exchanged. The reaction of the SUT is checked by a pattern matcher on validity. Additionally, the workbench does the logging of all messages exchanged and generates a verdict at the end of every test case on basis of all reactions got from the SUT. All results in a test campaign are summarised in a test report. An example of a test cases, written in TTCN-3, is shown in figure 2.11.

```
function f_S_SUB_R_200_R_NOT_S_200(
    in bitstring setVerdicts, inout Response loc_response,
    inout SUBSCRIBE_Request v_SUBSCRIBE_Request,
    inout NOTIFY_Request v_NOTIFY_Request,
    in REGISTER_Request v_REGISTER_Request)
runs on ImsComponent {
f_setHeadersForSubscribe(loc_CSeq_s);
SIPP.send(v_SUBSCRIBE_Request) to sent_label;
TAck.start(PX_TACK);
alt {
    [] SIPP.receive(mw_Subscribe_Response_401(v_CallId))
        -> value loc_response {
        f_createSubscribeAuthentication(v_SUBSCRIBE_Request);
        // Resend the saved SUBSCRIBE with Authorization header included.
        if ((setVerdicts and4b '0001'B) == '0001'B) {
           setverdict(pass)
        }
        SIPP.send(v_SUBSCRIBE_Request) to sent_label;
           repeat
    }
    [not received_200] SIPP.receive(mw_Subscribe_Response_200(v_CallId))
        -> value loc_response {
        if ((setVerdicts and4b '0010'B) == '0010'B) {
            setverdict(pass)
        }
        received_200 := true;
        if (not received_not) {
          repeat
        } else {
          TAck.stop
        }
    }
[not received_200] SIPP.receive(mw_Subscribe_Response_202(v_CallId))
        -> value loc_response {
        if ((setVerdicts and4b '0100'B) == '0100'B) {
            setverdict(pass)
        }
        received_200 := true;
        if (not received_not) {
           repeat
        } else {
           TAck.stop
        }
    }
...
```

Set parameters for stimulating message

Send Subscribe message

Start timer for response

Alternative #1

Alternative #2

Alternative #3

Pattern Matcher, judging responses on validity

Figure 2.11: Test Case Example (TTCN-3, simplified)

2.6 Differences between Protocol and Service

A service in the IMS is a function which can be used by the terminals connected to the transport layer. Services utilise protocols to implement a certain function [62, 63]. One example of one such a service is telephony. Telephony, a very basic service, makes use of several protocols, such as SIP for the session setup, the *Session Description Protocol* (SDP) for the session description, and the *Real-Time Transport Protocol* (RTP) for the user data such as speech or video [29].

2.7 Test Coverage

The coverage of a test suite typically reflects the quality of the tested software [64]. Several publications have been concerned with the measurement of the test coverage [65, 66]. Most of these are of a theoretical nature, discussing the benefits of various measurement criteria. However, there has been little discussion yet on the successful use of test coverage measurements on large projects, and whether these have proven to be a cost effective for detecting errors and assuring the quality (e.g. correctness, reliability) of a product [67].

One reason for the lack of data on test coverage measurements during software testing is the difficulty of getting test coverage numbers on large complex products [68]. Despite the academic discussion on the advantages of different test coverage measuring techniques, simple concepts and guidelines for a pragmatic solution for users are not yet available [69].

In the domain of conformance testing ETSI states that the number of tests should be kept reasonable in terms of coverage and costs [18]. This means:

- the more pertinent tests there are, the higher is the coverage
- the more tests there are, the higher are the costs:
  - of production of the test suite
  - of maintenance of the test suite
  - of implementation of the test suite

As a pragmatic solution the test suite should never exceed a few hundreds of test cases regardless of the complexity of the protocol. Furthermore, no metrics exist to assess the coverage of a test specification. The only solution to define coverage metrics is to use formal description techniques. With this it is theoretically possible to determine the number of tests covering the totality of all protocol aspects. With this number a coverage ratio may be computed for a certain test suite. However, in practice this ratio is insignificant, because it does not take into account the fact that all the tests are not equally relevant. This is the reason why a formal and automatic derivation of tests generally leads to a large number of tests without meaning.

2.8 Conclusion

The IMS is a platform for the easy implementation for access independent services such as Presence. The Presence Service, enabling users to share their status information between each other, is a very important part in the IMS' concept and enriching further new services (i.e. location based applications).

The Presence Service in the IMS is standardised by the three organisation bodies IETF, 3GPP and OMA. Each of these organisations has its own view on the IMS' architecture and therefore observes different aspects. In the case of the IETF's Presence standardisation there exist roughly 30 protocol specifications where several of them have not reached the standard stage level.

Conformance testing is a standardised methodology which provides methods on how to develop test cases for protocol specifications. It is used to verify that the implementations work according to their protocol specifications with the goal to ensure the interoperability between them. It has already been applied successfully to protocols such as SIP [21]. However, until now there is no method for testing services on conformity to their specifications.

The challenges for this report are:

- Creation a novel conformance testing methodology for a service
- Higher complexity of services compared to protocols
- Consideration of specifications of three standardisation bodies
- No reference implementations available

CHAPTER 3

Methods for the Development of Conformance Test for Protocols and Services

In this chapter a methodology for the development of conformance test cases for a SIP-based service, such as the IMS' Presence Service, is chosen and then described. The selection of the utilised methodology is discussed and reasoned.

Deriving conformance tests from a single protocol specification, such as SIP, the ETSI's process is utilised [18]. Although it mentions that it is generally possible to derive test cases from service specifications, such as Presence, it leaves the description of possible differences in the process open.

In this report the ETSI's standardised approach for conformance testing for protocol specification is utilised for the test of services - as described in section 2.6 - on conformity. Differences between the existing methods and the methods needed for services are described.

3.1 Selection of a Test Methodology

The challenge when testing any complex software is on the one hand that the test suite used gives a certain degree of confidence concerning the test coverage. On the other hand the execution of the developed test cases has still to be practical and feasible.

There are several academic approaches on how to ascertain a test coverage with good degree of confidence (see for example [70, 71]). However, the development of a test suite for a complex service using these approaches would imply the implementation and execution of an excessive amount of test cases.

Therefore, a compromise has to be found. ETSI's methodology for conformance testing for single protocols does not guarantee a complete test coverage, but still gives - by pursuing several defined steps - a good degree of confidence. The process has been successfully used so far for testing several protocol specifications such as SIP [72]. Another strong point of ETSI's conformance testing process is that it is an openly available specification and standardised.

3.2 Classification of Presence Test Cases

The first step in the conformance test process is the classification of the test cases in groups. In ETSI's conformance process this is done within the *test suite structure* (TSS) [18]. The TSS reflects the coverage of the reference specification by the test suite (TS).

The TSS is divided into five levels:

1. Name of the base specification
2. Role or major functions
3. Pre-defined groups according to the nature of the test
4. Pre-defined groups according to the functional aspected of the test
5. Grouping relevant to the base specification

In figure 3.1 the structure of a TSS is exemplified with the protocol specification for the Event Notification (RFC 3265, [37]).

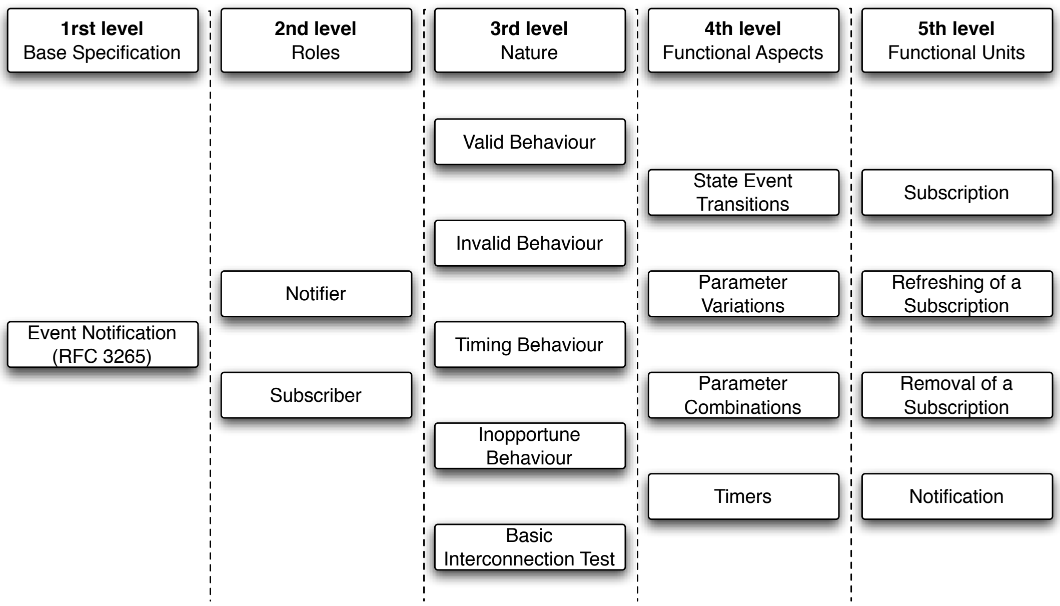

Figure 3.1: Classification of Test Purposes for RFC 3265

For a service such as Presence, comprising of several protocol specifications, there are no rules on how to implement a TSS. Implementing one TSS for every single protocol specification is not sufficient, because interactions between protocols are not considered.

Therefore, additional test cases have to be developed ensuring the correct interworking of two or more different protocol instances in one network element. A specification of a certain function is typically divided across several different protocol specifications. It has to be verified that stimulation of one protocol instance leads to the correct reaction of another protocol instance in the same network element.

A TSS for a service, in this thesis called *service test suite structure* (STSS), has to be designed. The proposed STSS is divided into five levels. These are:

1. Name of the service specification
2. All roles of the included base specifications of the service
3. Pre-defined groups according to the nature of the test
4. Pre-defined groups according to the functional aspected of the test
5. Functions defined in the service specification

For the Presence Service a STSS has to be developed on the basis of the presence specification by the Open Mobile Alliance (OMA) [34].

3.3 Derivation of Test Purposes

In a second step in the conformance test process, test purposes (TPs) are – according to the TSSs and the STSS – derived from each protocol specification. Each TP checks whether the system under test (SUT) fulfils one single conformance requirement of the protocol specification. The TPs are typically written in prose and have to be unambiguous in a way that checking of the IUT can be performed explicitly and exhaustively.

Additionally, *combined test purposes* (CTPs) can be derived from the protocol specification optionally as well. CTPs are a combination of related TPs covering another conformance requirement. CTPs can be used to reduce the number of test cases, while leaving the test coverage acceptable.

For a service, providing several functions which utilise more than one protocol, test purposes have also to be developed according to the STSS. In this report this is done by combining conformance requirements from different protocol specifications into a new combined test purpose. These *service test purposes* (STPs) are comparable to CTPs, but are developed independently to the TPs and CTPs of the protocol specifications.

As described in section 2.7 it is important to limit the number of test cases that their execution is still feasible. Nevertheless the selection of certain test purposes should be reasonable. In this report the test purposes are selected as follows:

1. The number of test purposes is roughly limited to a certain amount (e.g. 300).
2. An estimation is done on how to divide the number of test purposes between the protocol specifications. More test purposes are stipulated for for lower layer protocols (i.e. SIP Presence and XCAP for Presence) and protocols for network elements with multiple interfaces.
3. Test purposes are derived from the protocol specifications covering the mandatory aspects. If possible, invalid test cases for the same aspect are derived as well. If the aspect is concerned with a timing behaviour, the thresholds are tested.

3.4 Test Suite for Profiles

With the presented approach for the derivation of test purposes in section 3.3 services are tested without taking into account of possible influences of the infrastructure they are implemented in such as timing constraints. Therefore, profiles are created, identifying a set of options from a protocol specification, in order to provide a given function in a given environment. With profiles the possible choices for a certain function are limited. Thus, profiles further increase the probability that two network elements can successfully interoperate using the same function on the same infrastructure.

For Presence a typical environment is the IP Multimedia Subsystem. To emulate its network behaviour a profile with the respected attributes (e.g. authorisation headers) is developed for all base specifications. These additional modifications are applied to all test cases.

3.5 Evaluation of TTCN-3 Test Cases

Firstly, TTCN-3 test cases have to be proven to be syntactically correct. This is done by the workbench's compiler. In a second step, every test case has to be checked whether its behaviour during execution is as defined in its respective test purpose. This is typically done by connecting a reference implementation of the protocol to be tested to the workbench [73, 74].

For the Presence Service there are no reference implementations due to the missing of protocol specifications having reached standard stage level. This is because some of the IETF specifications are only in draft stage making the development of reference implementations impossible. Thus, existing Presence Service implementations are the only possibility for the evaluation of the TTCN-3 test cases. If errors are detected during execution their causes have typically to be found in the SUT. But when testing implementations of specifications in draft stage it can also be a problem of misinterpretations of the specification due to ambiguities.

A good quality of the Presence Test Suite is guaranteed by using the ETSI's standardised methodology for the development of conformance test cases. During this process the test coverage is checked by comparing the planned test categories in the TSSs and STSS with the actual derived test purposes from the specifications. If for one category there does not exist a test purpose, it has to be reasoned why.

3.6 Conclusion

For testing a service in general, three categories of test purposes and test cases are needed:

- each single protocol specification
- constraints according to the network architecture
- application-oriented functions of the service

In the context of IMS' Presence Service these are:

- Test cases for every single Presence protocol specifications (IETF based test cases)
- A profile for testing the Presence protocol specification in the IMS environment (3GPP based test cases)
- Test cases for testing the application-oriented functions of the Presence Service (OMA based test cases)

Test purposes for each single protocol specification are not enough for testing the Presence Service. To ensure that a network element supports the Presence Service, the interactions between all respective protocols have to be supported and thus be tested. Implementing the protocols in a certain environment with certain parameters affords further testing.

The proposed approach extends the development process of conformance tests for protocols, that it can now be used for services as well. A test suite for a service, such as the IMS' Presence Service, comprises of three sets of test cases. Each set tests aspects of the service according to the view on the IMS' architecture of section 2.1.

CHAPTER 4

Development of Test Cases for IMS' Presence Service

This chapter describes how conformance tests are developed for protocols and services of a complex communication architecture exemplified with the IMS' Presence Service by applying the methods from chapter 3.

The development process of test cases for the Presence Service is explained and the differences between this and SIP, a lower OSI layer protocol, examined. Test purposes are derived from the protocol specifications and problems named, such as untestable test purposes, multiple protocol instances and hybrid network element testing. Finally, the sets of test cases are evaluated.

4.1 Development of the Presence Test Suite Structures

For a service test suite three categories of test cases have to be developed. Test cases for the protocol specifications, for profile definitions, and the service specification are needed (see figure 4.1).

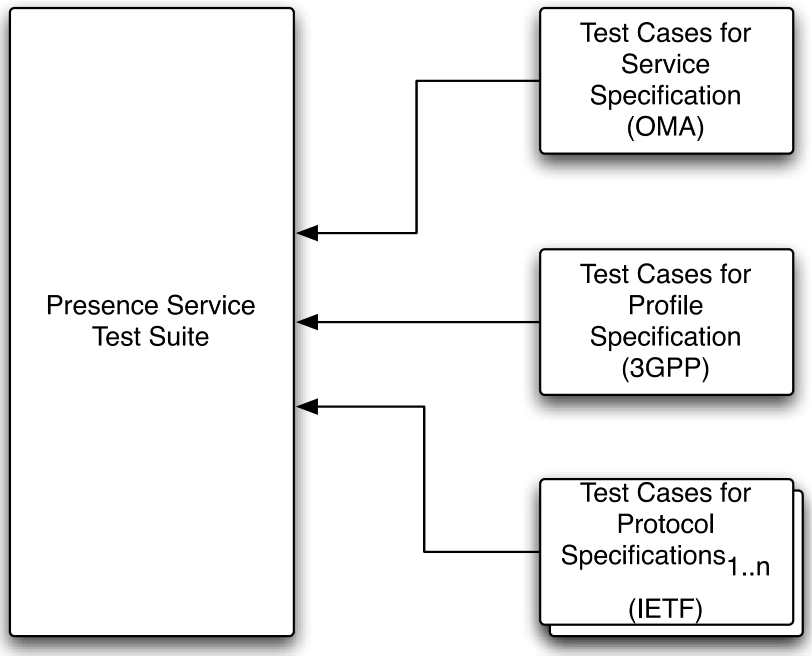

Figure 4.1: Structure of a Presence Service Test Suite. A Presence Service Test Suite consists of three parts. The first part comprises the test cases for the basic protocol specifications. The second part defines test cases for testing network parameters such as timing constraints. Finally, the third part describes the interaction between multiple basic protocol specifications.

4.1.1 Test Suite Structures for Protocol Specifications

For the Presence Service which comprises of several protocol specifications, a TSS for each of these has to be developed separately. For a better overview protocol specifications are grouped according to the protocol specification they are based on, e.g. SIP or HTTP. For each group one TSS, combining all of its protocol specifications, is developed. The TSSs are shown in figures 4.2 and 4.3.

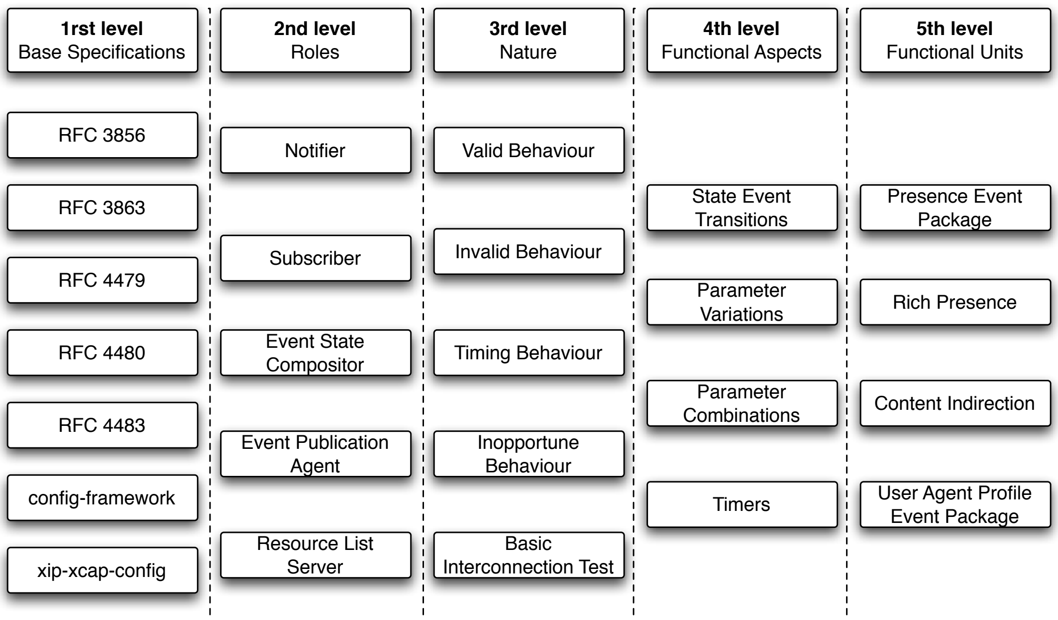

Figure 4.2: Classification of Test Groups for the Presence Service based on SIP

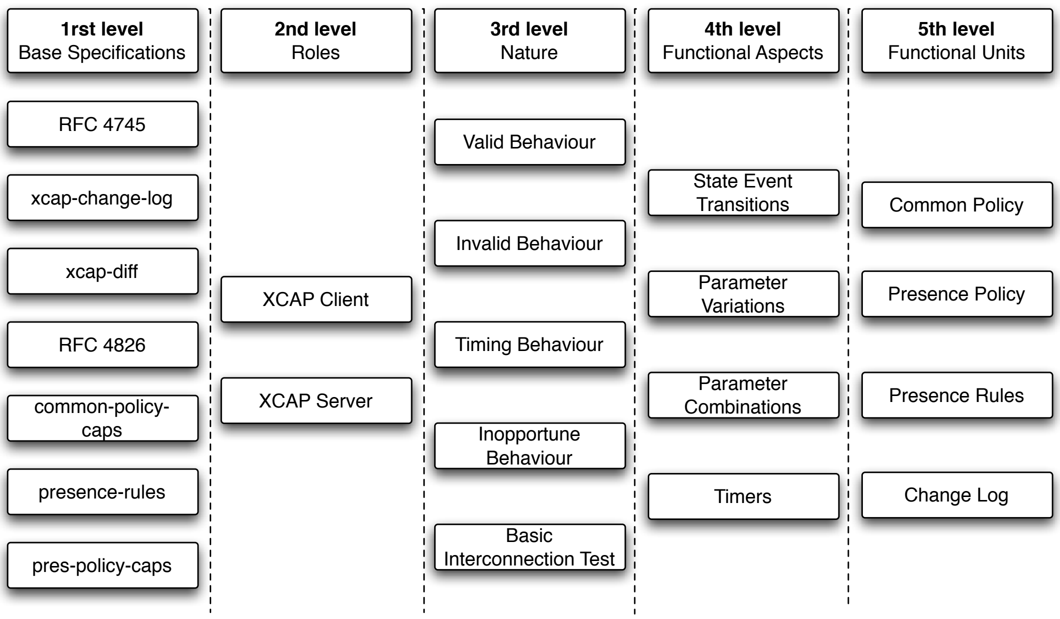

Figure 4.3: Classification of Test Groups for the Presence Service based on HTTP

All protocol specifications have the same roles in the second level of the TSS. Compared to the TSS of a single protocol specification, a combined TSS has more functional units in the fifth level. Elements in the third and fourth level stay the same.

