TPLan- A Notation for Expressing Test Purposes

Stephan Schulz, Anthony Wiles

European Telecommunications
Standards Institute (ETSI)
650 Route de Lucioles
F-06921 Sophia Antipolis Cedex
{Stephan.Schulz, Anthony.Wiles}@etsi.org

Steve Randall

PQM Consultants
4 The Myrtles
Tutshill, Chepstow
U.K.
Steve.Randall@pmqconsultants.com

Abstract. To this day test purposes are predominately defined in practice using natural language. This paper describes a more formal approach based on a notation which has been recently developed and standardized at the European Telecommunications Standards Institute (ETSI) called TPLan. We introduce here the motivation and main concepts behind this new notation, and share our experiences gathered from its application in the development of standardized test specifications. We also discuss how TPLan can help to make test development as a whole more efficient – especially in the context of suite based test specification.

1 Introduction

TPLan has been developed and standardized [1] by the ETSI Technical Committee Methods for Testing and Specification (TC-MTS). Members of this group include leading testing experts from industry and academia and it receives support from ETSI's own Protocol and Testing Competence Centre. For more than a decade MTS has been involved in the design of languages, methodologies, frameworks, and guidelines [2,3,4] to help rapporteurs to increase quality and effectiveness of their specifications. The Test Purpose notation, TPLan, was conceived when investigating approaches to improving efficiency in the development of test specifications based on patterns [5]. Here, it was realized that patterns could or should be identified much earlier than at the time of test case writing, i.e., when identifying test purposes.

Much of the current research on test specification development has focused on the development and use of either suite-based [2] or model-based [6] testing technologies. Test purposes have been anchored as a concept in conformance testing methodology [7] for a long time but, as such, have received little attention in the testing research community. Formal approaches to test purpose specification [9,10,11,12] have been proposed but these have yet to be deployed successfully in industry. Graphical approaches based on Message Sequence Charts (MSC) [13, 14] for specifying test purposes have had only limited success – arguably due to their limitation in expressing behaviour only in terms of interactions. In our experience natural language still dominates the specification of test purposes.

It was our intention with TPLan to make test purpose specification more structured but not completely formal. Evidence of this approach can be found in the notation where many of the base keywords have been selected from preferred writing styles used in ETSI's test purpose specifications. Another design criterion was to keep the core notation as independent as possible from any specific application area and testing technology while making it easily extensible. This opens TPLan to a wide range of applications from, for example, telecommunication to civil engineering. It also allows it to be used in conjunction with both suite-based and model-based testing.

After an introduction to test purposes and how they fit into test specification development we will discuss their specification with our notation in section 3 of this paper in more detail. TPLan has already been used within ETSI to specify more than a thousand test purposes in the context of test development for the Internet Protocol version 6 (IPv6) [15] and digital Public Mobile Radio (dPMR) [16]. Section 4 presents first experiences from these projects which have shown that TPLan can help to enforce uniformity of test purpose specifications and to identify inconsistencies in standard documents at an early stage before costly test case specification and validation. We believe, however, that this notation may have even more potential in the test specification process by reducing development times and increasing productivity. Section 5 proposes some ideas for more sophisticated tools that may achieve such additional gains.

2 About Test Purposes

As with any other development activity, better test specifications can be produced when a structured approach is followed. For more than 15 years ETSI has applied the methodology prescribed by [7,8] where the development of a complete test specification is broken down into five discrete steps as shown in Table 1. These steps can be understood as different levels of abstraction that bridge the large intellectual gap between a base specification and the final executable test suite. They are not only an essential framework to the test engineer but also enable a common understanding of the complete test suite between different target audiences.

Test Specification Step	Means of Specification	Question answered
Requirement (RQ) identification	Text, Tables	Which requirements are to be tested?
Test Purpose (TP) specification	Text, Tables, **TPLan**	What is to be tested?
Test Description (TD)	Text, Tables, MSCs, etc	How is it to be tested? (informally)
Test Case (TC) specification	TTCN-3, C, Java, Perl, Python, MSC, etc	How is it to be tested? (executable)
Test validation	-	Is test implemented correctly?

Table 1. Steps in test specification development

Test purposes are derived from the requirements stated in one or more base specifications that define the implementation. This direct relationship to the requirements makes it possible to make an early assessment of test coverage of the specification and to determine the inter-dependencies between different requirements. Each test purpose usually focuses on one specific requirement. Within ETSI, these base specifications are most often protocol standards.