4.1.2 Profile Test Suite

This Profile Test Suite verifies that the Presence Service's protocols are implemented in the way that they can be used in a NGN's architecture. 3GPP defines several parameters – which modify or extend the basic protocol specifications – allowing the successful interoperation between the network elements in the terms of [36]:

- timing constraints for various kinds of terminal equipment
- security aspects
- authorisation headers
- headers for provisioning and charging

4.1.3 Service Test Suite Structure

The Open Mobile Alliance (OMA) defines the specifications for the Presence Service out of an application-oriented view. OMA defines the Presence Service's functions in [34]. This functions are provided by utilising several basic protocol specifications. Implementing test cases verifying these functions are derived from a STSS.

4.2 Test Purpose Nomenclature

Each derived test purpose is named according to a scheme that reflects its position in the TSS or STSS. The parameters are:

- test category (test case or service test case)
- protocol specification
- role of the System under Test (SUT)
- nature of the test
- index

A test case for the event notification (RFC 3265 [37]), which verifies that a network element sends a correct notification after having received a valid subscription could be named *TC_EVNT_NOT_V_001* (see appendix A for a complete list).

4.3 Identification of Test Purposes

Parallel to developing the test suite structures, test purposes have to be identified from the presence protocol and service specifications. The number of test purposes is limited by [18] in the way that for each test suite at most a few hundreds of test cases should exist. This limitation is needed so that the execution of a set of test cases is still feasible. The development of the Presence Service, conducted as a project for a commercial partner, was limited to 300 test cases beforehand.

Firstly, as a first estimation the number of test purposes is divided between the two categories (i.e. SIP-based and XCAP-based specifications) with a ratio of 2:1. There are less XCAP-based specifications and most of them are in draft status, which means these might still be incomplete and ambiguous.

In a second step for each SIP method (Subscribe, Notify, Publish) and for the basic XCAP method 30 test purposes are planned. The rest of the number of test purposes are divided equally between the specifications except for the more complex standards (e.g. multiple interfaces, many combination of values of certain attributes).

As an example for the derivation of test purposes from a protocol specification RFC 3903 [75] and its TSS (see figure 4.4) are chosen and described. Additionally, for all of the following groups one representative test purpose is presented in section 4.3.1 and section 4.3.2:

- SIP-based Presence Test Purposes
  - valid
  - invalid
  - timing
- HTTP-based XCAP Test Purposes
  - valid
  - invalid
  - timing

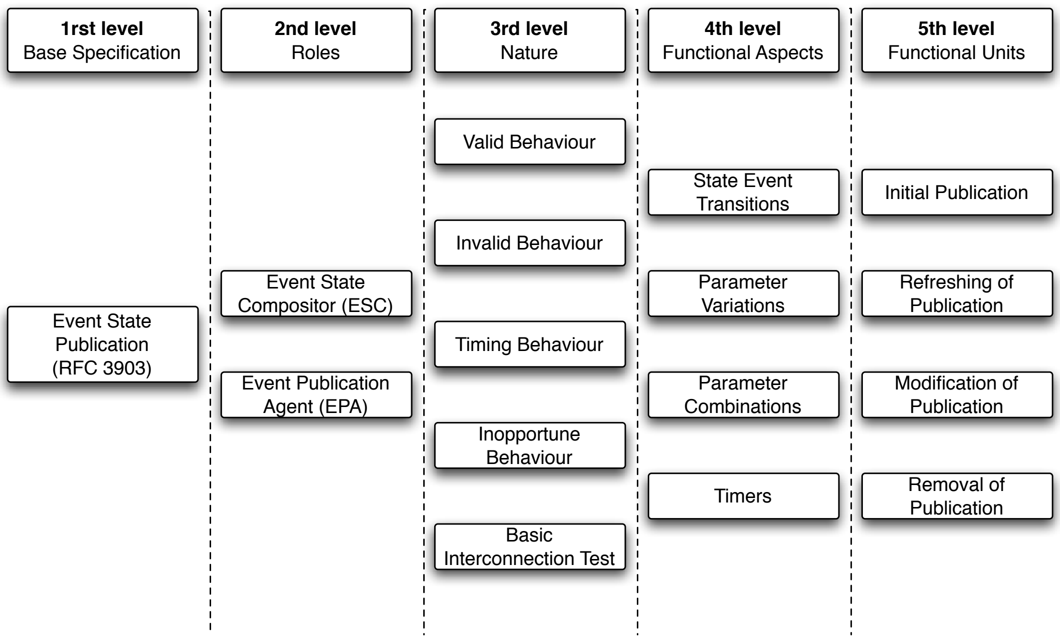

Figure 4.4: Classification of Test Purposes for Event State Publication (RFC 3903)

The roles and functions are described in the TSS. Next, mandatory elements are searched in the specifications and mapped to one of the combinations of the TSS. A few examples on how test purposes are derived are shown:

1. RFC 3903, section 4.1

   - Specification: *For determining the type of the published event state, the EPA MUST include a single Event header field in PUBLISH requests.*

   - Valid test purpose: Ensure that the SUT sends a PUBLISH request containing an Event header field. (Role tested: EPA, Function: Initial Publication, Nature: Valid)

   - Invalid test purpose: Ensure that the SUT after having received a PUBLISH request containing no Event header field, answers with a 400 Bad Request. (Role tested: ESC, Function: Initial Publication, Nature: Invalid)

2. RFC 3903, section

   - Specification: *A publication refresh MAY contain a single Expires header field. This value indicates the suggested lifetime of the event state.*
     The ESC may lower the suggested lifetime of the publication refresh, but it

will never extend it. If an Expires header field is not present, the EPA is indicating its desire for the ESC to choose. The Expires header field in a 2xx response to the publication refresh indicates the actual duration for which the publication will remain active.

- Valid test purpose: Ensure that the SUT, after having received a PUBLISH with an Expires header, sends a 200 OK or 202 ACCEPTED with an expires value less than or equal to the publish expires value and more than zero. (Role tested: ESC, Function: Refresh Publication, Nature: Valid)
- Valid test purpose: Ensure that the SUT, after having received a PUBLISH with no Expires header, sends a 200 OK or 202 ACCEPTED with an expires value greater than zero. (Role tested: ESC, Function: Refresh Publication, Nature: Valid)
- Untestable, timing test purpose: Ensure that the SUT after having received a 2XX with a greater value in the Expires header than in its Expires header of the PUBLISH request sent, ignores this new value. (Role tested: EPA, Function: Refresh Publication, Nature: Timing)

4.3.1 Presence Test Purposes

Table 4.1: Presence Test Case - Valid Behaviour

Test Purpose ID	TC_EVNT_NOT_V_013
Status	Mandatory
Reference	RFC 3265, section 3.1.1
Description	Ensure that the SUT, after having received a SUBSCRIBE request, sends a 200 OK or 202 ACCEPTED with an expires value less than or equal to the SUBSCRIBE's expires value and more than zero.
Test Steps	1. Send SUBSCRIBE request to the SUT. 2. Wait for 200 OK or 202 ACCEPTED response from the SUT. Expires value in the received response has to be not greater than the expires value in the SUBSCRIBE request and greater than zero.

Table 4.2: Presence Test Case - Invalid Behaviour

Test Purpose ID	TC_EVNT_NOT_I_001

Continues on next page

Status	Mandatory
Reference	RFC 3265, section 3.1.6.1
Description	Ensure that the SUT, having received a SUBSCRIBE with an invalid event-type in the event-header, sends a 489 Bad Event response.
Test Steps	1. Send SUBSCRIBE request with an invalid event-type ("xyz") to the SUT. 2. Wait for 489 Bad Event response from the SUT.

Table 4.3: Presence Test Case - Timing Behaviour

Test Purpose ID	TC_EVNT_NOT_TI_001
Status	Mandatory
Reference	RFC 3265, section 3.3.4
Description	Ensure that the SUT, after having accepted a subscription, terminates the subscription according to the expires value in the 200 OK.
Test Steps	1. Send SUBSCRIBE request to the SUT. 2. Wait for 200 OK or 202 ACCEPTED response from the SUT. 3. Start a timer according to the value found in the 200 OK / 202 ACCEPTED response from the SUT. 4. Wait for NOTIFY request from the SUT. 5. Send 200 OK response to the SUT. 6. Wait for a NOTIFY request from the SUT with a subscription-status of terminated before the timer runs out. 7. Send a 200 OK response to the SUT.

4.3.2 XCAP Test Purposes

Table 4.4: XCAP Test Case - Valid Behaviour

Test Purpose ID	TC_XCAP_SERVER_V_001
Status	Mandatory
Reference	RFC 4825, section 7.1

Continues on next page

Description	Ensure that the SUT creates the document and sends an 200OK response after having received a HTTP PUT request with a valid XCAP root and a valid document selector within the URI. The MIME content type is the type defined by the application usage.
Test Steps	1. Send a HTTP PUT request with a valid XCAP root and a valid document selector within the URI to the SUT. 2. Wait for 200 OK from the SUT.

Table 4.5: XCAP Test Case - Invalid Behaviour

Test Purpose ID	TC_XCAP_SERVER_I_001
Status	Mandatory
Reference	RFC 4825, section 7.0
Description	Ensure that the IUT sends a 404 Not Found response after a second DELETE request on an deleted document.
Test Steps	1. Send a HTTP PUT request with a valid XCAP root and a valid document selector within the URI to the SUT. 2. Wait for 200 OK from the SUT. 3. Send a HTTP GET request to the SUT with an URI pointing to the created XCAP document. 4. Wait for a 200 OK from the SUT. 5. Send a HTTP PUT request to the SUT for deleting the XCAP document. 6. Wait for a 200 OK from the SUT 7. Send a HTTP GET request to the SUT with an URI pointing to the deleted XCAP document. 8. Wait for a 404 Not Found response from the SUT.

4.4 Multiple Protocol Instances

To test typical functions of the Presence Service, like shown in figure 2.4 (page 12), the SUT has to be stimulated on all its interfaces according to the protocol specification. E.g., testing the correct processing of a subscription by an implementation of the presence server; a watcher and a presentity have to be emulated by the test system. In figure 4.5 the watcher and the presentity are the actuators, which stimulate the presence service on two interfaces.

For each emulated actuator a so called parallel test component (PTC) is generated. These PTCs have to be controlled and synchronised. This is done by a main test component (MTC). The test purpose of such a test is listed in table 4.6 and the execution of its test case, using two PTCs, is shown in figure 4.6. Each PTC stimulates the SUT in an independent process. The reactions of the SUT are checked by the PTCs. On the summarised basis of this, the verdict is determined by the MTC.

It is important that all test cases in a test set can be executed automatically one after another. Thus, each test case is designed in a way that the SUT starts in the same initial position. With a preamble the SUT is transferred into a position where the test purpose can be executed. After this, the SUT is transferred back to its initial position with in a post-amble. If not applying pre- and post-amble, each test case has to be started separately.

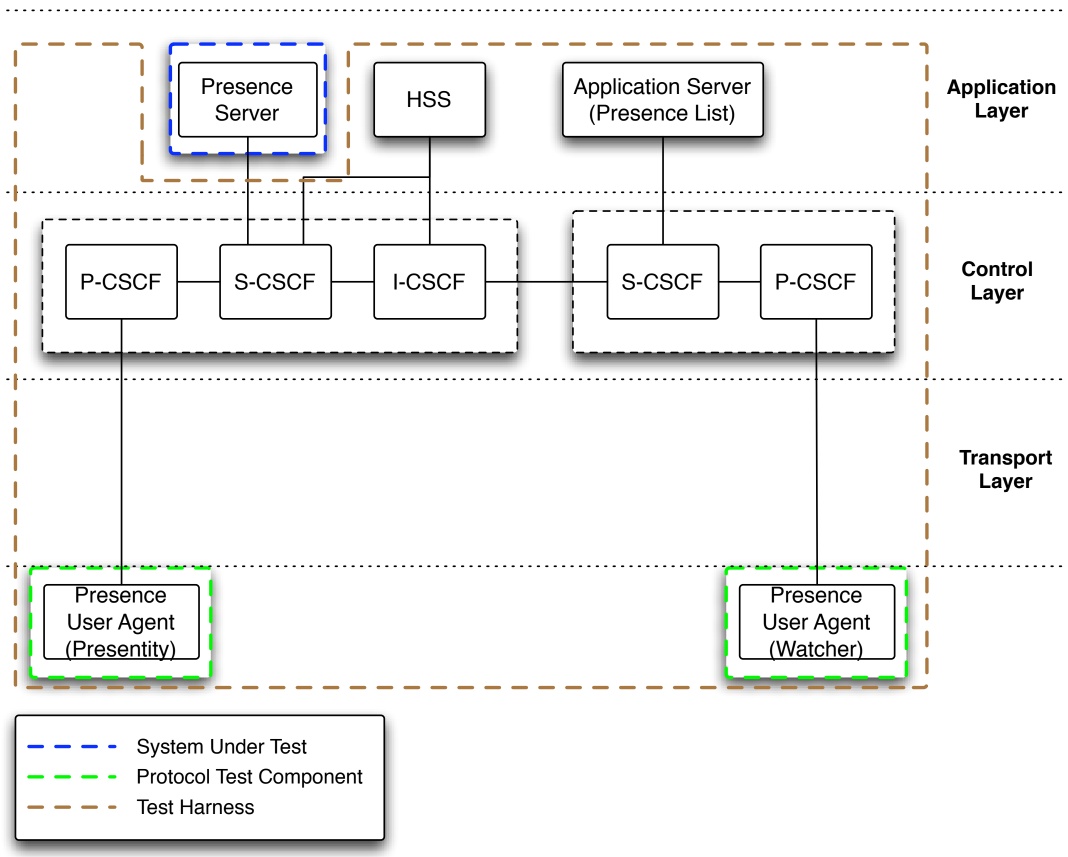

Figure 4.5: Testing with Multiple Protocol Instances

Table 4.6: Multiple Protocol Test Case

Test Purpose ID	TC_SEL_RLS_V_005
Status	Mandatory
Reference	RFC 4662, section 4.5
Description	Ensure that the IUT, after having received a SUBSCRIBE, sends NOTIFIES containing a state for each instance of a resource.
Test Steps	1. PTC1: Send SUBSCRIBE request with an event-list value in the support header to the SUT. 2. PTC2, PTC3: Wait for SUBSCRIBE request from the SUT and send 200 OK and NOTIFY. 3. PTC1: Wait for 200 OK from the SUT. 4. PTC1: Wait for NOTIFY from the SUT with one state from PTC2 and PTC3.

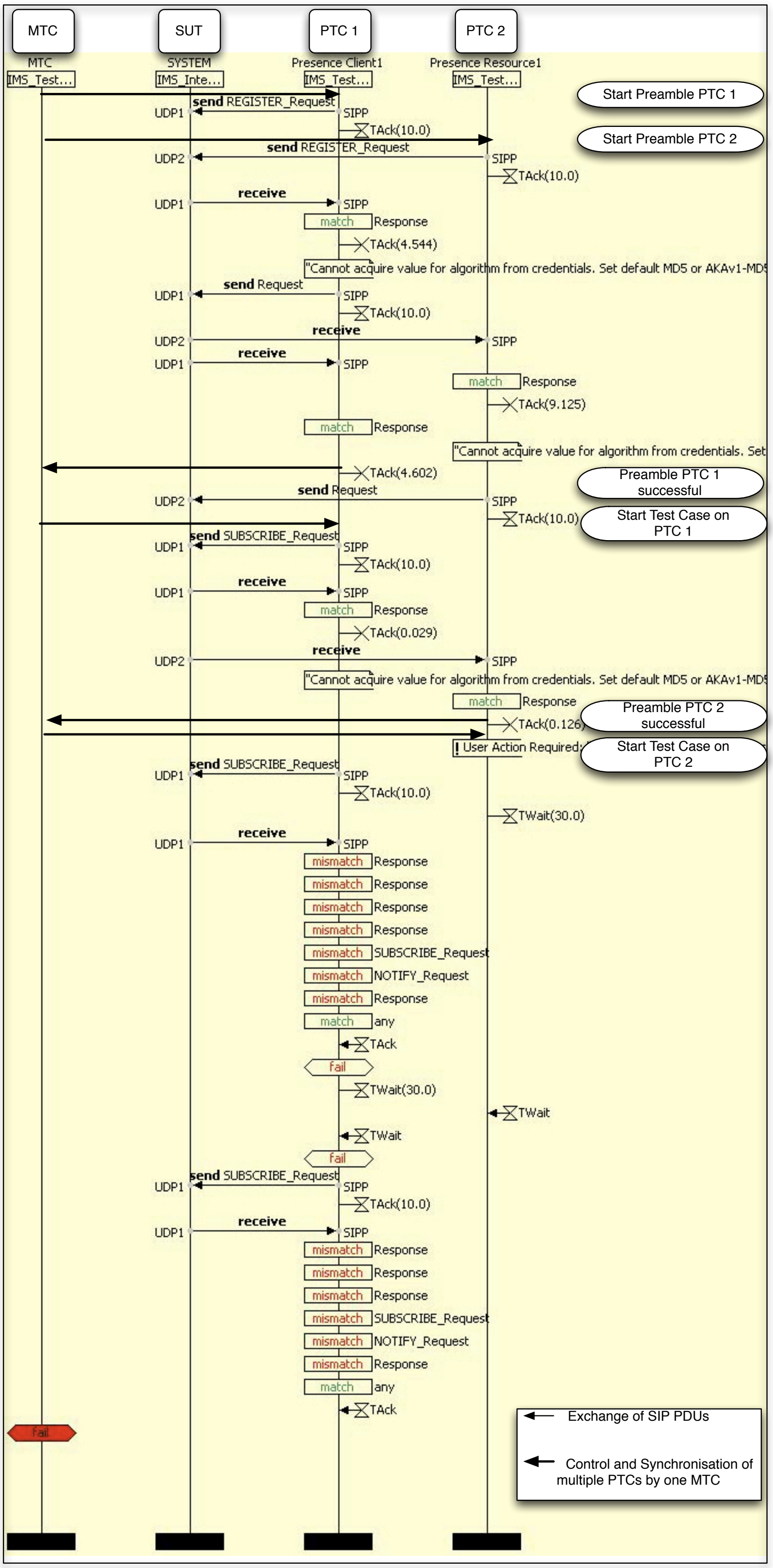

Figure 4.6: Test Execution with Multiple Protocol Instances

4.5 Hybrid Network Elements

On top of testing network elements with multiple interfaces using the same protocol, there are also elements which use different protocols at different interfaces. These are called hybrid network elements. They are characterised by several interacting state machines, each implementing one protocol specification. A test purpose for a hybrid network element is listed in table 4.7.

Table 4.7: Hybrid Protocol Test Case

Test Purpose ID	STC_PX_PS_V_003
Status	Mandatory
Reference	RFC 4662, section 4.5
Description	Ensure that the IUT, after having received a SUBSCRIBE, sends NOTIFIES containing a state for each instance of a resource.
Test Steps	1. PTC1: Send SUBSCRIBE request to the SUT. 2. PTC2, PTC3: Wait for SUBSCRIBE request from the SUT and send 200 OK and NOTIFY. 3. PTC1: Wait for 200 OK from the SUT. 4. PTC1: Wait for NOTIFY from the SUT with one state from PTC2 and PTC3.

Figure 4.7 shows an example for a hybrid message flow. In this case the SUT is a presence server with three interfaces, two of them using SIP and one using XCAP (XML Configuration Access Protocol) [38]. For simplification the MTC as well as pre- and post-amble are omitted. In this scenario the watcher (PTC 1) subscribes to the status of the presentity (PTC 2). The presence server has to request the XCAP server, if it is allowed to forward the information to the watcher. It enquires the XCAP server (PTC 3), which has no entry for this user. Therefore, the presentity is asked for permission. Here, the presentity accepts this and the presence server has to notify the watcher that it is allowed to see the presentity's status and add an entry at the XCAP server for future requests of this watcher.

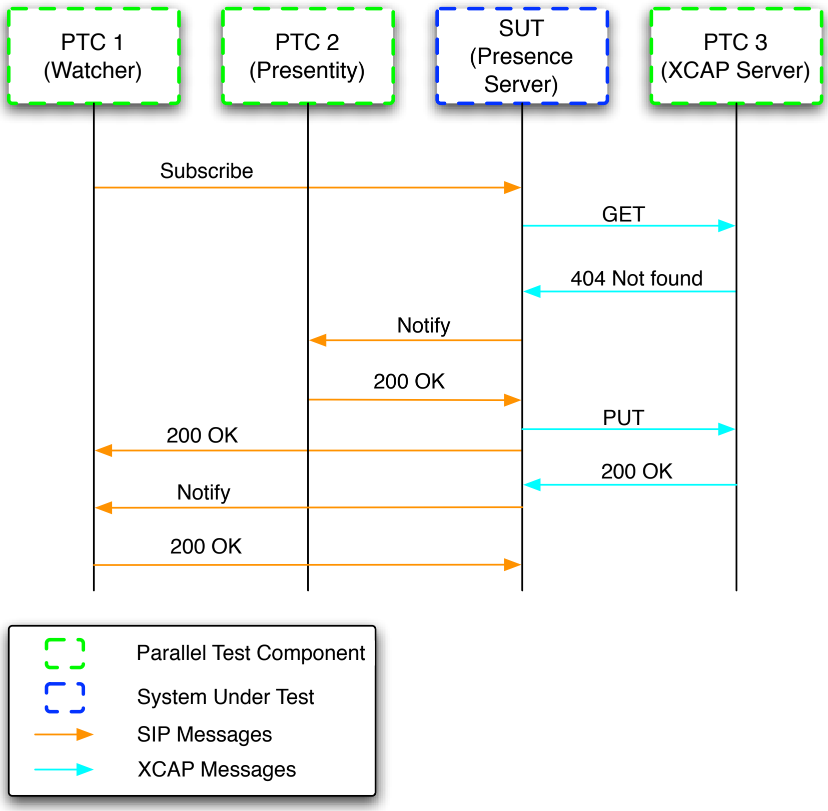

Figure 4.7: Testing on Multiple Interfaces with Different Protocols

For each interface implemented by a network element, one PTC with the respective protocol is needed. These, once again, have to be synchronised. If the SUT responds in an unexpected way, it has to be correctly dealt with by every PTC. Even after an error of the SUT, it has to be led back in its initial position.

Testing hybrid devices, comprising different interacting protocol state machines, means that the stimulation of the SUT with one message on one protocol interface is not necessarily inducing the sending of exactly one message of the SUT on the other protocol interface. In addition to this, the increasing number of messages in a test sequence with a larger amount of parameters (due to multiple protocols) results in a huge amount of test case combinations.

4.6 Implementation of Test Cases

The test purposes, identified in sections 4.3-4.5, have to be made executable. This is done by utilising the *Testing and Test Control Notation* (TTCN-3).

The test cases are grouped in two main categories SIP and XCAP having both one group of test cases for each related protocol specification. Additionally, there is one group for the service test cases (i.e. hybrid test cases).

For a high re-use of the code basic parameters such as message headers can be stored in so called templates. The test environment (i.e. workbench) is connected via a test adapter to the SUT. This is done by defining interfaces with TTCN-3. The test adapter, which is basically a UDP socket, can easily be integrated. The codec, responsible for the conversion of a TTCN-3 template into a SIP message, is provided by the commercial partner. After the SIP message has been generated by the coded it is sent via system adapter to the SUT.

The response of the SUT is received by the test adapter and handed to the codec. The codec tries to decode the SIP message according to a template, which describes the expected response. If the received message and the template match, the verdict can be set.

4.7 Correctness of Test Cases

Syntax and semantic of an implemented TTCN-3 test case have to be correct for a successful execution. While the syntax can be checked within the programming environment, e.g. TTCN-3 compiler, it's hard to evaluate the semantic of a test case.

One promising approach for checking the semantic of implemented test cases, is the use of a reference implementation as system under test. This is a convenient way but – due to the lack of reference implementations – is often not possible.

An alternative to this approach is to take any implementation as system under test available and test it against the test suite. For each test case, which returns a fail as verdict, the reason is identified, whether it is an error in the SUT or in the test suite. Using this approach, the test suite's quality is increased incrementally.

4.8 Conclusion

In this chapter the newly extended conformance testing methodology was applied on the specifications of the Presence Service. The resulting test suite consists of three sets of test cases. These are:

- Test cases for the protocol specifications of the IETF
- Test cases for the specifications of 3GPP to utilise the IETF protocols in a 3GPP network such as the IMS
- Test cases for functions of the Presence Service comprising of several interacting protocol specifications

For the implementation of these sets of test cases test purposes had to be derived from the specifications. The selection of the test purposes was conducted by structuring each protocol specification in categories by using a TSS. Using examples, the derivation of test purposes for one protocol specification was illustrated.

Next, examples were shown for testing network elements on multiple interfaces implementing one or more protocols. These test cases could be implemented with TTCN-3 as well by emulating each network element using a PTC and controlling all PTCs with a MTC.

Finally, the selected test cases were made executable by using TTCN-3. In terms of syntactically correctness the test cases were verified by using the workbench's compiler. The semantic of the test cases is more difficult to examine because of the lack of reference implementation. Therefore, the recently, but incomplete, developed SUTs were used and the log of each failed test case execution inspected manually. This was done to verify that the cause of the error was not due to an ambiguous specification.

CHAPTER 5

Discussion

This chapter checks how and to which extent the research objectives, which were defined in chapter 1, have been fulfilled. Firstly, the modifications made to the conformance test development process are reviewed. In a next step the evaluation process for the developed Presence Service test cases is examined. Finally, the test coverage is evaluated.

5.1 Conformance Testing Methodology for Services

For the development of conformance test suites for services ETSI's conformance testing methodology for protocols was applied and extended. In a first step the necessary specifications were selected and examined with the goal to define the structure of the test suites. Services in the IMS are typically described by several organisations, which have different views on the architecture. Consequently, service test suites are structured by breaking down the test cases in three categories for:

- protocol specifications
- network specifications
- service specifications

This extension to the previous methods improves the overview on the specifications of a service in a complex communication system and by systematically dividing the test suite structure.

In a second step for each group of specifications test purposes have to be identified. This process is supported by designing Test Suite Structures (TSSs) for the base protocol specifications, defining the network parameters as well as implementing a special TSS for the service specification; the service test suite structure (STSS).

With these TSSs and the STSS the specifications are further broken down in groups (e.g. roles, nature of the test, functions). This method helps ordering the derived test purposes into groups. This is supported by using a nomenclature (see section 4.2, page 40). Moreover, in the later stages of the development process these TSSs and the STSS can be used to evaluate if all groups are covered with test purposes.

5.2 Evaluation of Test Purpose Derivation

In table 5.1 the specifications, which were considered for the Presence Test Suite, are listed. For each specification the following aspects are included:

- Standardisation status of the specification
- Number of pages of the specification
- Base protocol (SIP/XCAP)
- Estimated number of test purposes (ETPs)
- Actual number of derived test purposes from the specification (TPs)
- Implemented number of test cases with TTCN-3 (TCs)

The commercial partner of the project agreed upon the development of roughly 300 test purposes. The number of these test purposes were in a first step assigned to the protocol specifications. This was done by applying the method presented in section 3.3, page 34. Lower layer protocols such as number 1-3 in table 5.1 were considered with a higher number of test purposes. With this approach conformance errors can be traced back to the correct protocol specification more easily. For example, if a certain conformance requirement of a lower layer protocol is not tested and if a test case for higher layer protocol which is based upon this requirement fails, it is hard to detect the origin of the error.

Table 5.1: Comparison of Estimated and Derived Test Purposes

No.	***Description***	***Status***	***Pages***	***Base Protocol***	***ETPs***	***TPs***	***TCs***
1	RFC 3265 - sip-events	proposed standard	38	SIP	60	59	57
2	RFC 3903 - sip-publish	proposed standard	32	SIP	30	33	30
3	RFC 4825 - simple-xcap	proposed standard	71	HTTP	30	33	30
4	RFC 4662 - simple-event-list	proposed standard	39	SIP-Presence	20	19	16
5	RFC 3856 - simple-presence	proposed standard	27	SIP-Presence	10	11	10
6	RFC 3863 - impp-cpim-pidf	proposed standard	28	SIP-Presence	10	16	14
7	RFC 4479 - simple-presence-data-model	proposed standard	35	SIP-Presence	15	20	16
8	RFC 4480 - simple-rpid	proposed standard	37	SIP-Presence	15	18	16
9	RFC 4483 - sip-content-indirect-mech	proposed standard	17	SIP-Presence	10	16	8
10	sipping-config-framework	draft 15	55	SIP-Presence	10	17	17
11	sip-xcap-config	draft 00	14	SIP-Presence	10	10	9
12	RFC 4826 - simple-xcap-list-usage	proposed standard	32	XCAP	10	11	10
13	RFC 4745 - geopriv-common-policy	proposed standard	32	XCAP	10	9	9
14	RFC 5025 - simple-presence-rules	proposed standard	28	XCAP	10	9	9
15	simple-common-policy-caps	draft 02	13	XCAP	10	11	7
16	simple-pres-policy-caps	draft 01	10	XCAP	10	2	2
17	simple-xcap-diff	draft 08	16	XCAP	15	6	6
18	simple-xcap-change-log	draft 00	12	XCAP	10	9	9
19	OMA-TS-Presence_SIMPLE	-	95	-	20	20	20
				Total:	**315**	**329**	**295**

Further criteria for the number of assigned test purposes were the complexity and the stage in the standardisation process of the specifications. It is useful to test complex specifications thoroughly ensuring that they are understood correctly by the implementor. In contrary to this, test purposes for incomplete or ambiguous specifications are less useful. In case of an update of the specification the test purposes have most likely to be updated as well.

The number of test purposes, which were derived from the protocol specifications, match closely the estimated numbers. The test purposes, which had to be derived from specifications written in prose, represent the mandatory attributes. Moreover, invalid test purposes were designed verifying if an implementation still works correctly if stimulated with erroneous protocol messages. In this way the SUT is tested on robustness as well.

For the derivation of the test purposes from the service specification the same technique as for base protocol specifications is used. Although there are a lot more theoretically combinations, the Open Mobile Alliance simplifies the process by providing certain scenarios, which can be used as test purposes [76].

5.3 Evaluation of Test Cases Implementation and Correctness

Test cases were developed by utilising the dedicated test programming language Testing and Test Control Notation (TTCN-3), which has already been successfully applied for implementing SIP test cases. Most of the test purposes from table 5.1 could be transferred into test cases. However, some of them could not implemented. These were mainly invalid test purposes which could be expressed with TTCN-3 but not interpreted by the codec.

Network elements with multiple interfaces implementing multiple protocols could easily be tested with TTCN-3 by creating a parallel test component (PTC) for each interface to be tested emulating different network elements simultaneously. Controlling and synchronising was simplified by instantiating a main test component (MTC).

A reference implementation was needed to evaluate a set of developed conformance test cases. Conformance test cases and a reference implementation should typically be developed in parallel (by independent developers). In case of the Presence Service only a few implementations, comprising only subsets of the functionality, existed. Thus, making a complete evaluation impossible.

In this report the developed test cases were evaluated with these incomplete implementations. Several errors could be detected - in the test cases as well as in the implementations - and fixed. However, for a better evaluation the used reference implementations have to be improved as well.

Executing opposite test cases with two workbenches is a good approach for detecting errors in sequences of exchanged messages. Applying this approach makes the development of further – in this report called opposite – test cases necessary by emulating the SUT on another workbench. For a valid test case a opposite test case, which is valid as well, most likely exists. In contrary to this, a developed opposite test case for an invalid test purpose would require an invalid opposite test case as well.

5.4 Test Coverage

The test coverage indicates to which extent the conformance requirements are covered by the developed test suite. In respect of service test suites and the assumptions of section 2.7, page 29, the following limitations exist:

- Protocol specifications are written in prose. There is no methodology for directly deriving test cases from the specification automatically.
- The number of test cases in a test suite - as well as in a service test suite - is limited, ensuring that the execution is still feasible.
- If the number of test cases is fixed, the most meaningful conformance requirements would have to be chosen. For this approach there is no technique for comparing the quality of different test cases.

In this thesis the quality of the test coverage was ensured by using the conformance testing methodology and a number of guidelines. For the assignment of test purpose numbers to protocol specifications the rules in sections 3.3 and page 5.2, 54 were utilised. By classifying the test purposes by functional aspects into groups and focussing on the lower layer specifications, a promising approach was utilised.

Furthermore, the test coverage of the Presence Service Test Suite – as well as for other service test suites – could be measured by comparing the developed test cases with the previously designed TSSs and the STSS. This method indicates if all functional aspects of each specification are covered with test purposes.

CHAPTER 6

Conclusions and Further Work

Conformance testing checks whether a network element implements its specifications correctly. It has been proven reliable and efficient for ensuring the interoperability between network elements implementing simple protocols such as SIP, but so far there does not exist a method for testing complex communication services.

Currently, interoperability testing is used to ensure the successful interoperation between network elements implementing complex services in Next Generation Networks (NGNs). Interoperability testing means a combinatoric explosion of tests, the more vendors and operators a market has. However, interoperability can only be ensured between tested network elements. In contrast to this, network elements having passed the conformance test are likely to interoperate with other network elements tested to be conform as well.

The main objective of this research was to propose and create a new conformance testing methodology for services in Next Generation Networks (NGNs). This was successfully achieved by using the ETSI's, already proven, conformance methodology for protocols and extending it by adding further methods.

Based on a critical review on the existing testing methods as well as the standardisation processes for protocol and service specifications a concept was proposed with the aim to enable conformance testing of services in Next Generation Network. One important aspect during this phase was the definition of a reasonable test coverage.

The created methods consider the different views from different standardisation organisations (i.e. IETF, 3GPP, OMA) on the NGN's architecture by developing a test suite comprising of three sets of test cases:

- Test cases for base protocols (IETF test cases)

- A profile for implementing protocols in a NGN (3GPP test cases)
- Test cases for the service out of an application-oriented perspective (OMA test cases)

For the derivation of test purposes from each of the protocol and service specifications test suite structures have to be designed building different groups. Based on these test suite structures, derived test purposes can systematically assigned to one of these groups. Derivation of test purposes were supported by defining a set of guidelines, controlling how to arrange a fix number of test purposes on all specifications. In a later step, the actually derived test purposes could be evaluated by checking if all groups of the test suite structures were represented.

One of the challenges during the implementation of the test purposes into test cases with the Testing and Test Control Notation (TTCN-3) was the test of network elements with multiple interfaces implementing different protocol specifications. This could be tested reliable by emulating multiple network elements. This was done with so called Parallel Test Components (PTCs), which were controlled and synchronised using a Main Test Component (MTC).

A practical evaluating of the developed test cases concerning the semantic was complicated because of the lack of reference implementations implementing all functions of the specifications. The test cases were successfully executed using currently available prototypes as SUTs which were supplied by the commercial partner of the project. Due to the fact that the SUTs only implemented a subset of the Presence Service's functions not every part of the test suite could be evaluated. Not until reference implementations are complete, the semantic of the developed test cases cannot be fully verified.

In a next step the new testing methodology was applied on the IMS' Presence Service. Presence, a fundamental service in Next Generation Networks, comprises several protocol specifications. One of the objectives was to evaluate the methodology by developing a Presence Test Suite. A set of approximately 300 test purposes were derived from the Presence Service's specifications and made executable by using TTCN-3. The Presence Test Suite was implemented within a cooperation with Testing Technologies IST, Berlin and is commercially available [77].

The application of the Presence Test Suite revealed several errors in the existing functions of the SUTs mainly caused by misunderstandings of the protocol specifications (see table 6.1). Only about 80% of the implemented functions worked according to the specifications. The test results of the two SUTs were in total even worse (for presence server 45% and for the the presence client 56%) clearly demonstrating the lack of adequate testing methods. Already now the application of the Presence Test Suite for the detection of conformance errors is an efficient asset to the development process and provides promising results.

Table 6.1: Evaluation of the Presence Test Suite

SUT	***Type***	***Test Cases***	***Passed***	***Inconclusive***	***Failed***	***Passed / Implemented***	***Total***
Oracle Communication Server	Presence Server	190	86	19	85	80%	45%
Oracle Communicator	Presence Client	105	59	32	14	77%	56%

The research objectives of this report have been accomplished. One result is a novel conformance testing methodology which can be used to develop sets of conformance test cases for service specifications in NGNs. During the development process of new services this new conformance testing methods significantly improves the chance of interoperability between network elements and provides a considerable enhancement to the currently utilised interoperability testing. Another result, gained by evaluating the new methods, is a set of conformance test cases for the IMS' Presence Service, which shows that the method is applicable. In the future further test suites for other IMS' services can be developed using the new methodology. It is planned to make the created test methodology as well as the test purposes for the Presence Service available to the ETSI with the aim to standardise them.

Bibliography

[1] M. Schmidt, A. Wilde, A. Schülke, and H. Costa, "IMS Interoperability and Conformance Aspects," *IEEE Communications Magazine*, vol. 45, pp. 138–142, March 2007.

[2] S. Hekmat, *Communication Networks*, PragSoft Corporation, Ed. PragSoft Corporation, 2005. [Online]. Available: www.pragsoft.com

[3] S. Ahson and M. Ilyas, Ed., *SIP Handbook: Services, Technologies, and Security of Session Initiation Protocol*. Boca Raton, FL: CRC Press, 2008.

[4] Telecommunication Standardization Sector of ITU, "OSI Conformance Testing Methodology and Framework for Protocol Recommendations for ITU-T Applications - General Concepts," International Telecommunication Union, ITU-T Recommendation X.290, April 1995.

[5] ——, "OSI Conformance Testing Methodology and Framework for Protocol Recommendations for ITU-T Applications - Abstract Test Suite Specification," International Telecommunication Union, ITU-T Recommendation X.291, April 1995.

[6] ——, "OSI Conformance Testing Methodology and Framework for Protocol Recommendations for ITU-T Applications - The Tree and Tabular Combined Notation (TTCN)," International Telecommunication Union, ITU-T Recommendation X.292, May 2002.

[7] ——, "OSI Conformance Testing Methodology and Framework for Protocol Recommendations for ITU-T Applications - Test Realization," International Telecommunication Union, ITU-T Recommendation X.293, April 1995.

[8] ——, "OSI Conformance Testing Methodology and Framework for Protocol Recommendations for ITU-T Applications - Requirements on Test Laboratories and Clients for the Conformance Assessment Process," International Telecommunica-

tion Union, ITU-T Recommendation X.294, April 1995.

[9] ——, "OSI Conformance Testing Methodology and Framework for Protocol Recommendations for ITU-T Applications - Protocol Profile Test Specification," International Telecommunication Union, ITU-T Recommendation X.295, April 1995.

[10] ——, "OSI Conformance Testing Methodology and Framework for Protocol Recommendations for ITU-T Applications - Implementation Conformance Statements," International Telecommunication Union, ITU-T Recommendation X.296, November 1995.

[11] International Organization for Standardization, "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General concepts," International Organization for Standardization, ISO/IEC 9646-1, 1994.

[12] ——, "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 2: Abstract Test Suite specification," International Organization for Standardization, ISO/IEC 9646-2, 1994.

[13] ——, "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 3: The Tree and Tabular Combined Notation (TTCN)," International Organization for Standardization, ISO/IEC 9646-3, 1998.

[14] ——, "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 4: Test realization," International Organization for Standardization, ISO/IEC 9646-4, 1994.

[15] ——, "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 5: Requirements on test laboratories and clients for the conformance assessment process," International Organization for Standardization, ISO/IEC 9646-5, 1994.

[16] ——, "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 6: Protocol profile test specification," International Organization for Standardization, ISO/IEC 9646-6, 1994.

[17] ——, "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 7: Implementation Conformance Statements," International Organization for Standardization, ISO/IEC 9646-7, 1995.

[18] European Telecommunications Standards Institute, "Methods for Testing and Specification (MTS); Protocol and profile conformance testing specifications; Standardization methodology," European Telecommunications Standards Institute, ETS 300 406, April 1995.

[19] H. Schulzrinne and J. Rosenberg, "The Session Initiation Protocol: Internet-Centric Signaling," *IEEE Communications Magazine*, vol. 38, no. 10, pp. 134–141, October 2000.

[20] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," Internet Engineering Task Force, Tech. Rep., June 2002. [Online]. Available: http://www.ietf.org/

[21] European Telecommunications Standards Institute, "Methods for Testing and Specification (MTS); Conformance Test Specification for SIP (IETF RFC 3261); Part 2: Test Suite Structure and Test Purposes (TSS&TP)," European Telecommunications Standards Institute, TS 102 027-2, July 2006.

[22] S. Moseley, S. Randall, and A. Wiles, "Experience within ETSI of the combined roles of conformance testing and interoperability testing," in *The 3rd Conference on Standardization and Innovation in Information Technology*, October 2003, pp. 177–189.

[23] M. El Maarabani, A. Adala, I. Hwang, and A. Cavalli, "Interoperability Testing of Presence Service on IMS Platform," in *TridentCom, Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009*, May 2009, pp. 1–6.