Test purposes provide an essential abstraction of a test that specifies what is to be tested without going into the details of how a test is to be implemented. Test purposes are not test steps; they specify pass verdict criteria. Test purposes are written using the language and terminology of the base specification(s) and are independent of any particular programming language, test system or platform on which corresponding tests might eventually be executed. They need to be developed, discussed and stabilized prior to any test case specification.

Test purpose specification results in a rigid assessment of the requirements with which they are associated and can identify problems in base specifications long before any test is ever implemented or executed against an Implementation Under Test (IUT). Not all requirements will lead to test purposes due to the limitations imposed by the chosen type of testing, e.g., conformance or interoperability testing.

Test purposes serve an important role as a basic documentation tool. They do not only bridge the gap between original requirements and test case specification but also between technology experts (who are not necessarily test engineers), managers, and the test engineers. At ETSI this aspect is very important since test specifications are often reviewed and approved by standards working groups. These groups need to understand the requirements which are being tested without having to read detailed test case specifications. In addition, ETSI is an environment where test purposes and corresponding test case specifications are developed for a wide variety of technologies in a distributed (and multi-cultural) environment. Such an environment clearly has a need for a consistent and uniform approach to test purpose specification, i.e., a notation which provides a common and recognisable level of understanding.

3 Test purpose specification with TPLan

TPLan has been designed to make test purpose specification more formal without inhibiting the expressive power of prose. The intent was to enable a consistent and structured representation of test purposes across a wide range of application domains and cultural backgrounds while retaining the informal "look and feel" of a natural language. It is for this reason that the core TPLan syntax and semantics have been kept small and left open. Of course this flexibility or “freedom of expression” inevitably results in weaker semantics and limits the checks that a tool can perform purely on the basis of the TPLan definition itself.

3.1 Test purpose structure

A TPLan test purpose comprises two segments as shown in Figure 1: a header and a body. The header provides a unique identifier for the test purpose and, optionally, references to other useful information for the understanding of the test purpose. These can include the requirement(s) covered by the test purpose, the type of test purpose, dependencies with other test purposes, and the tested role of the IUT.

The body of a test purpose specifies the specific initial IUT condition required for the test purpose to be valid and critical verdict criteria for a test - in the form of a stimulus and response - to ensure that the requirement(s) are met. The structuring of the test purpose body into the "`with`", "`when`" and "`then`" clauses clearly shows the roots of this notation in black box testing. A test purpose body is usually written from the perspective of the IUT, i.e., pre-conditions refer to the required initial state of the IUT, etc.

The keywords "`when`" and "`then`" should not be misunderstood to require a complete specification of accurate test sequence(s). Stimulus and response in a test purpose should focus and isolate only the directly relevant parts of information needed to assess if a requirement is indeed fulfilled by an IUT, for example, message types and critical information element values. Again, the level of information content and language used should be identical to the one of the requirement definition in the base specification.

```
-- test purpose header
TP id: <string>
< other test purpose headers (optional) >

-- test purpose body
with { <pre-conditions> }    -- optional clause

ensure that {

  when { <stimulus> }

  then { <response> }

}
```

Fig. 1. Basic structure of a TPLan test purpose specification

3.2 Fundamental building blocks

The initial conditions, the stimulus and the response in a test purpose body are constructed using the concepts which are listed in Table 2.

Concept	Definition
Entity	A physical or logical actor which applies a stimulus or receives response, and vice versa.
Event	The measurable basis of a stimulus or response which may be parameterized with Values

Value	An abstract identifier representing either
	– a literal constant;
	– a numeric constant;
	– a field or other container

Unit	A concrete qualifier to a number which helps to indicate the relative size or quantity of the number.
Condition	An abstract expression of the status or state of the entity or entities under test.
Word	Any other natural language element useful for the specification of the test purpose body, for example, – an action – an article, preposition, adjective, adverb, etc

Table 2. Key TPLan concepts

Instances of these concepts can be created using quoted strings containing free form text. Some instances for entities and words such as "`IUT`", "`sends`" or "`containing`" have been pre-defined as keywords in the notation. An example of a basic TPLan test purpose is shown in Figure 2.

```
TP id     : CW_U01_002
Summary   : 'A busy user with information channel control
             but no B-Channel responds to an incoming
             SETUP'
RQ ref    : Section 9.5.1
IUT role  : user
with {  IUT in 'an information channel control state'
        and 'no B-Channel free'
     }
ensure that {
  when { the IUT receives 'a valid and compatible SETUP'
         from the TESTER
           containing 'a channel identification IE'
           indicating 'no B-channel available' }
  then { the IUT sends 'ALERTING' to the TESTER }
}
```

Fig. 2. A complete example of a test purpose

The drawback of quoted strings is, however, that it is impossible to associate much meaning with them. It is also not possible to check whether quoted strings specify instances of these concepts in the correct order; for example, that `ALERTING` in Figure 2 is really an event.