[24] N. Griffeth, R. Hao, D. Lee, and R. K. Sinha, "Interoperability Testing of VoIP Systems," in *IEEE GLOBECOM, Global Telecommunications Conference, 2000*, vol. 3, August 2000, pp. 1565–1570.

[25] R. Hao, D. Lee, R. K. Sinha, and N. Griffeth, "Integrated System Interoperability Testing With Applications to VoIP," *IEEE Transactions on Networking*, vol. 12, no. 5, pp. 823–836, October 2004.

[26] European Telecommunications Standards Institute, "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Presence service; Stage 1 (3GPP TS 22.141 version 8.0.0 Release 8)," European Telecommunications Standards Institute, TS 122 141, January 2009.

[27] Telecommunication Standardization Sector of ITU, “Next Generation Networks â Frameworks and functional architecture models - General overview of NGN,” International Telecommunication Union, ITU-T Recommendation Y.2001, December 2004.

[28] ——, “Next Generation Networks â Frameworks and functional architecture models - IMS for Next Generation Networks,” International Telecommunication Union, ITU-T Recommendation Y.2021, September 2006.

[29] M. Poikselkä, G. Mayer, H. Khartabil, and A. Niemi, *The IMS: IP Multimedia Concepts and Services.* John Wiley & Sons Ltd, 2007.

[30] D. Wermser, C. Büch, M. Hans, and F. Kowalewski, “Adaptive Mobile Multimedia-Dienste für das zukünftige All-IP UMTS,” *ITG Fachtagung "Ambient Intelligence"*, vol. 1, pp. 205–210, October 2004.

[31] P. Agrawal, J.-H. Yeh, J.-C. Chen, and T. Zhang, “IP Multimedia Subsystems in 3GPP and 3GPP2: Overview and Scalability Issues,” *IEEE Communications Magazine*, vol. 46, pp. 138–145, January 2008.

[32] C. J. Pavlovski, “Service Delivery Platforms in Practice,” *IEEE Communications Magazine*, vol. 45, pp. 114–121, March 2007.

[33] G. Camarillo and M. A. García-Martín, *The 3G IP Multimedia Subsystem (IMS): Merging the Internet and the Cellular Worlds*, 2nd ed. John Wiley & Sons Ltd, 2006.

[34] Open Mobile Alliance, “Presence SIMPLE Specification,” Open Mobile Alliance, Technical Specification Version 1.1, June 2008. [Online]. Available: http://www.openmobilealliance.org

[35] S. Chakraborty, J. Peisa, T. Frankkila, and P. Synnergren, *IMS Multimedia Telephony over Cellular Systems: VoIP Evolution in a Converged Telecommunication World.* John Wiley & Sons Ltd, 2007.

[36] European Telecommunications Standards Institute, “Universal Mobile Telecommunications System (UMTS); Presence Service; Architecture and functional description; Stage 2,” European Telecommunications Standards Institute, TS 123 141, October 2007.

[37] A. B. Roach, “Session Initiation Protocol (SIP)-Specific Event Notification,” Internet Engineering Task Force, RFC 3265, June 2002. [Online]. Available:

http://www.ietf.org/

[38] J. Rosenberg, "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)," Internet Engineering Task Force, RFC 4825, May 2007. [Online]. Available: http://www.ietf.org/

[39] J. Friedlander, K. Loganathan, R. Murphy, R. V. Pattabhiraman, and K. V. Vemuri, "Are you there? Reflections on Presence Server Architectures," *Bell Labs Technical Journal*, vol. 10, no. 4, pp. 77–82, 2006.

[40] H. Alvestrand, "A Mission Statement for the IETF," Internet Engineering Task Force, RFC 3935, October 2004. [Online]. Available: http://www.ietf.org/

[41] S. Bradner, "The Internet Standards Process - Revision 3," Internet Engineering Task Force, RFC 2026, October 1996. [Online]. Available: http://www.ietf.org/

[42] G. Thanos, A. Meliones, M. Marinidou, E. de la Fuente, and G. Konstantoulakis, "A 3GPP-SIP media gateway for the IP multimedia subsystem," in *Proc. 15th International Conference on Software, Telecommunications and Computer Networks SoftCOM 2007*, Sep. 27–29, 2007, pp. 1–5.

[43] Open Mobile Alliance, "OMA Organization and Process," Open Mobile Alliance, Tech. Rep. Version 1.5, October 2008. [Online]. Available: http://www.openmobilealliance.org/

[44] ——, "OMA Work Programme and Release Handling Processes," Open Mobile Alliance, Tech. Rep. Version 2.1, October 2008. [Online]. Available: http://www.openmobilealliance.org/

[45] ——, "OMA Interoperability Policy and Process," Open Mobile Alliance, Tech. Rep. Version 1.5, July 2007. [Online]. Available: http://www.openmobilealliance.org/

[46] B. W. Boehm, *Characteristics of Software Quality.* Elsevier Science Ltd, 1978.

[47] B. W. Boehm, J. R. Brown, and M. Lipow, "Quantitative evaluation of software quality," in *ICSE '76: Proceedings of the 2nd international conference on Software engineering.* Los Alamitos, CA, USA: IEEE Computer Society Press, 1976, pp. 592–605.

[48] W. R. Adrion, M. A. Branstad, and J. C. Cherniavsky, "Validation, Verification, and Testing of Computer Software," *Computing Surveys*, vol. 14, no. 2, pp. 159–92, 1982.

[49] B. W. Boehm, "Seven basic principles of software engineering," *Journal of Systems and Software*, vol. 3, pp. 3–24, 1983.

[50] J. Collofello and K. Vehathiri, "An environment for training computer science students on software testing," in *Proc. 35th Annual Conference Frontiers in Education FIE '05*, Oct. 19–22, 2005, pp. T3E–6.

[51] R. J. Linn, "Conformance Evaluation Methodology and Protocol Testing," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 7, pp. 1143–1158, September 1989.

[52] European Telecommunications Standards Institute, "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: TTCN-3 Core Language," European Telecommunications Standards Institute, ES 201 873-1, September 2008.

[53] ——, "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 2: TTCN-3 Tabular presentation Format (TFT)," European Telecommunications Standards Institute, ES 201 873-2, February 2007.

[54] ——, "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 3: TTCN-3 Graphical presentation Format (GFT)," European Telecommunications Standards Institute, ES 201 873-3, February 2007.

[55] ——, "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 4: TTCN-3 Operational Semantics," European Telecommunications Standards Institute, ES 201 873-4, September 2008.

[56] ——, "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 5: TTCN-3 Runtime Interface (TRI)," European Telecommunications Standards Institute, ES 201 873-5, April 2008.

[57] ——, "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 6: TTCN-3 Control Interface (TCI)," European Telecommunications Standards Institute, ES 201 873-6, September 2008.

[58] ——, "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 7: Using ASN.1 with TTCN-3," European Telecommunications Standards Institute, ES 201 873-7, April 2008.

[59] ——, "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 8: The IDL to TTCN-3 Mapping," European Telecom-

munications Standards Institute, ES 201 873-8, April 2008.

[60] ——, “Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 9: Using XML schema with TTCN-3,” European Telecommunications Standards Institute, ES 201 873-9, July 2008.

[61] ——, “Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 10: TTCN-3 Documentation Comment Specification,” European Telecommunications Standards Institute, ES 201 873-10, September 2008.

[62] Telecommunication Standardization Sector of ITU, “Data Networks and Open System Communications: Open Systems Interconnection - Service Definitions,” International Telecommunication Union, ITU-T Recommendation X.213, October 2001.

[63] C. Ware, “The OSI network layer: Standards to cope with the real world,” in *Proceedings of the IEEE*, vol. 71, no. 12, Dec. 1983, pp. 1384–1387.

[64] M. Ohba, “Software Quality = Test Accuracy x Test Coverage,” in *Proc. of 6th ICSE*, 1982, pp. 287–293.

[65] V. Basili and R. W. Selby, “Comparing the Effectiveness of Software Testing Strategies,” *IEEE Transactions on Software Engineering*, vol. SE-13, pp. 1278–1296, 1987.

[66] J. C. Huang, “An Approach to Program Testing,” *ACM Computing Surveys*, vol. 7, pp. 113–128, 1975.

[67] W. Stahl, “Packing Your Testing Toolbox,” *ComputerWorld*, pp. 83–89, 1989.

[68] D. Gelperin and B. Hetzel, “The Growth of Software Testing,” *Communications of the ACM*, vol. 31, pp. 687–695, 1988.

[69] P. Piwowarski, M. Ohba, and J. Caruso, “Coverage measurement experience during function test,” in *ICSE ’93: Proceedings of the 15th international conference on Software Engineering*. Los Alamitos, CA, USA: IEEE Computer Society Press, 1993, pp. 287–301.

[70] C. Hagwood and L. Rosenthal, “Reliability of Conformance Tests,” *IEEE Transactions on Reliability*, vol. 50, no. 2, pp. 204–208, June 2001.

[71] M. Marré and A. Bertolino, “Using Spanning Sets for Coverage Testing,” *IEEE Transactions on Software Engineering*, vol. 29, no. 11, pp. 974–984, November 2003.

[72] B. K. Aichernig, B. Peischl, M. Weiglhofer, and F. Wotawa, “Protocol Conformance

Testing a SIP Registrar: an Industrial Application of Formal Methods," in *Proc. Fifth IEEE International Conference on Software Engineering and Formal Methods SEFM 2007*, September 2007, pp. 215–226.

[73] D. Eric, E. Fong, and A. Goldfine, "Requirements for GSC-IS reference implementations," National Institute of Standards and Technology, Tech. Rep., 2003.

[74] P. Curran, "Conformance Testing: An Industry Perspective," Sun Microsystems, 2003.

[75] A. Niemi, "Session Initiation Protocol (SIP) Extension for Event State Publication," Internet Engineering Task Force, RFC 3903, October 2004. [Online]. Available: http://www.ietf.org/

[76] Open Mobile Alliance, "Enabler Test Specification for Presence SIMPLE: Conformance test cases," Open Mobile Alliance, Test Specification Version 1.1, March 2008. [Online]. Available: http://www.openmobilealliance.org

[77] Testing Technologies IST GmbH, "TTsuite-Presence," 2008. [Online]. Available: http://www.testingtech.com/solutions/ttsuite-presence_w.php

List of Figures

List of Tables

Glossary

Term	***Definition***
Conformance Test Suite	A complete set of test cases, possibly combined into nested test groups, that is needed to perform dynamic conformance testing for one or more OSI protocols.
Conformance Testing	Testing the extent to which an IUT is a conforming implementation.
Conformance Test Specification	One or more specifications that contain a standardised ATS, together with its related TSS and TP, partial IXIT proforma, and TMP specification, if any.
Conforming Implementation	An IUT which satisfies both static and dynamic conformance requirements, consistent with the capabilities stated in the ICS(s).
Conforming System	A real system which satisfies both static and dynamic conformance requirements consistent with the capabilities stated in the ICS(s) referenced by the SCS.
Executable Test Case	A realisation of an abstract test case.
Executable Test Suite (ETS)	A test suite composed of executable test cases.
Fail (verdict)	A test verdict given when the observed test outcome either demonstrates non-conformance with respect to (at least one of) the conformance requirement(s) on which the test purpose of the test case is focused, or contains at least one invalid test event, with respect to the relevant specification(s).
Implementation Conformance Statement (ICS)	A statement made by the supplier of an implementation or system claimed to conform to a given specification, stating which capabilities have been implemented. The ICS can take several forms: protocol ICS, profile ICS, profile specific ICS, and information object ICS.
Implementation Conformance Statement (ICS) Proforma	A document, in the form of a questionnaire, which when completed for an implementation or system becomes an ICS.

Continues on next page

Term	***Definition***
Implementation Extra Information for Testing (IXIT)	A statement made by a supplier or implementer of an IUT which contains or references all of the information (in addition to that given in the ICS) related to the IUT and its testing environment, which will enable the test laboratory to run an appropriate test suite against the IUT. An IXIT can take several forms: protocol IXIT, profile IXIT, profile specific IXIT, and information object IXIT, TMP implementation statement.
Implementation Extra Information for Testing (IXIT) Proforma	A document, in the form of a questionnaire, which when completed for an IUT or SUT becomes an IXIT.
Implementation Under Test (IUT)	An implementation of one or more OSI protocols in an adjacent user/provider relationship, being that part of a real open system which is to be studied by testing.
Inconclusive (verdict)	A test verdict given when the observed test outcome is such that neither a pass nor a fail verdict can be given.
Initial Testing State	The testing state in which a test body starts.
Inopportune Test Event	A test event which occurs when not allowed to do so by the relevant specification(s) to which conformance is being tested.
Invalid Test Event	A test event that violates at least one conformance requirement of the relevant specification(s) to which conformance is being tested.
Multi-protocol Testing	Testing of more than one protocol within the IUT by means of test cases which have test purposes which cover conformance requirements that relate to more than one protocol.
Multi-specification Dependency	A conformance requirement in one specification which specifies a requirement upon the support of another specification within a conforming system.
Next Generation Network	A packet-based network able to provide telecommunication services and able to make use of multiple broadband, QoS-enabled transport technologies and in which service-related functions are independent from underlying transport-related technologies. It enables unfettered access for users to networks and to competing service providers and/or services of their choice. It supports generalised mobility which will allow consistent and ubiquitous provision of services to users.
Pass (verdict)	A test verdict given when the observed test outcome gives evidence of conformance to the conformance requirement(s) on which the test purpose of the test case is focused, and when no invalid test event has been detected.
Repeatability (of results)	Characteristic of a test case, such that repeated executions on the same IUT under the same conditions lead to the same test verdict, and by extension a characteristic of a test suite.
Service	A set of functions and facilities offered to a user by a provider.

Continues on next page

Term	***Definition***
Single-protocol Testing	Testing specified for a single protocol within a single-protocol or multi-protocol IUT.
Standardised Abstract Test Suite; Standardised ATS	An ATS specified within an ITU-T or ISO/IEC published specification or, in the absence of such a specification, within a publicly available specification which is in the process of being standardised within ITU-T or ISO/IEC, and which has the highest standardisation status available, and which has the status of at least a Committee Draft or equivalent.
System Under Test (SUT)	The real open system in which the IUT resides.
Test Campaign	The process of executing the Parameterised Executable Test Suite for a particular IUT and producing the conformance log.
Test Case	An abstract or executable test case.
Test Case Error	The term used to describe the result of execution of a test case when an error is detected in the test case itself.
Test Event	An indivisible unit of test specification at the level of abstraction of the specification (e.g. sending or receiving a single PDU).
Test Group	A named set of related test cases.
Test Postamble	The sequences of test events from the end of the test body up to the finishing stable testing state(s) for the test case.
Test Preamble	The sequences of test events from the starting stable testing state of the test case up to the initial testing state from which the test body will start.
Test Purpose	A prose description of a well defined objective of testing, focusing on a single conformance requirement or a set of related conformance requirements as specified in the appropriate OSI specification (e.g. verifying the support of a specific value of a specific parameter).
Test Realiser	An organisation which takes responsibility for providing, in a form independent of the clients of a test laboratory and their IUTs, a Means of Testing IUTs in conformance with an ATS.
Testing State	A state encountered during testing, comprising the combination of the states of the SUT, the test system, the protocols for which control and observation is specified in the ATS, and, if relevant, the state of the underlying service.
Test Step	A named subdivision of a test case, constructed from test events and/or other test steps.
Test System	The real system which includes the realisation of the Lower Tester.
Test Verdict	A statement of pass, fail or inconclusive, as specified in an abstract test case, concerning conformance of an IUT with respect to that test case when it is executed.

Continues on next page

Term	***Definition***
Unidentified Test Event	A test event which is used to provide for receipt of PDUs and/or ASPs without identifying them explicitly in the test case.
Valid Test Event	A test event which is allowed by the protocol specification, being both syntactically and semantically correct, and occurring when allowed to do so by the protocol specification.

List of Acronyms

Acronym	*Meaning*
3GPP	Third Generation Partnership Project
ADSL	Asynchronous Subscriber Line
AS	Application Server
CDMA	Code Division Multiple Access
CSCF	Call Session Control Function
CTP	Combined Test Purpose
ETSI	European Telecommunications Standards Institute
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication
HSS	Home Subscriber Server
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IMS	IP Multimedia Subsystem
ISO	International Organization for Standardization
ITU	International Organization for Standardization
IUT	Implementation under Test
MTC	Main Test Component
NGN	Next Generation Network
OMA	Open Mobile Alliance
PSTN	Public Switched Telephone Network
PTC	Parallel Test Component
QoS	Quality of Service
RFC	Request for Comments

Continues on next page

Acronym	***Meaning***
RTP	Real-Time Transport Protocol
SDP	Session Description Protocol
SIP	Session Initiation Protocol
STP	Service Test Purpose
STSS	Service Test Suite Structure
SUT	System under Test
TP	Test Purpose
TS	Test Suite
TSS	Test Suite Structure
TTCN-3	Testing and Test Control Notation Version 3
UMTS	Universal Mobile Telecommunications System
VoIP	Voice over IP
WLAN	Wireless Local-area Network
XCAP	XML Configuration Access Protocol
XML	Extensible Markup Language

APPENDIX A

Acronyms for Presence Test Case Nomenclature

Table A.1: Test Categorie

Acronym	*Meaning*
TC	Test Case
STC	Service Test Case

Table A.2: Protocol Specifications

Acronym	*Meaning*
EVNT	Event Notification (RFC 3265)
PUB	Publication (RFC 3903)
PRES	Presence (RFC 3856)
PDM	Presence Data Model (RFC 4479)
PIDF	Presence Information Data Format (RFC 3863)
RPID	Rich Presence Extension to the Presence Information Data Format (RFC 4480)
CIN	Content Indirection (RFC 4483)
SEL	SIP Event Lists (RFC 4662)
XCAP	XML Configuration Access Protocol (RFC 4825)
XRL	XCAP Resource Lists (RFC 4826)
XCP	(XCAP) Common Policy (RFC 4745)
XPA	(XCAP) Presence Authorisation (RFC 5025)

Continues on next page

Acronym	***Meaning***
SCF	SIP Configuration Framework (Draft V 15)
PVX	SIP UA Profiles via XCAP (Draft V 00)
SCPC	Simple Common Policy Capabilities (Draft V 02)
SPPC	Simple Presence Policy Capabilities (Draft V 01)
PX	Presence-XCAP Interaction

Table A.3: Roles of System Under Test

Acronym	***Meaning***
SUB	SUT acts as a Subscriber
NOT	SUT acts as a Notifier
EPA	SUT acts as an Event Publication Agent
ESC	SUT acts as an Event State Compositor
RLS	Resource List Server
LIST	XCAP Resource List Server
SERVICE	XCAP Resource List Server Services
XPS	XCAP Policy Server
PS	Presence Server

Table A.4: Nature of Test

Acronym	***Meaning***
V	Valid Test Case
I	Invalid Test Case
TI	Timing Test Case

Evaluation of Impact of Test Methods for Ensuring Interoperability between Network Elements in Next Generation Networks

Matthias Bormann

Second report is part of a portfolio and submitted in partial fulfilment of the requirements of the University of Glamorgan for the degree of Master of Philosophy

Faculty of Advanced Technology
University of Glamorgan
Pontypridd
CF37 1DL

June 29, 2010

Certificate of Research

This is to certify that, except where specific reference is made, the work described in this thesis is the result of the candidate. Neither this thesis, nor any part of it, has been presented, or is currently submitted, in candidature for any degree at any other University.

Signed ....
Candidate

Signed ....
Director of Studies

Date ....

Abstract

Next Generation Networks (NGNs) offer a generic architecture for *Voice over IP* (VoIP) as well as multimedia services based on a *Session Initiation Protocol* (SIP) infrastructure. They provide a flexible way to deliver services in a standardised manner, accelerating the creation of services. Experience shows that there is a demand for new services [1]. Especially the number of applications that provide multimedia services are growing rapidly [2].

Ensuring the interworking between all network elements in Next Generation Networks is an important aspect. To this end conformance and interoperability testing are common approaches. Both testing methods are typically conducted in a laboratory under controlled conditions. However, several errors of network elements in communication networks can only be detected while being in operation such as when reaching its load limit.

On the one hand this report discusses and compares the mentioned existing testing methods and presents the tool IMPACT (IP-based Multi Protocol Post-hoc Analyzer and Conformance Tester), which was developed at the Ostfalia University in Germany. With IMPACT, interoperability testing, combined with a conformance verification, can be carried out between network elements in daily operation. Therefore, IMPACT parses traces of *protocol data units* (PDUs), recorded during operation, and stores them in a relational database. Formulating rules with SQL, errors in protocol sequences can be detected. An efficient analysis on records containing many PDUs is only possible by using a database, which is one of the big differences between IMPACT and other network traffic analyser.

On the other hand an approach is taken to combine the advantages of the three discussed testing methods with a process model, which is used for designing large scale software systems. Additionally, detected errors in one testing method can be used to implement missing test cases in a previous test set and further improve the test coverage as well as

the confidence that the interoperability is ensured in a communication network.

Contents

CHAPTER 1

Introduction

The first report of this thesis considered ETSI's conformance testing methodology, which can be applied to Internet protocols. A novel concept was created extending this methodology in a way that it can be used for complex services as well. This report examines different testing methods with the aim to further improve the interoperability between network elements in *Next Generation Networks* (NGNs) such as conformance or interoperability testing.

Interoperability testing verifies that two different implementations of the same protocol have the capability of interworking. It is used for testing prototypes built on protocol specifications, such as Internet Engineering Task Force's Request for Comments (RFCs), and products implementing Internet protocol standards. Interoperability testing is a well accepted verification method in the IETF so that a protocol draft can be an IETF standard only, if there exist at least two interoperating implementations for it [3, 4].

Conformance testing is applied in the telecommunication sector and has already proven its stability over years. ETSI, ITU, 3GPP and other standardisation bodies develop conformance test suites [5]. Vendors of telecom equipment or operators - the customers - are applying these conformance test suites to show conformance of the products. Interoperability testing is also done after the conformance tests. Its main function is to verify if a new network element can interoperate with the other nodes of the network on the main operation level.

If two *Implementations under Test* (IUT) pass the same conformance test suite they can very likely interoperate as well. That is how conformance testing is defined [6]. But conformance test suites cannot cover the protocol's functionalities completely. It may happen that interoperability testing of two IUTs fails even if the IUTs passed the conformance test. Especially, if the conformance test suite is not well proven. So it may happen that the interoperability test of two IUTs fails even if the IUTs passed the conformance test before. Protocol specifications contain optional elements. It may happen that the two IUTs implement or do not implement the optional features in a way that in certain cases this leads to problems in interoperability.

Furthermore, the situation may occur where two IUTs can interoperate using a protocol, but one or both fail on a conformance test. There can be invalid behaviours that were not tested during interoperability testing in which the IUTs fail [7]. Invalid behaviour can be triggered by conformance testing but usually not with interoperability testing. Another possibility for non-conforming IUTs, which can interoperate with each other is that the implementers misunderstood the protocol standard in the same way. So, the two IUTs can interoperate but conformance testing fails for both of them. In this case it is very likely that interoperability testing with other IUTs would also fail.

Interoperability testing is a method widely acknowledged by the telecommunication industry. Configuration of an implementation for interoperability testing has typically be done manually [3]. Moreover, at the end of each interoperability test case the analysis of the logs is also made by hand. For the evaluation of the test result knowledge of the test system as well as the test case is required. This implies that only the developer of the implementation can perform the interoperability testing.

Although developed network elements are analysed using conformance or interoperability testing (or even both) these two testing methods can't guarantee the interoperability. Both testing methods are typically conducted in a laboratory under controlled conditions. A network element, outside of a dedicated test environment, is open to a bigger variety of faulty effects such as when reaching its load limits.

Errors found in operation have to be detected and their cause analysed. This is typically done manually, takes a lot of time and is inefficient. To fix these errors a quick and reliable detection of their cause is of utmost importance and therefore, for the customer's satisfaction. For this purpose additional methods – so called post-hoc analyses – as well as dedicated tools are needed.

The research objectives of this project are:

- Critical comparison of conformance and interoperability testing methods that ensure the interoperability in a communication network
- Creation of post-hoc analysis methodologies for the efficient detection of faulty message sequences

- Improvement of the interoperability between the interaction of network elements by systematically combining the three different testing methodologies

This report is structured to reflect these objectives. The next chapter presents the literature review and is followed by a chapter covering the methods on how to improve the interoperability of developed network elements. An approach on how to efficiently detect interoperability errors of network elements in operation is presented in the next chapter. This is followed by a model on how to systematically combine testing methods to further improve the interoperability. The report continues with a discussion of the results and ends with a conclusion and the future work.

CHAPTER 2

Literature Review

This chapter gives an overview and a critical evaluation of selected testing methods for ensuring the interoperability in *Next Generation Networks* (NGNs).

First, the architecture of *Next Generation Networks* (NGNs) is described. Next, models for the software development process are being introduced and conformance and interoperability testing presented and compared. Then a third method, testing of network elements in operation, is shown. Finally, methods for ensuring software quality are considered based on analyses of the literature. It is investigated to what extent these methods are applicable in the telecommunication domain.

2.1 Next Generation Networks

NGNs are an architectural concept that uses one single network to transport all information and services. Information, such as media and data, is encapsulated into packets, like it is on the Internet. The characteristics of NGNs are specified by the *International Telecommunication Union* (ITU) [8]. One of the most important characteristics is the independence of service-related functions from underlying transport technologies, which means that in NGN Internet services can be accessed everywhere and at every time using different, mostly cellular, technologies.

One implementation of a NGN is the *IP Multimedia Subsystem* which provides an easy and flexible way to deliver services in a standardised manner accelerating the creation of services [9]. A service provides a function [10, 11] by utilising one or more protocols. Voice over IP (VoIP) as an example utilises among others the *Session Initiation Protocol* (SIP) and the *Real-Time Transport Protocol* (RTP) [12].

A protocol in the telecommunication system's domain is defined by a set of rules for data representation, signalling, authentication and error detection required to send information over a communications channel [13]. These protocols have features intended to ensure reliable interchange of data over an imperfect communication channel.

2.2 Models for the Software Development Process

A software development process or life cycle is a structure imposed on the development of a software product. There are several models for such processes, each describing approaches to a variety of tasks or activities that take place during the process.

2.2.1 Waterfall Model

The *Waterfall Model* is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design (validation), Construction, Testing and Maintenance.

The Waterfall Model has its origins in the manufacturing and construction industries; highly structured physical environments in which after-the-fact changes are prohibitively costly, if not impossible. Since no formal software development methodologies existed at the time, this hardware-oriented model was simply adapted for software development.

The first formal description of the Waterfall Model is often cited to be an article published in 1970 [14] although Royce did not use the term "waterfall" in this article.

To follow the Waterfall Model, one proceeds from one phase to the next in a purely sequential manner. For example, one first completes requirements specification. When the requirements are fully completed, one proceeds to design. The software is designed and a blueprint is drawn for the implementers. This design should be a plan for implementing the requirements given. When the design is fully completed, an implementation of that design is implemented. Towards the later stages of this implementation phase, separate software components produced are combined to introduce new functionality and reduced risk through the removal of errors.

Thus the Waterfall Model maintains that one should only move to a phase when its preceding phase is completed and perfected. However, there are various modified Waterfall Models that may include slight or major variations upon this process.

2.2.2 Spiral Model

The *Spiral Model* is a software development process combining elements of both design and prototyping in stages, in an effort to combine advantages of top-down and bottom-up concepts (see figure 2.1). Also known as the spiral life cycle model, it is a systems development method (SDM) used in information technology. This model of development combines the features of the prototyping model and the waterfall model. The spiral model is intended for large, expensive and complicated projects [15, 16]. The spiral model was defined by Barry Boehm in 1988 [17]. This model was the first model to explain an iterative development.

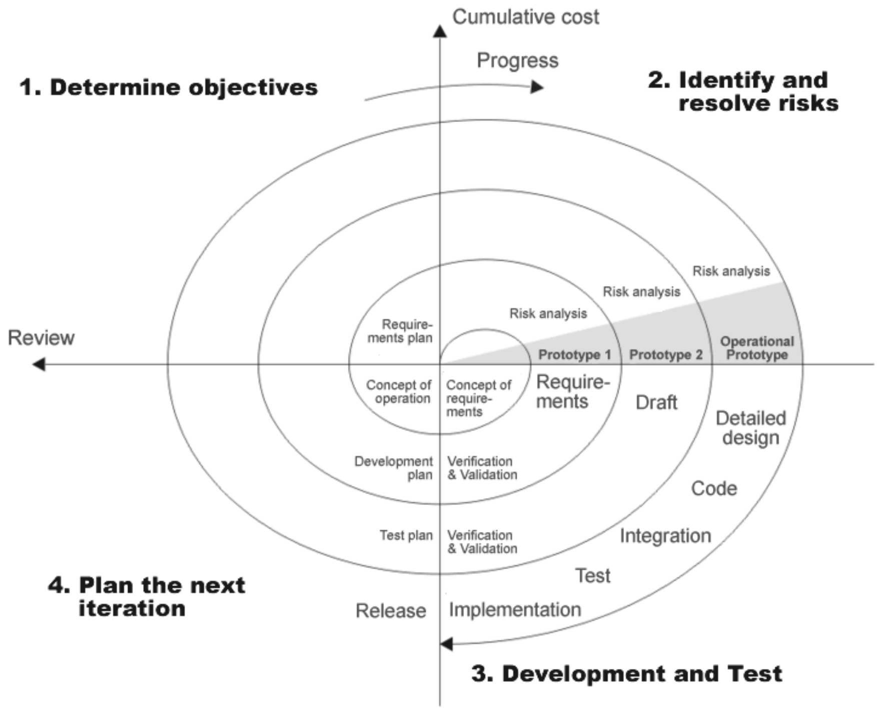

Figure 2.1: Spiral Model by Barry Boehm

The steps in the spiral model can be generalised as follows:

1. The new system requirements are defined in as much detail as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.

2. A preliminary design is created for the new system. This phase is the most important part of Spiral Model. In this phase all possible (and available) alternatives, which can help in developing a cost effective project are analysed and strategies

are decided to use them. This phase has been added specially in order to identify and resolve all the possible risks in the project development. If risks indicate any kind of uncertainty in requirements, prototyping may be used to proceed with the available data and find out possible solution in order to deal with the potential changes in the requirements.

3. A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.

4. A second prototype is evolved by a fourfold procedure:

   a) evaluating the first prototype in terms of its strengths, weaknesses, and risks;

   b) defining the requirements of the second prototype;

   c) planning and designing the second prototype;

   d) constructing and testing the second prototype.

2.2.3 V-Model

Although not a standard, the V-Model is a very popular model of the system engineering process. The original V-Model is shown in figure 2.2 [18]. For top-down system engineering (i.e. forward engineering), the process starts on the upper left and goes to the upper right. For bottom-up (i.e. reverse engineering), it starts on the upper right and goes to the upper left [19].

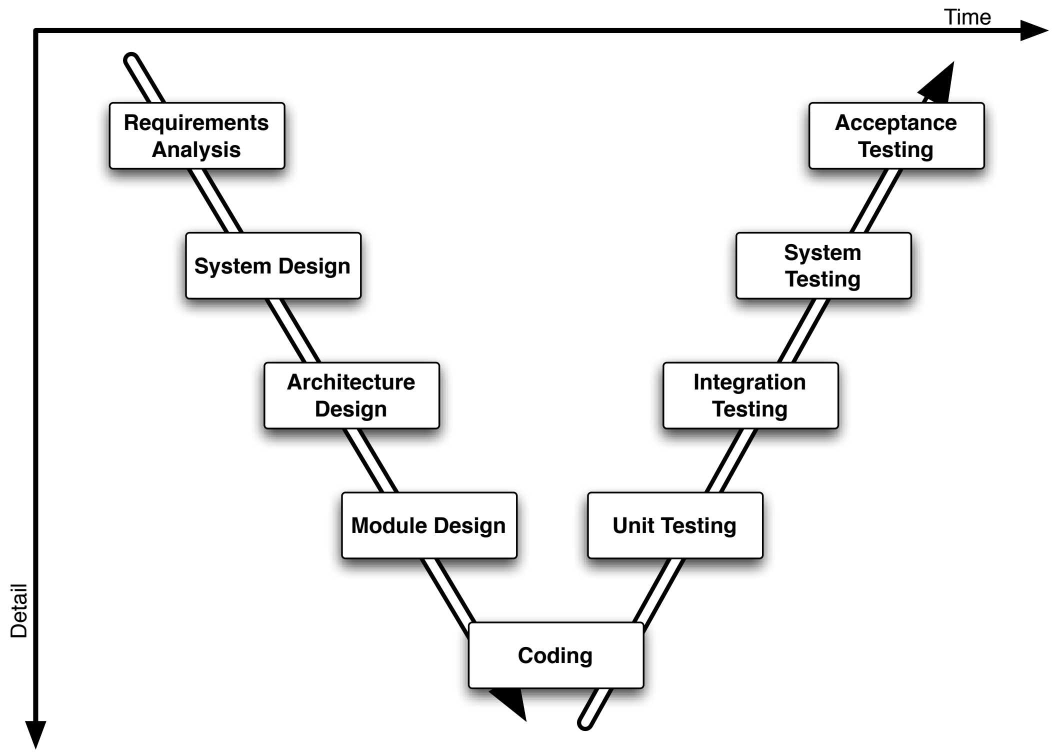

Figure 2.2: V-Model

Starting from the bottom the first test level is *Component Testing*, sometimes called *Unit Testing*. It involves checking that each feature in the specification has been implemented in the component.

In theory an independent tester should do this, but in practise the developer usually does it, as they are the only people who understand how a component works. The problem with a component is that it performs only a small part of the functionality of a system, and it relies on co-operating with other parts of the system, which may not have been built yet. To overcome this, special software is used that the component believes it is working in a fully functional system.

As the components are constructed and tested they are then linked together to check if they work with each other. It is a fact that two components that have passed all their tests, need not to be working anymore when combined. New tests have to be developed and are typically applied by the developers.

Integration Testing is not focussed on what the components are doing but on how they communicate with each other, as specified in the system's specification. This specification defines relationships between components, and involving:

- What a component can expect from another component in terms of services.
- How these services will be asked for.
- How they will be given.
- How to handle non-standard conditions, i.e. errors.

Tests are constructed to deal with each of these. The tests are organised to check all the interfaces, until all the components have been built and connected to each other producing the whole system.

Once the entire system has been built it has to be tested against the system's specification to check if it delivers the features required. It is still developer focussed, although specialist developers known as systems testers are normally employed to do it.

In essence *System Testing* is not about checking the individual parts of the design, but about checking the system as a whole. System testing can involve a number of specialist types of test to see if all the functional and non-functional requirements have been met. In addition to functional requirements these may include the following types of testing for the non-functional requirements:

- Performance - Are the performance criteria met?
- Volume - Can large volumes of information be handled?
- Stress - Can peak volumes of information be handled?
- Robustness - Does the system remain stable under adverse circumstances?

2.3 Protocol Testing Methods

In the telecommunication system's domain software development is mainly concerned with the correct implementation of protocol specifications. These specifications are – in contrary to software specifications in other domains – standardised and openly available. The major advantage of standardisation enables the possibility that implementations of the same standard but from different developers can successfully interoperate.

However, to ensure the interoperability between the products of different developers the implementations have to be tested as well. In order to be able to compare the quality of implementations from different developers, independently designed methods and techniques are needed. In the telecommunication domain there are several testing methods and techniques available such as conformance and interoperability testing.

2.3.1 Conformance Testing

One approach to assure the interoperability between protocol entities is conformance testing. Conformance testing is used for verifying that an implementation works according to its relevant protocol specifications [6]. The external behaviour of an implementation, the system under test (SUT) – as shown in figure 2.3 – is tested in order to find logical errors and ensure the interoperability with further systems tested to be conform as well.

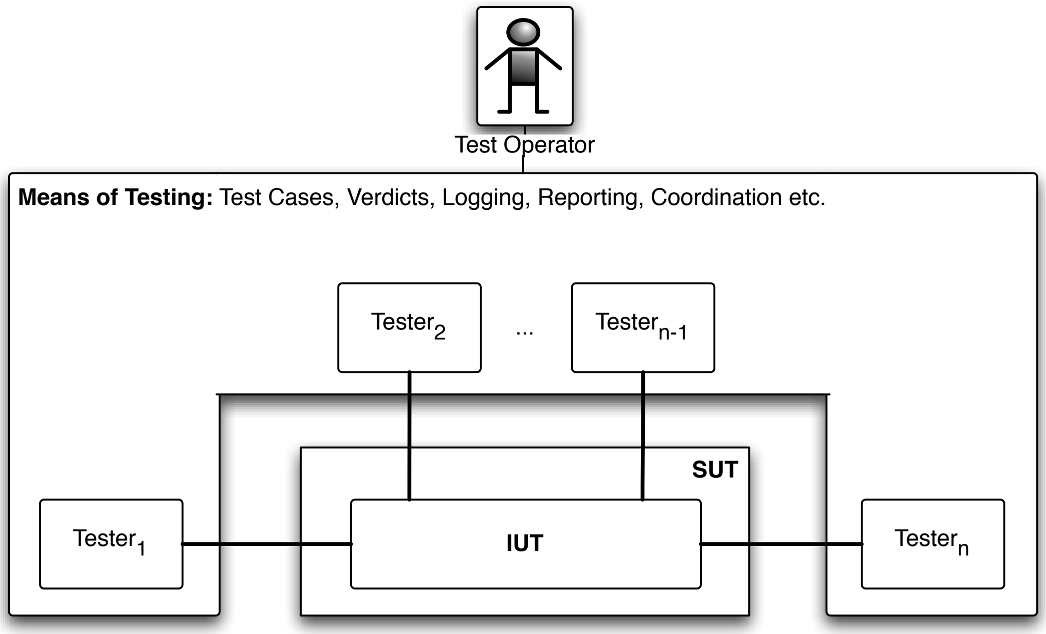

Figure 2.3: Abstract Conformance Test Method. Modified from [3].

Figure 2.4 shows a SUT, which implements four different protocol specifications (A-D). Using conformance testing each of these protocols can be tested with respect to the test coverage of the set of test cases.

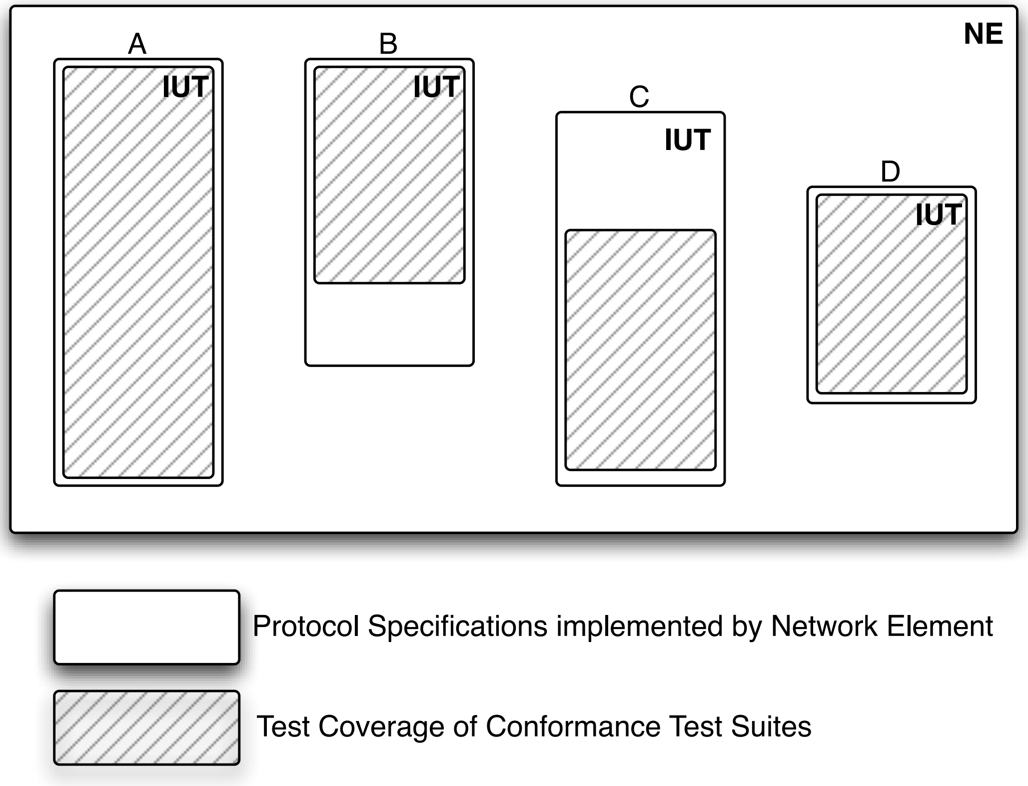

Figure 2.4: Scope of Conformance Testing. Modified from [3].

In order to test an implementation, test purposes have to be derived from the protocol specification. These test purposes have to be unambiguous and should be reusable. For the execution these purposes have to be written down in a formal way and then be made executable. Therefore, a formal testing language like TTCN-3 (Testing and Test Control Notation) is utilised [20]. To test the implementation, an executable test environment is necessary. This task is done by a workbench, which compiles the TTCN-3 into an executable format and manages the logging of sent messages during test execution. The result is summarised in a test report. The process on how to develop and implement conformance test cases is standardised by several organisation such as the ETSI [21] and illustrated in figure 2.5.

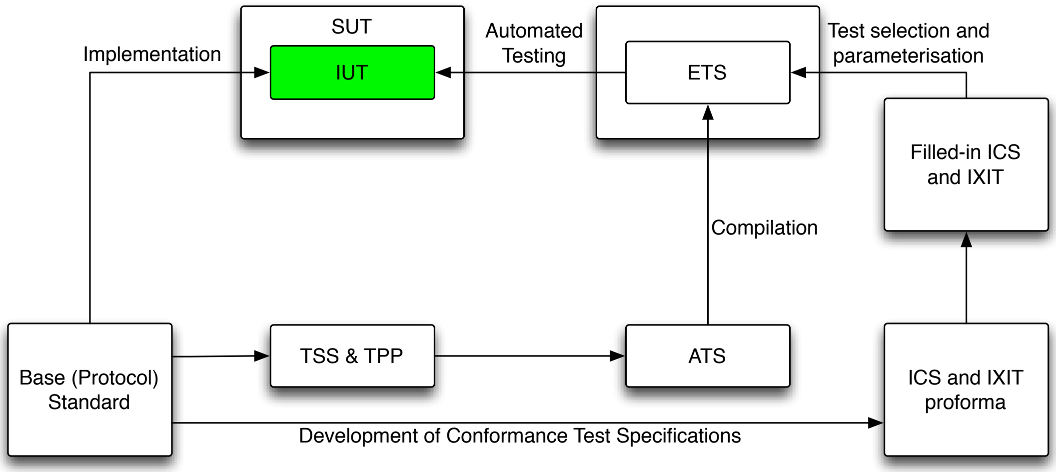

Figure 2.5: Conformance Testing Process. Modified from [3].

An example of such a test purpose for RFC 3261 [22] is shown in table 2.1. Each of this test purposes describes one single solitary aspect of the behaviour stipulated in the protocol specification. The TTCN-3 code for this test purpose is illustrated in figure 2.6.

Table 2.1: SIP Test Purpose - Valid Behaviour [23]

Test Purpose ID	SIP_TC_TE_CE_V_001
Status	Mandatory
Reference	RFC 3261, sections 8, 8.2, and 13.3.1.1
Description	Ensure that the IUT on receipt of an INVITE request, sends a success (200 OK) or provisional (100-199) response.
Test Steps	1. Send INVITE request to the IUT. 2. Wait for 200 OK or 100-199 response from the IUT.

```
testcase SIP_CC_TE_CE_V_001() runs on SipTcComponent {
      (...)
 SIPP.send (INVITE_Request_s_2(v_CallId,loc_CSeq_s));
 TResp.start (PX_TRESP);
 action ("Please hook off");
 alt {
 [] SIPP.receive (Response_101_199_r_1(v_CallId,loc_CSeq_s)) {
      setverdict (pass);
      (...)
    }
 [] SIPP.receive (Response_200_r_1(v_CallId,loc_CSeq_s)) {
      setverdict (pass);
      (...)
    }
 [] SIPP.receive (Response_100_r_1(v_CallId, loc_CSeq_s)) {
      repeat;
    }
 [] TResp.timeout {setverdict (fail);
      stop;
    }
}
```

1: Stimulation part
2: Reaction part

Figure 2.6: SIP Test Case in TTCN-3

2.3.2 Interoperability Testing

Compared to conformance testing, interoperability testing does not check whether a single protocol is used correctly, but if whole devices comprising of several protocol state machines cooperate. Devices, equipments under test (EUTs), are connected with each other and form the system under test (SUT) (see figure 2.7).

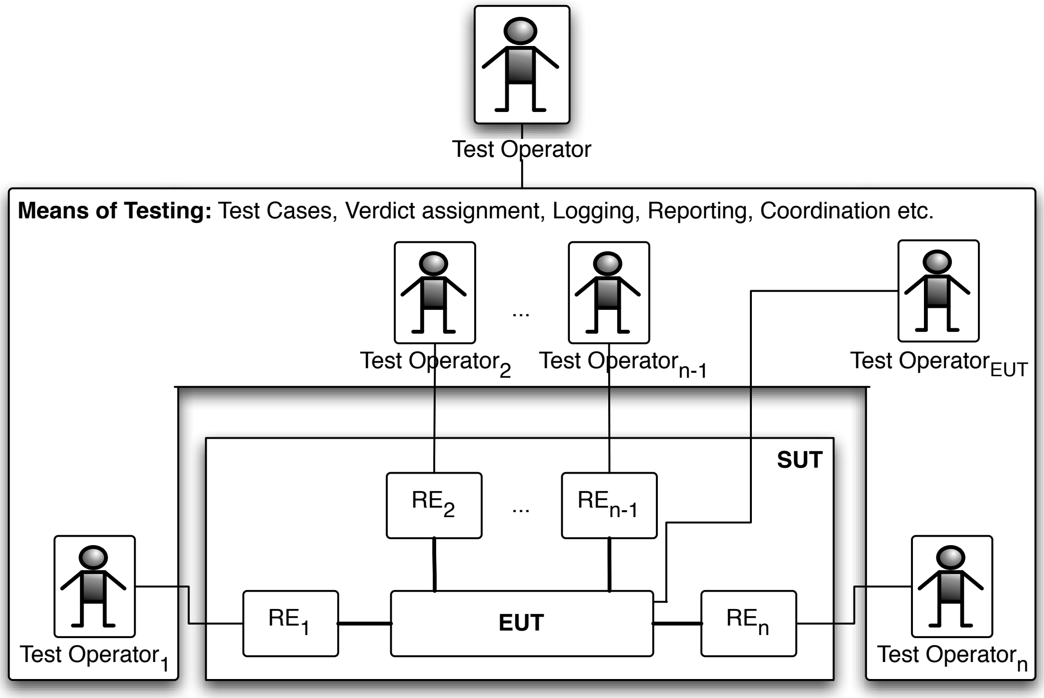

Figure 2.7: Abstract Interoperability Test Method. Modified from [3].

Focus of the test is the interaction of several protocol instances in an EUT as shown in figure 2.8. While conformance testing checks a single protocol instance thoroughly, interoperability testing is wider and less deep.

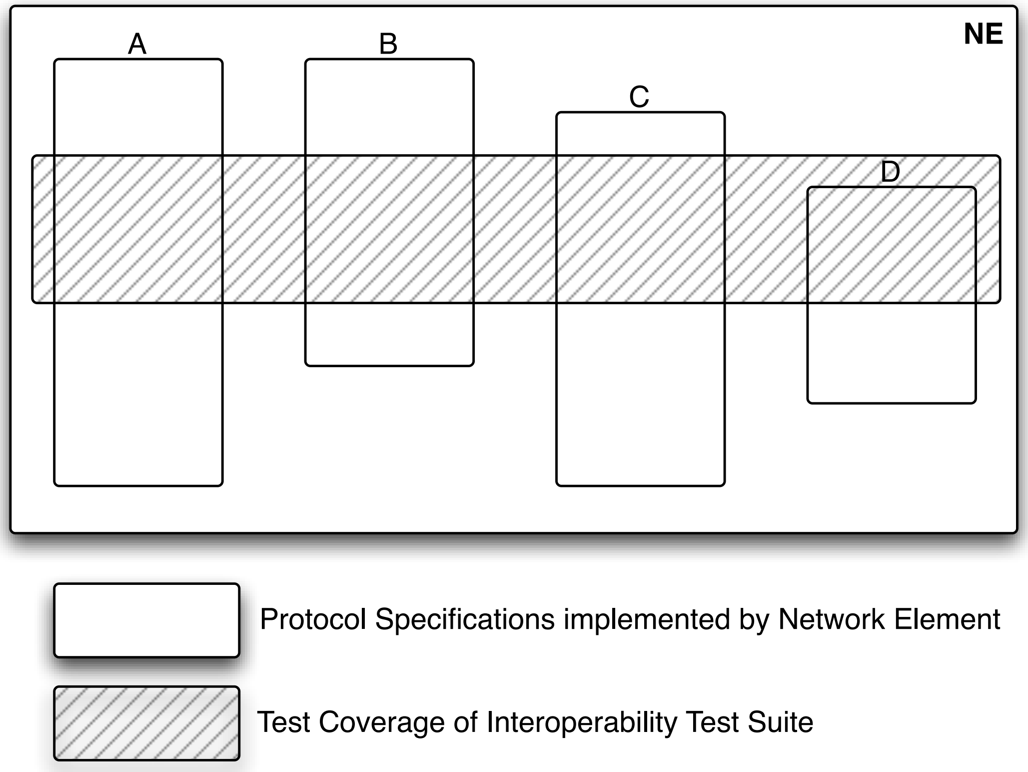

Figure 2.8: Scope of Interoperability Testing. Modified from [3].

One interoperability test scenario is defined as the sequence of user inputs, i.e. dialling a number or picking up the handset. The actual behaviour is compared with the expected behaviour defined in the scenario (i.e. the ringing of the phone). An interoperability test case for Voice over IP (VoIP) is shown in figure 2.9. If these behaviours match, the EUT passes the test scenario. For interoperability testing no dedicated test equipment, such as a workbench, is needed.

Test: 01			
Title:	Voice Call Establishment from User A to User B		
Test Purpose:	To verify that a call can be established successfully to User 8 by User A and that speech communication is possible between User A and User B		
Pre-test conditions:	Configure Network Elements to support at least Codec G.711		
Step	**Test Description**	**Verdict**	
		pass	**fail**
1	Initiate a new call from User A to User B	-	-
2	A: Is dial tone heard?	Yes	No
3	B: Is the phone ringing?	Yes	No
4	User B answers the phone	-	-
5	A: Is dial tone heard?	No	Yes
6	B: Is the phone ringing?	No	Yes
7	User A talks	-	-
8	B: Can User A be heard by User B?	Yes	No
9	user B talks	-	-
10	A: Can User B be heard by User A?	Yes	No
11	Clear the call	-	-

Figure 2.9: Interoperability Test Case for VoIP

Scenarios for interoperability testing are, as with conformance testing, derived from the protocol specification. When creating these scenarios it has to be considered that the EUTs can only be stimulated over the user interface. A purposeful stimulation of the EUT with faulty protocol messages (invalid behaviour) cannot be done. The process is defined by the ETSI [24] and shown in figure 2.10.

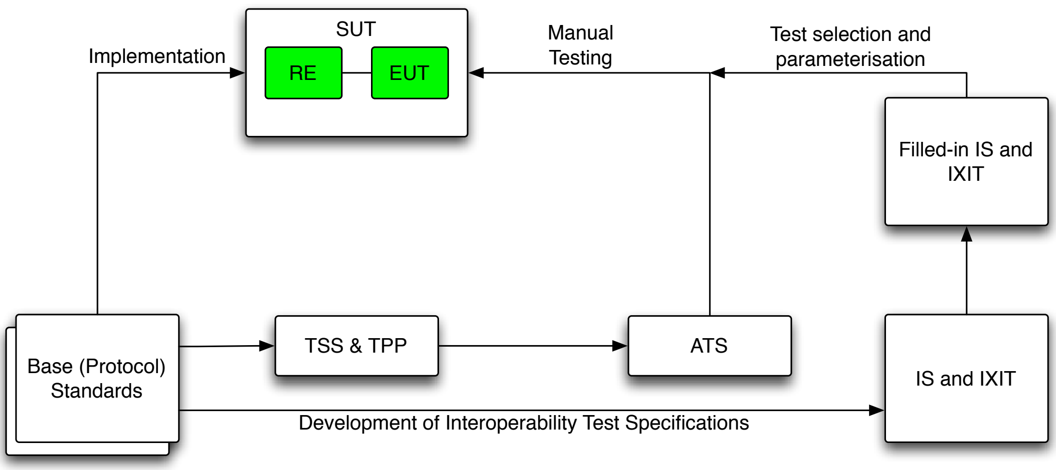

Figure 2.10: Interoperability Testing Process. Modified from [3].

2.3.3 Comparison of Conformance and Interoperability Testing

Conformance testing is mainly used for single components of a complete system such as testing a single protocol entity. Conformance testing is conducted under fixed constraints using a dedicated testing framework, i.e. a workbench. These types of test systems are, when based on standard interfaces (e.g. IP), low priced to develop. For the implementation are only a test programming language like TTCN-3 and a compiler needed. One strong point of conformance tests is that the correct behaviour of a SUT can be tested by stimulating it with invalid messages. This is an advantage over interoperability testing because aspects, like the robustness of a network element, can be verified as well.

In contrary to this, interoperability testing is concerned with complete network elements or a collection of network elements from different vendors respectively. Interoperability testing is conducted between network elements in a network end-to-end. Testing shows that certain functions are implemented but not how. Interoperability testing ensures the interworking between different network elements providing a function but not complying with the protocol requirements.

A summarised comparison of conformance and interoperability testing is illustrated in table 2.2. To sum up, these methods are diverse and can complement one another when using combined [25].

Table 2.2: Comparison of Conformance and Interoperability Testing

Criterion	***Conformance Testing***	***Interoperability Testing***
Test Setup	Test between a dedicated test system and protocol entity	Test between several network elements from different vendors end-to-end
Interoperability?	Guaranteed only with 100% test coverage (unrealistic) [26]	Only between tested network elements
Test Focus	Unit Testing - focused only on a single protocol instance	System Testing - wider focus than conformance testing: interaction between several protocol instances in a network element is tested (e.g. telephony: signalling and user data)
SUT	Protocol entity (Implementation under Test)	Equipment under Test (EUT), comprising of several interacting IUTs

Continues on next page

Criterion	*Conformance Testing*	*Interoperability Testing*
Test Types	Valid, invalid and timing behaviour can be tested. Through invalid tests robustness of SUT is tested as well.	Only valid behaviour can be tested
Test Execution	High degree of control (protocol messages can be designed as needed)	Only input through the devices' interfaces
Test Evaluation	High transparency due to powerful pattern matcher	Cause of errors can only be detected by analysing the trace (is done mostly manually)

2.4 Post-hoc Analysis

Conformance and interoperability testing are both conducted under controlled conditions in a lab. Not all influences of a network in operation can be emulated, such as timing constraints or interdependencies between the network elements. Therefore, a communication network has also be surveyed in production usage and all exchanged messages, so called *Protocol Data Units* (PDUs) recorded. Faulty behaviour, noticed and reported by the users of the network's terminal equipment, is analysed for the cause of the error.

The amount of data of all PDUs in a complex communication system is typically very large. Finding the cause of an error efficiently calls for an appropriate filtering method. The faulty behaviour has to be expressed in a formal way so that the erroneous message sequences can be found in the recorded data.

There are already solutions available which support a post-hoc analysis [27]. However, these solutions only record statistical data, such as the duration, participants and the status (e.g. successful or cancelled call) of a Voice over IP call. An error can be detected but the cause still has to be examined manually.

2.5 Methods for Ensuring Software Quality

Verification, validation, and testing are closely tied to software quality [28, 29]. There are many methods for the verification and validation of software systems such as black box and white box testing.

Beyond that, using process models representing the life cycle of a software – such as the Spiral Model – ties testing closely to the development process. This is achieved by including testing activities such as test development and test execution in the life cycle. But, the later in the life cycle of software an error is found, the higher is the cost of its correction [30].

Each testing method is concerned with the verification of certain aspects of the software (i.e. correctness, robustness, efficiency, functionality, and ease of use) [31]. Therefore, combining testing methods with different hypotheses is an important and popular statistical practice [32]. However, this approach is of a highly theoretical nature and there exists no practical guidelines on how to apply it.

A more practical solution is the selection of testing methods and map them to phases in the development process according to tested aspect of the software. This approach has – so far – been applied successfully in different areas of software development such as expert systems and simulation studies [33, 34].

In the telecommunication system's domain software is mainly concerned with the correct implementation of protocol specifications. A typical software implementing a protocol is a network element. Utilising protocol testing it has to be ensured that developed network elements can interoperate. There are several types of testing methods for software implementing protocols such as conformance and interoperability testing verifying different aspects of the network element (see section 2.3). Due to their different test aspects ETSI encourages using both of them [3]. However, until now there are no guidelines available how to combine these methods and if using these two methods are sufficient for ensuring the interoperability between network elements. Furthermore, an integration of protocol testing methods into a process model is highly desirable.

2.6 Conclusion

Next Generation Networks (NGNs) implement a platform for the easy implementation of access independent services. Typically, these services provide certain functions such as telephony or messaging to the end user. Services are composed by utilising one or more protocols (e.g. SIP, SDP, RTP for Voice over IP).

Ensuring the interoperability between network elements in NGNs implementing protocols and services is of utmost importance. Therefore, different testing methods like conformance and interoperability testing are available. However, these two methods do not consider all influences of network in operation. For this purpose a special method – so called post-hoc analysis – is needed, which efficiently finds the causes of errors of a network in operation.

In software development the use of models for the software development process and the utilisation of different testing methods increases the software's quality. There have been first efforts to further improve the quality of the software by systematically combining these two approaches.

The challenges for this report are:

- Creation of a post-hoc analysis method for efficiently finding the causes of errors of a network in operation considering:
  - Multiple interacting protocols
  - Extensibility for further protocol types and analysis functions
- Mapping of protocol testing methods to phases of a process development model

CHAPTER 3

Methods for Improving the Interoperability in Communication Networks

In this chapter different approaches on how to improve the interoperability between network elements are being discussed. Firstly, a testing method, the so called post-hoc analysis, for analysing a complete communication network, comprising of several interacting network elements, on interoperability is presented. Next, the three presented testing methods (i.e. conformance and interoperability testing, post-hoc analysis) are mapped to a phase in a process model to further improve the interoperability.

3.1 Post-hoc Analyses on Stored Protocol Data Units

As shown in section 2.4, page 18 there is no solution for the efficient detection of errors in protocol traces of communication systems in operation.

In this report the approach was taken to record the exchanged messages and then store them in a relational database. Using a database allows the usage of the *Standardised Query Language* (SQL). With SQL's algebra any set of messages in the database can be selected. With this, the encountered faulty behaviour can be reproduced with SQL. Additionally, the usage of a database offers - especially with an index - an optimised search.

Another aspect of improving the interoperability between network elements in a communication network can be achieved by implementing further conformance test cases based on the errors found during production usage (see figure 3.1).

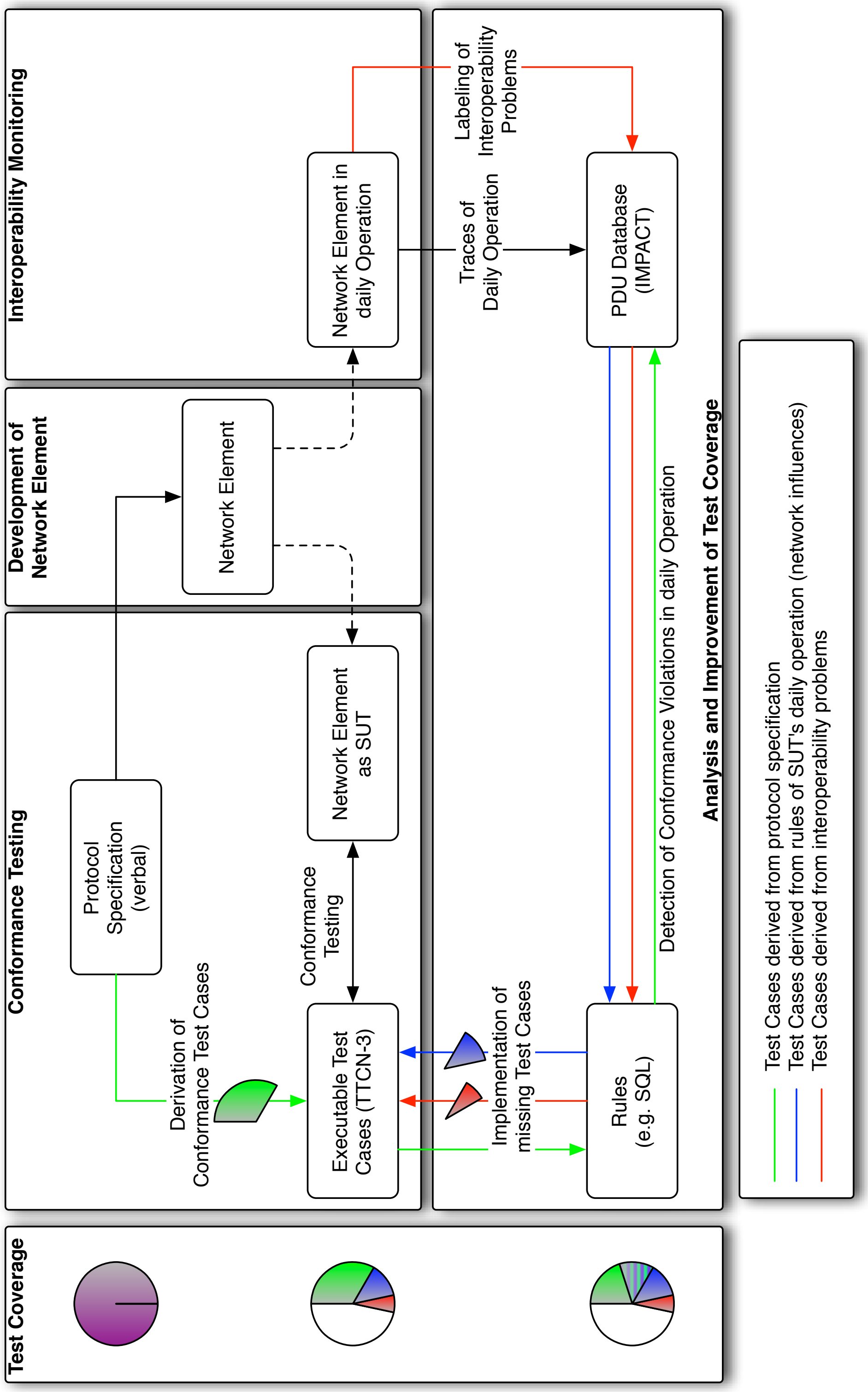

Figure 3.1: Analysis and Improvement of Test Coverage

3.2 Combining Software Development Process and Testing Methods

There are several testing methods, each of them focussing on a certain aspect. While conformance testing checks whether a protocol instance is implemented as described in its specification, interoperability testing verifies if a function can be used between network elements from different vendors successfully. The third testing methods considered in this thesis, the post-hoc analysis (see section 3.1), is used to find the cause of errors in production usage efficiently.

One way to improve the confidence of an applied testing method would be to increase the number of its test cases aiming for a complete test coverage. However, this increases the time of the execution of the set of test cases as well. A complete test coverage is unrealistic and its execution would not be feasible anymore.

Another way is to improve the confidence with which network elements interoperate with each other by combining several testing methods. In this case different aspects of an implementation can be evaluated while leaving the amount of time for the execution of the tests at an acceptable level.

For an effective combined use of the three presented testing methods the order of application is an important part. If applied in the right order, results of a test can be used to improve the test being used previously (see figure 3.1).

Therefore, in this thesis an approach is taken to assign each testing method a certain phase in a software development process. A software development process defines in structured steps how a software has to be designed, implemented and tested.

3.3 Conclusion

This report considers two different, but related, approaches in the testing process of network elements in complex communication systems. The first approach's objective is to provide a method for analysing erroneous PDUs in a communication system in operation efficiently. The second approach aims at improving the test coverage by combining the presented testing methods (conformance and interoperability testing, post-hoc analysis) and a software development process (V-Model).

CHAPTER 4

Post-hoc Analysis of Protocol Data Units

Many errors concerning conformance and interoperability are often only revealed in production use of a communication system. The system, outside of a dedicated test environment, is open to a bigger variety of faulty effects, such as when reaching its load limit.

A network element, being in production use, can be further checked with post-hoc analyses. With post-hoc analyses a network element is not stimulated, but its behaviour in daily operation is recorded and afterwards analysed.

At the Ostfalia University the tool IMPACT (IP-Based Multi Protocol Post-hoc Analyzer and Conformance Tester) is being developed and has been used so far for analysing SIP-ISDN-gateways. The network traffic, protocol data units (PDUs), of a network element is recorded and then stored in a database. Using a database gives the advantages for applying relational algebra for complex search and analysis functions. This is a major requirement for the evaluation of a large amount of PDUs. Insights gained during this research are implemented in the current version of IMPACT and can then be evaluated.

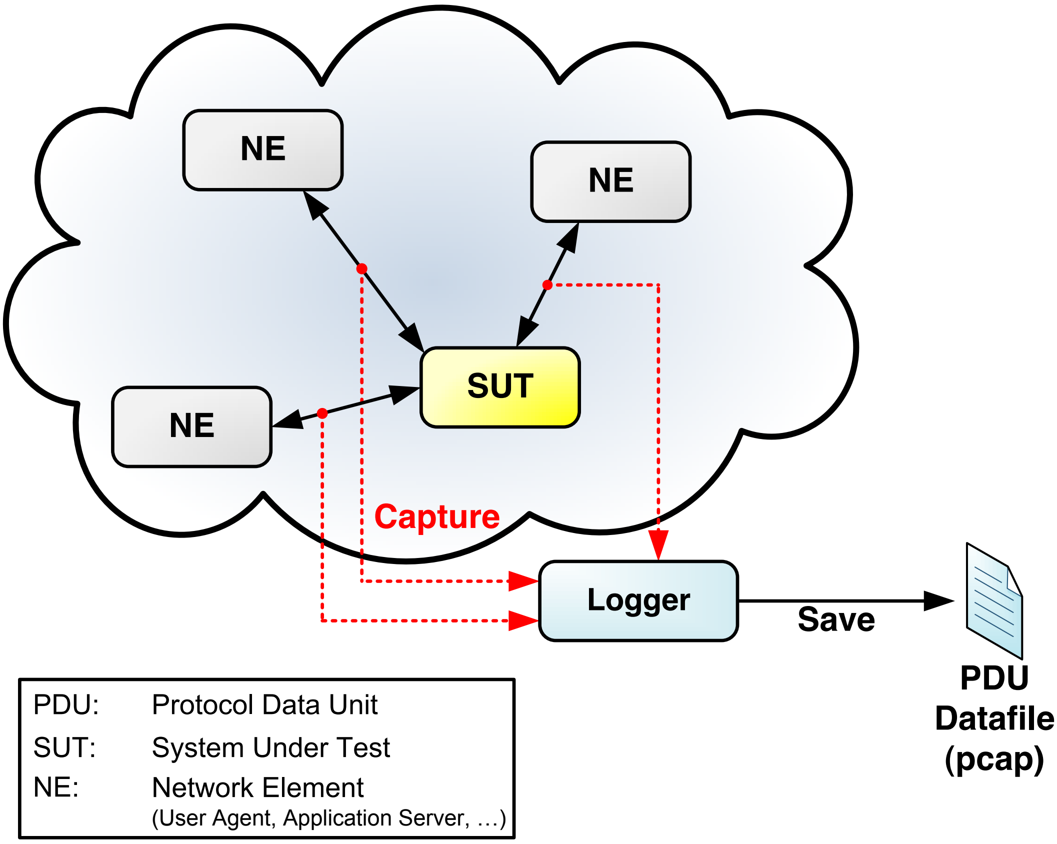

Figure 4.1: Network in Daily Operation

4.1 IMPACT Architecture

PDUs are recorded at an IP-based network interface between the system under test and its connected devices (see figure 4.1). The four main components of IMPACT are the database and its database management system (DBMS) for storing the recorded PDUs as well as the results of analyses, the parser for transferring recorded PDUs into the database, the GUI for control and the analysis functions written in SQL (see figure 4.2).

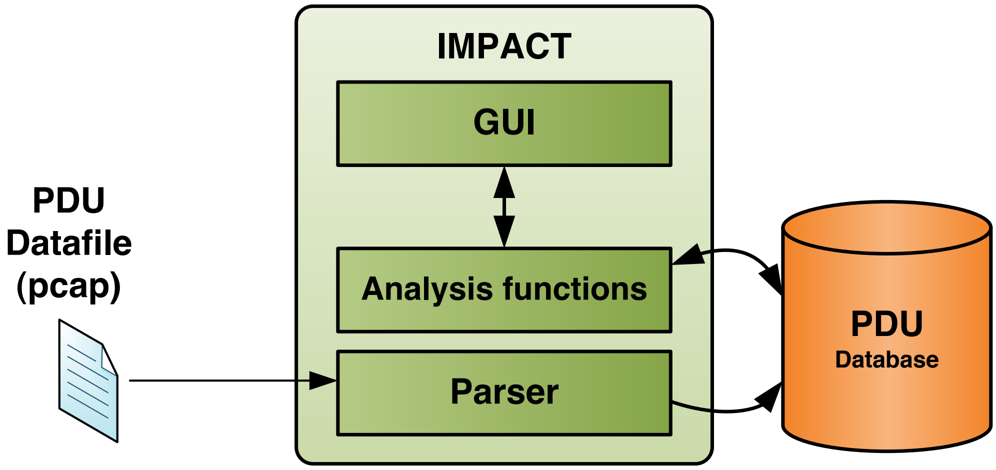

Figure 4.2: IMPACT Architecture

4.2 Recording and Storing of Protocol Data Units

Network traffic generated by the SUT in production use is recorded with a sniffer (i.e. Wireshark or TShark [35]), and PDUs are stored in binary files, i.e. packet capture (pcap). Using the GUI and utilising the Parser, PDUs can be transferred into the database, which is typically located on a server where they are stored permanently for future analyses.

The most recent version of IMPACT not only supports storing and analysing of signalling but also stores user data (e.g. media streams such as RTP). Analysing the interaction between signalling and user data is therefore possible as well.

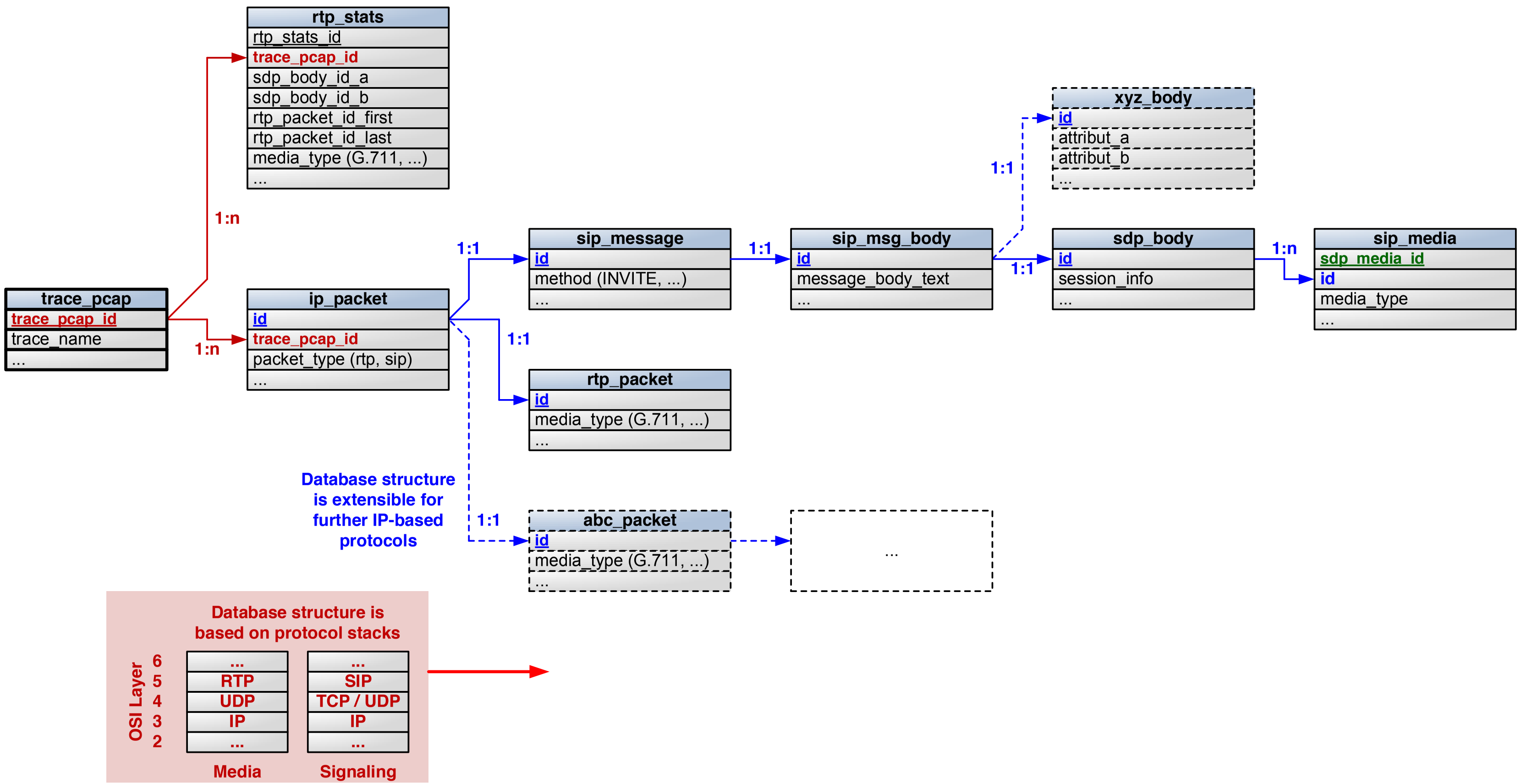

Figure 4.3: IMPACT's Database Structure. IMPACT's database is designed in a way that it represents an IP-based protocol stack.

IMPACT's database was designed in a way that it represents an IP-based protocol stack (see figure 4.3). The lowest layer in IMPACT is IP. On top of it are higher layer OSI protocols such as SIP or RTP. Using a database structure design according to the protocol stacks makes an extension of further protocols easily possible.

4.3 Analysis of Stored Protocol Data Units

Finding errors with IMPACT test cases have to be designed utilising SQL queries. Similar to TTCN-3 a SQL query formulates a stimulation part and a reaction part, which have to be found in the database.

Queries are formulated in a way that errors are detected explicitly. If the system's behaviour was faultlessly and conform during the time of recording, the result of the query is an empty relation.

The amount of PDUs, which are recorded of over a long time, can easily exceed several millions. Therefore, utilising a modern, indexed relational database is the only way to effectively detect faulty error caused by networks in production use.

4.4 Conformance Verification with IMPACT

One research objective is to find out whether conformance verifications to specific protocol specifications can be implemented in an analysis tools such as IMPACT. Therefore, both methods are compared.

The main difference between a conformance testing workbench and IMPACT is that conformance testing can actively stimulate network elements while IMPACT can only be used for passive observations. A message sent by a workbench stimulating the SUT has to be mapped to a row in IMPACT's database. All dialogues with this matching message are chosen. The possible valid reactions of the SUT defined with TTCN-3 are each mapped to an SQL query (figure 4.4) and combined with an OR-function (figure 4.5).

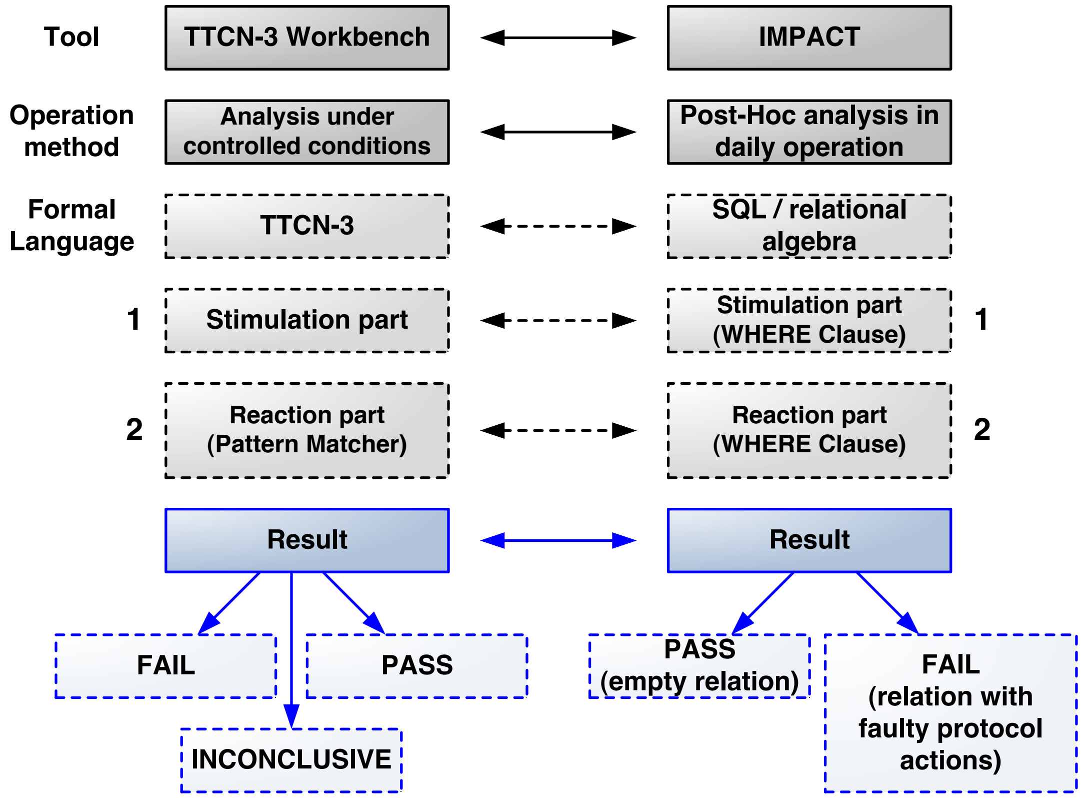

Figure 4.4: TTCN-3 Test Purpose Converted to a SQL Query

In general any combination of messages can be described with SQL. Valid, invalid and timing aspects can be looked up in the database. However, the meaning of searching for invalid messages – in conformance testing used to test the SUT in terms of robustness – is questionable because this behaviour is unlikely to happen in a communication system in operation.

```
-- SIP_CC_TE_CE_V_001
SELECT *
FROM SIP_Nachricht
WHERE Call_ID IN
(
    SELECT DISTINCT(CALL_ID)
    FROM SIP_Nachricht
    WHERE
        Method = "INVITE"  1
        AND
        (Call_ID, CSeq) NOT IN
        (
            SELECT X.CALL_ID, X.CSeq
            FROM SIP_Nachricht AS X, SIP_Nachricht AS Y
            WHERE
                X.Call_ID = Y.Call_ID
                AND
                X.ID < Y.ID
                AND
                X.CSeq = Y.CSeq
                AND
                X.Method = "INVITE"  1
                AND
                (
                    Y.Status_Line LIKE "%200 OK%"
                    OR                                          2
                    Y.Status_Line RLIKE '[1][0-9]{2,2}'
                )
        )
)
```

1: Stimulation part
2: Reaction part

Figure 4.5: TTCN-3 Test Purpose Converted to a SQL Query

4.5 Multi Protocol Interoperability Analysis

Another field of application for a post-hoc analysis tool is to check whether interaction between several, different protocol instances in a communication network work as expected. In IMPACT's database the protocol stacks for signalling as well as media are represented. For the analysis of VoIP calls the SIP, SDP and RTP are relevant. After a successful session setup IMPACT can check whether two RTP channels have been established and whether the correct codecs described in the SDP body of the SIP message are used.

4.6 Conclusion

In this chapter the concept for a post-hoc analysis tool is described with IMPACT as an example. Recording the network traffic in a communication network is important for finding the cause for errors in production use. IMPACT parses all PDUs and stores them in a database. Using SQL any message sequence can be described and detected in the typical very large PDUs.

The main focus of post-hoc analysis tools is to support the network operators by tracking down errors in production usage. Especially when problems, caused by changes to the network (i.e. new software, terminals of another vendor, firmware upgrade), occur, these can be easily detected by formulating SQL queries describing the erroneous behaviour. With this relevant PDUs can be narrowed down efficiently.

Additionally, a conformance verification by transferring TTCN-3 test purposes to SQL queries in IMPACT can increase the degree of confidence that a network not only interoperates but also does this according as defined in the protocol specifications. Finally, interoperability analyses can be performed with IMPACT examining the interaction of several protocols such as signalling and media streams. Conformance and interoperability analyses can both be conducted automatically.

CHAPTER 5

Systematic Combination of Controlled Stimulation And Post-hoc Test Methods

Our approach for increasing the confidence that a developed network elements interoperates with other network elements in a communication system is by combining the presented testing methods. The confidence that a communication systems works as expected shall be achieved by mapping each of these testing methods to a development phase in the V-Model.

In general, testing should be done as early as possible in the development process while modifications can be done easily. Errors which are detected later are typically more expensive to fix.

5.1 Mapping of Testing Methods to Phase in V-Model

In this section for each phase in the V-Model the best suitable test methods is selected. The result is presented in figure 5.1 and explained in following subsections.

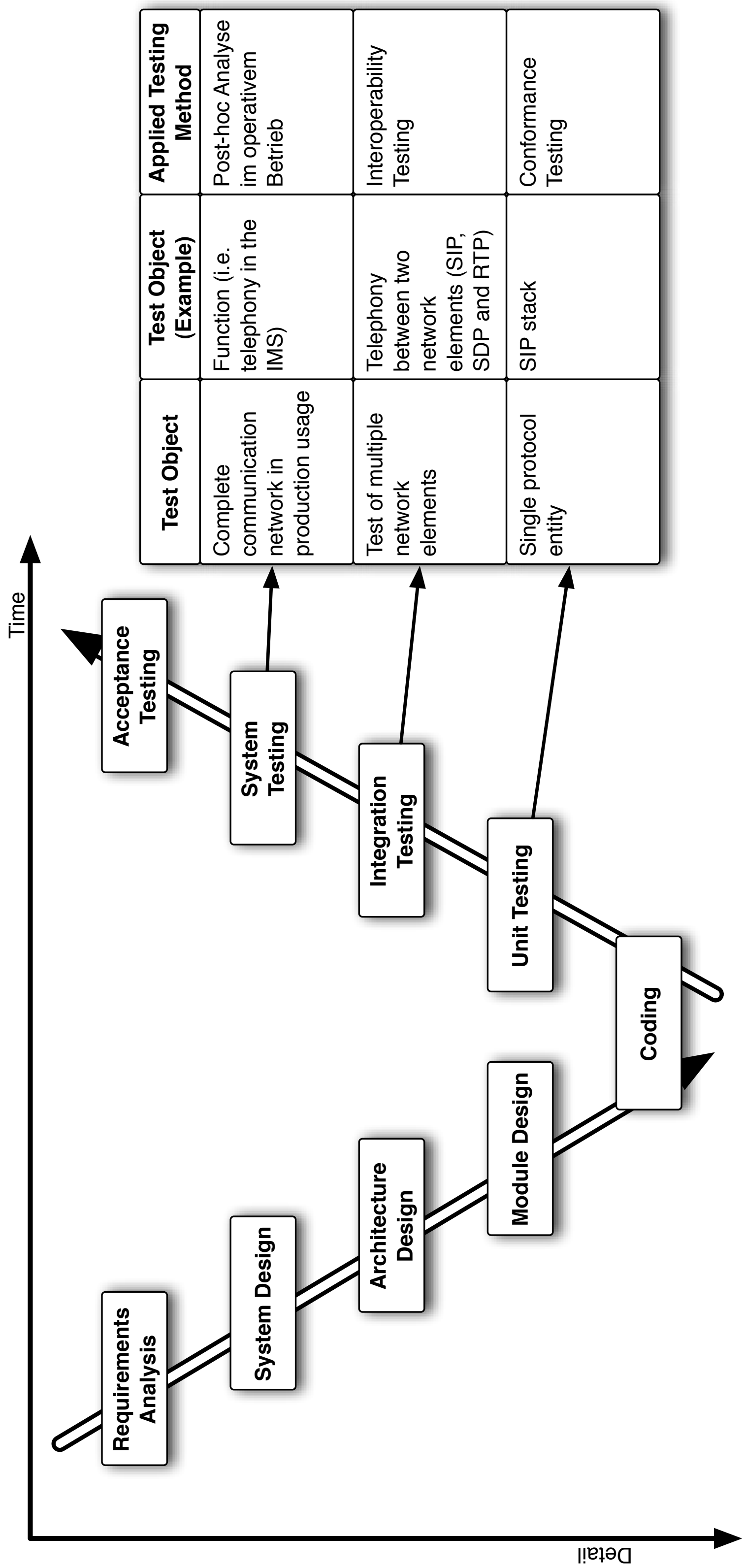

Figure 5.1: V-Modell with Testing Methods

5.1.1 Unit Testing - Conformance Testing

Unit Testing is the first phase in the V-Model and concerned with testing a single component. In terms of network elements in communication systems such a component would implement one single protocol. The protocol is defined by its respective protocol specification. Using conformance testing checks if the protocol specifications are implemented correctly.

Ideally, two network elements successfully passing a conformance test with a 100% test coverage are interworking correctly as well. However, this is unrealistic because the amount of test cases required would be great and their execution not feasible as already shown in the first report.

5.1.2 Integration Testing - Interoperability Testing

The next phase in the V-Model is the integration test. In this phase the single components of the previous phase are combined to create network elements providing new functions. Each of these network elements in this phase do not necessarily have to implement the same protocol specifications.

For this kind of setup interoperability testing is the best approach because its focus is not only centred on one protocol specification but on the interaction of several different ones.

5.1.3 System Testing - Post-hoc Analysis

Finally, in the third phase of the V-Model a complete network is setup consisting of several interacting network elements of the previous phase. The complete system has to be checked in terms of interoperability but also on reliability and robustness.

Analysing the behaviour demands the observation of several session at different points in the network. For an interpretation that the network works error free the exchanged messages have to be synchronised. Therefore, the best approach is to record all of them and store them in a database like it does IMPACT. After having done this post-hoc analyses are possible examining different aspects (i.e. conformance verification and interoperability analysis) of the network.

5.2 Conclusion

Combining testing methods and a process model increases the confidence that the network elements in a communication systems work according to their protocol specifications, functions (i.e. telephony) utilising multiple protocols are implemented as expected and the whole network is observed in production usage (i.e. under load limits).

Additionally, errors found during interoperability testing can be used to increase the test coverage of the conformance test suite. This can be done for example by formulating a new test purpose and implement it with TTCN-3. If errors in the communication system are revealed during the post-hoc analysis, their cause can be used for increasing the test coverage as well by implementing further conformance and interoperability test cases.

CHAPTER 6

Discussion

This chapter discusses how and to what extent the research objectives, that were defined in chapter 1, have been fulfilled. In the first section the methods and results for post-hoc analyses with IMPACT are analysed. After this the combination of a process model, the V-Model, and the presented testing methods is reviewed.

6.1 Post-hoc Analysis

Finding the cause for errors in a communication network is a tedious task. Pinpointing the source of the problem is typically linked with manually reading large log files or traces of protocol data units (PDUs), which consumes a lot of time.

In this thesis a concept was presented to store all exchanged PDUs in a communication network in a relation database and formulate analysis functions with SQL. The strong point in using SQL is that any sequence of messages can be efficiently selected in the PDU trace narrowing down the PDUs having to be analysed.

An example tool implementing these concepts is IMPACT (IP-base Multi Protocol Post-hoc Analyzer and Conformance Tester). Its database was designed in a way that it represents a typical OSI protocol stack. This gives IMPACT the advantage that it can be easily extended with any IP-based protocol and thus be used in other domains as well (such as smart metering, smart grid or in-vehicle applications).

Additionally, conformance verifications and interoperability analyses can be implemented with SQL. Using conformance verifications with IMPACT do not guarantee that all network elements work as described in their protocol specification but increases the confidence of a successful interoperating network.

6.2 Combination of Process Model and Testing Methods

Ensuring the interoperability between different, interacting network elements is an important goal. Testing is typically expensive. Test cases have to be designed and executed. The results have to be evaluated and the tested network elements improved. Often there are discussions between the network element's developers which testing method is best suited concerning effort and insight.

In this thesis an approach to develop a concept which interlinks a process model with different testing methods. Each testing method has its strong and weak points. One single testing method cannot (practically) ensure the interoperability of all network elements in a communication network. By combining testing methods with a process model the advantages are accumulated and their disadvantages minimised.

A complete test coverage can never be guaranteed but the presented approach increases the chance of interoperability. Additionally, causes of detected errors in one of the test phases can be used to increase the test coverage of a previous phase.

6.3 Conclusion

Conformance testing with a sufficient coverage gives a sound basis for ensuring the interoperability. However, it's expensive and time consuming. Interoperability testing means a combinatoric explosion of tests, the more vendors and operators a market has. Both methods are diverse and can complement one another when using combined [25].

The post-hoc analysis is a method to identify ultimate causes for interoperability problems in daily operation, which becomes ever more important. IMPACT is an efficient tool for this task and has been used so far in the telecommunication domain. It provides it solid basis for the demanding testing needs of new application areas such as smart grid in multi-vendor / multi-operator environments.

Based on the literature the combination of diverse testing methods is a popular approach [32]. In this report a concept was presented to combine test methods and process model improving the chance of interoperability between all network elements in a network. The V-Model was chosen and for each phase an appropriate test method was selected. This strategy provides the developer with designing and testing guidelines which improves not only the quality of the product but also increases the chance to detect errors earlier in the product's life cycle.

CHAPTER 7

Conclusion and Further Work

In this report testing methods for ensuring the interoperability between network elements in Next Generation Networks (NGNs) have been critically evaluated and compared. To this end conformance and interoperability testing were scrutinised. Although following the same goal of ensuring the interoperability between different network elements, these two methods are diverse and examine different aspects.

However, these two methods cannot ensure the interoperability. Both testing methods are typically conducted in a laboratory under controlled conditions. A network element, outside of a dedicated test environment, is open to a bigger variety of faulty effects such as when reaching its load limits. Finding errors and their causes in exchanged message sequences of all network elements in a communication system in production usage is an important aspect of ensuring the interoperability. This is typically done manually, takes a lot of time and is inefficient.

One objective of this thesis was to propose a method for the efficient detection of faulty message sequences. As an approach to this, the post-hoc analysis tool IMPACT (IP-Based Multi Protocol Post-hoc Analyzer and Conformance Tester) was being extended that it can be used in NGNs. IMPACT is based on a relational database and stores all Protocol Data Units (PDUs). Using SQL any sequence of messages containing errors can be efficiently detected in a large amount of PDUs in the database. Additionally, it is possible to transfer TTCN-3 test cases into SQL queries for a conformance verification. With this, distinguishing conformance from interoperability problems is possible.

Another objective was to provide a concept on when to use which testing method in the life cycle of a network element. In this report a strategy was presented to combine test methods and process model improving the chance of interoperability between all network elements in a communication network. This was achieved by mapping conformance and interoperability testing as well as the newly developed post-hoc analysis to the different testing phases in the V-Model.

The objectives of this report have been accomplished. One result is the post-hoc methodology which enables the identification of the ultimate causes for interoperability problems in daily operation, which becomes ever more important. IMPACT is an efficient tool for this task and has been used so far in the telecommunication domain. It provides it solid basis for the demanding testing needs of new application areas such as smart grid in multi-vendor / multi-operator environments. Moreover, storing of all PDUs in IMPACT's database enables a detailed conformance analysis and thus gives an essential advantage above currently available statistical network analyser. Another result is the improvement of the chance that two network elements interoperate by mapping the presented test methods to phases in the V-Model according to their specialisations. This approach helps the developer by providing well defined steps on which test method is the best suited at a certain stage in the development process. In this way not only the quality of the software product, but also the development process, is significantly improved.

For the future there are several interesting aspects to be examined. The number of SQL queries for detecting errors in Voice over IP calls should be extended. Additionally, IMPACT can be introduced for detecting conformance and interoperability problems in other domains such as smart grid. Moreover, the synchronisation of exchanged messages of different protocols at different interfaces of a network element (e.g. gateways) is a challenging task.

Bibliography

[1] P. Agrawal, J.-H. Yeh, J.-C. Chen, and T. Zhang, “IP Multimedia Subsystems in 3GPP and 3GPP2: Overview and Scalability Issues,” *IEEE Communications Magazine*, vol. 46, pp. 138–145, January 2008.

[2] C. J. Pavlovski, “Service Delivery Platforms in Practice,” *IEEE Communications Magazine*, vol. 45, pp. 114–121, March 2007.

[3] S. Moseley, S. Randall, and A. Wiles, “Experience within ETSI of the combined roles of conformance testing and interoperability testing,” in *The 3rd Conference on Standardization and Innovation in Information Technology*, October 2003, pp. 177–189.

[4] S. Bradner, “The Internet Standards Process - Revision 3,” Internet Engineering Task Force, RFC 2026, October 1996. [Online]. Available: http://www.ietf.org/

[5] M. Schmidt, A. Wilde, A. Schülke, and H. Costa, “IMS Interoperability and Conformance Aspects,” *IEEE Communications Magazine*, vol. 45, pp. 138–142, March 2007.

[6] R. J. Linn, “Conformance Evaluation Methodology and Protocol Testing,” *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 7, pp. 1143–1158, September 1989.

[7] S. Kang, “Relating interoperability testing with conformance testing,” in *Proc. Bridge to Global Integration. IEEE Global Telecommunications Conference GLOBECOM 98*, vol. 6, Nov. 8–12, 1998, pp. 3768–3773.

[8] Telecommunication Standardization Sector of ITU, “Next Generation Networks â Frameworks and functional architecture models - General overview of NGN,” International Telecommunication Union, ITU-T Recommendation Y.2001, December 2004.

[9] D. Wermser, C. Büch, M. Hans, and F. Kowalewski, "Adaptive Mobile Multimedia-Dienste für das zukünftige All-IP UMTS," *ITG Fachtagung "Ambient Intelligence"*, vol. 1, pp. 205–210, October 2004.

[10] Telecommunication Standardization Sector of ITU, "Data Networks and Open System Communications: Open Systems Interconnection - Service Definitions," International Telecommunication Union, ITU-T Recommendation X.213, October 2001.

[11] C. Ware, "The OSI network layer: Standards to cope with the real world," in *Proceedings of the IEEE*, vol. 71, no. 12, Dec. 1983, pp. 1384–1387.

[12] M. Poikselkä, G. Mayer, H. Khartabil, and A. Niemi, *The IMS: IP Multimedia Concepts and Services*. John Wiley & Sons Ltd, 2007.

[13] S. Hekmat, *Communication Networks*, PragSoft Corporation, Ed. PragSoft Corporation, 2005. [Online]. Available: www.pragsoft.com

[14] W. W. Royce, "Managing the Development of Large Software Systems," in *Proc. IEEE WESCON*, August 1970, pp. 328–338.

[15] F. A. Cioch, J. Brabbs, and S. Kanter, "Using the spiral model to assess, select and integrate software development tools," in *Proc. Third Symposium on Assessment of Quality Software Development Tools*, Jun. 7–9, 1994, pp. 14–28.

[16] B. Boehm, W. Brown, and R. Turner, "Spiral development of software-intensive systems of systems," in *Proc. 27th International Conference on Software Engineering ICSE 2005*, May 15–21, 2005, pp. 706–707.

[17] B. Boehm, "A spiral model of software development and enhancement," *Computer*, vol. 21, no. 5, pp. 61–72, May 1988.

[18] K. Forsberg and H. Mooz, "Visualizing Project Management," in *Proc. of the First Annual NCOSE Conference*, 1990.

[19] J. O. Clark, "System of Systems Engineering and Family of Systems Engineering from a standards, V-Model, and Dual-V Model perspective," in *Proc. 3rd Annual IEEE Systems Conference*, Mar. 23–26, 2009, pp. 381–387.

[20] C. Willcock, T. Deiß, S. Tobies, S. Keil, F. Engler, and S. Schulz, *An Introduction to TTCN-3*. John Wiley & Sons Ltd, 2005.

[21] European Telecommunications Standards Institute, "Methods for Testing and Specification (MTS); Protocol and profile conformance testing specifications; Standard-

ization methodology," European Telecommunications Standards Institute, ETS 300 406, April 1995.

[22] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," Internet Engineering Task Force, Tech. Rep., June 2002. [Online]. Available: http://www.ietf.org/

[23] European Telecommunications Standards Institute, "Methods for Testing and Specification (MTS); Conformance Test Specification for SIP (IETF RFC 3261); Part 2: Test Suite Structure and Test Purposes (TSS&TP)," European Telecommunications Standards Institute, TS 102 027-2, July 2006.

[24] ——, "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; Interoperability test methods and approaches; Part 1: Generic approach to interoperability testing," European Telecommunications Standards Institute, TS 102 237-1, December 2003.

[25] H. Van der Heer and A. Wiles, "Achieving Technical Interoperability - the ETSI Approach," European Telecommunications Standards Institute, ETSI White Paper 3, April 2008.

[26] N. Griffeth, R. Hao, D. Lee, and R. K. Sinha, "Interoperability Testing of VoIP Systems," in *IEEE GLOBECOM, Global Telecommunications Conference, 2000*, vol. 3, August 2000, pp. 1565–1570.

[27] D. Bao, D. L. Carni, L. De Vito, and L. Tomaciello, "Session Initiation Protocol Automatic Debugger," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 6, pp. 1869–1877, June 2009.

[28] W. R. Adrion, M. A. Branstad, and J. C. Cherniavsky, "Validation, Verification, and Testing of Computer Software," *Computing Surveys*, vol. 14, no. 2, pp. 159–92, 1982.

[29] B. W. Boehm, "Verifying and validating software requirements and design specification," *IEEE Software*, vol. 1, no. 1, pp. 75–88, January 1984.

[30] ——, "Seven basic principles of software engineering," *Journal of Systems and Software*, vol. 3, pp. 3–24, 1983.

[31] B. W. Boehm, J. R. Brown, and M. Lipow, "Quantitative evaluation of software quality," in *ICSE '76: Proceedings of the 2nd international conference on Software*

engineering. Los Alamitos, CA, USA: IEEE Computer Society Press, 1976, pp. 592–605.

[32] R. H. Berk and A. Cohen, “Asymptotically Optimal Methods of Combining Tests,” *Journal of the American Statistical Association*, vol. 74, no. 368, pp. 812–814, December 1979.

[33] S. Lee and R. M. O’Keefe, “Developing a Strategy for Expert System Verification and Validation,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 4, pp. 643–655, April 1994.

[34] O. Balci, “Validation, Verification, and Testing Techniques Throughout the Life Cycle of a Simulation Study,” in *Proceedings of the 26th Conference on Winter Simulation*, 1994, pp. 215–220.

[35] G. Combs, “Wireshark.” [Online]. Available: http://www.wireshark.org/

List of Figures

List of Tables

Glossary

Term	***Definition***
Conformance Test Suite	A complete set of test cases, possibly combined into nested test groups, that is needed to perform dynamic conformance testing for one or more OSI protocols.
Conformance Testing	Testing the extent to which an IUT is a conforming implementation.
Conformance Test Specification	One or more specifications that contain a standardised ATS, together with its related TSS and TP, partial IXIT proforma, and TMP specification, if any.
Executable Test Case	A realisation of an abstract test case.
Executable Test Suite (ETS)	A test suite composed of executable test cases.
Fail (verdict)	A test verdict given when the observed test outcome either demonstrates non-conformance with respect to (at least one of) the conformance requirement(s) on which the test purpose of the test case is focused, or contains at least one invalid test event, with respect to the relevant specification(s).
Implementation Conformance Statement (ICS)	A statement made by the supplier of an implementation or system claimed to conform to a given specification, stating which capabilities have been implemented. The ICS can take several forms: protocol ICS, profile ICS, profile specific ICS, and information object ICS.
Implementation Conformance Statement (ICS) Proforma	A document, in the form of a questionnaire, which when completed for an implementation or system becomes an ICS.
Implementation Extra Information for Testing (IXIT)	A statement made by a supplier or implementer of an IUT which contains or references all of the information (in addition to that given in the ICS) related to the IUT and its testing environment, which will enable the test laboratory to run an appropriate test suite against the IUT. An IXIT can take several forms: protocol IXIT, profile IXIT, profile specific IXIT, and information object IXIT, TMP implementation statement.

Continues on next page

Term	***Definition***
Implementation Extra Information for Testing (IXIT) Proforma	A document, in the form of a questionnaire, which when completed for an IUT or SUT becomes an IXIT.
Implementation Under Test (IUT)	An implementation of one or more OSI protocols in an adjacent user/provider relationship, being that part of a real open system which is to be studied by testing.
Inconclusive (verdict)	A test verdict given when the observed test outcome is such that neither a pass nor a fail verdict can be given.
Initial Testing State	The testing state in which a test body starts.
Inopportune Test Event	A test event which occurs when not allowed to do so by the relevant specification(s) to which conformance is being tested.
Invalid Test Event	A test event that violates at least one conformance requirement of the relevant specification(s) to which conformance is being tested.
Multi-protocol Testing	Testing of more than one protocol within the IUT by means of test cases which have test purposes which cover conformance requirements that relate to more than one protocol.
Multi-specification Dependency	A conformance requirement in one specification which specifies a requirement upon the support of another specification within a conforming system.
Pass (verdict)	A test verdict given when the observed test outcome gives evidence of conformance to the conformance requirement(s) on which the test purpose of the test case is focused, and when no invalid test event has been detected.
System Under Test (SUT)	The real open system in which the IUT resides.
Test Case	An abstract or executable test case.
Test Case Error	The term used to describe the result of execution of a test case when an error is detected in the test case itself.
Test Event	An indivisible unit of test specification at the level of abstraction of the specification (e.g. sending or receiving a single PDU).
Test Purpose	A prose description of a well defined objective of testing, focusing on a single conformance requirement or a set of related conformance requirements as specified in the appropriate OSI specification (e.g. verifying the support of a specific value of a specific parameter).
Test System	The real system which includes the realisation of the Lower Tester.
Test Verdict	A statement of pass, fail or inconclusive, as specified in an abstract test case, concerning conformance of an IUT with respect to that test case when it is executed.

Continues on next page

Term	***Definition***
Valid Test Event	A test event which is allowed by the protocol specification, being both syntactically and semantically correct, and occurring when allowed to do so by the protocol specification.

List of Acronyms

Acronym	*Meaning*
3GPP	Third Generation Partnership Project
ETSI	European Telecommunications Standards Institute
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IMPACT	IP-Based Multi Protocol Post-hoc Analyzer and Conformance Tester
IMS	IP Multimedia Subsystem
ISO	International Organization for Standardization
ITU	International Organization for Standardization
IUT	Implementation under Test
MTC	Main Test Component
NGN	Next Generation Network
PDU	Protocol Data Unit
PTC	Parallel Test Component
QoS	Quality of Service
RFC	Request for Comments
RTP	Real-Time Transport Protocol
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SQL	Standardised Query Language
SUT	System under Test
TP	Test Purpose
TS	Test Suite

Continues on next page

Acronym	***Meaning***
TSS	Test Suite Structure
TTCN-3	Testing and Test Control Notation Version 3
VoIP	Voice over IP