3.3 User defined extensions

TPLan allows users to extend or customize its vocabulary based on the concepts introduced in the previous section with keywords which are relevant to their own specific application domain. This concept makes TPLan much more powerful than other forms of test purpose specification. Although the notation does not support an explicit definition of the semantics associated with a word or phrase, such semantics can often be implied from application domain within which TPLan is being used.

As an example, assume that we define a new word "`accepts`". When we use this new word in a TPLan "`when`" clause, e.g., "`when { the IUT accepts 'this message' }`", then TPLan itself does not define or restrict what "`accepts`" actually means or how such acceptance is measured in an eventual test case specification. The word "`accepts`" could mean any of the following actions:

- the IUT displays a message to the user.
- no error is displayed to the user.
- the IUT will continue interacting normally.
- the IUT does nothing that is externally observable

However, the meaning of "`accepts`" is likely to be obvious to technology experts as well as test engineers familiar with the domain or technology. As a result, this word is a valid abstraction of either one or possibly more interactions with the user or internal or external entities.

```
xref CW_U { ETS_300_058_1 } -- ETSI standard reference

def condition information_channel_control_state

def event     SETUP { Channel_identification_IE }
def event     ALERTING

def value     no_B_channel_available

TP id     : CW_U01_002
Summary   : 'A busy user with information channel
             control but no B-Channel responds to an
             incoming SETUP'
RQ ref    : Section 9.5.1
IUT role  : user

with {  IUT in an information_channel_control_state
        and 'no B-Channel free'
     }

ensure that {

  when { the IUT receives a valid and compatible SETUP
         from the TESTER
           containing a Channel_identification_IE
           indicating no_B_channel_available }

  then { the IUT sends ALERTING to the TESTER }

}
```

Fig. 3. A complete example of a test purpose with user definitions

Users have to declare specific instances of the main concepts shown previously in Table 2 when they use them in the test purpose definition. Figure 3 illustrates an example test purpose from the telecommunication domain written for conformance testing. The user has included a cross-reference to identify the ETSI standard ETS 300 058-1 as the base specification and then defined one initial condition and two events representing the different message types.

The definition of the SETUP event shows one parameter. That does not mean that in practice the message that this event represents only has one parameter. It means that only this parameter is significant in determining whether the IUT fulfills the referenced requirement. In test purposes, events are abstract representations of exchanged or observed information; they are not complete message instances. Similarly the user defined value in the example is an abstract representation of a concrete value.

Note also that within TPLan, user defined conditions, events and values are expressed as identifiers, i.e., they must not contain spaces. In our example we have chosen underscores to preserve a feel of natural language to the identifiers but this is only our naming convention. Finally, notice that one initial condition has been specified for the sake of this example using a quoted string. Quoted strings can still be useful in cases where, for example, a pre-condition is very complex.

By means of a simple notation, the user is also able to restrict the syntactical context in which user-defined words can be used. Within a context definition statement, any word prefixed with a tilde character (~) may only be used in that context, any word surrounded by square brackets is considered optional and any unencumbered word can be used in any other syntactical context. As an example, the following definitions can be made:

```
def word requested
def context is [not] ~requested to
```

Here, the words "`is`", "`not`" and "`to`" are included in the predefined TPLan vocabulary. The "`context`" statement constrains the user-defined keyword "`requested`" so that it is only syntactically correct in the contexts "`is requested to`" and "`is not requested to`".

3.3 Arrangement of test purpose definitions

In most cases and for a variety of reasons test purpose specifications need a logical structuring. To assist users in such structuring TPLan offers two complementary mechanisms which are grouping and inclusion.

Test purposes can be arranged into logical, hierarchical groups by using the "`Group`" and "`End Group`" statements as shown in Figure 4. These groups as well as individual test purposes can also be collected together into a single specification referred to as a Test Suite Structure (TSS) which also contains a header of its own.

In those cases where a number of test purpose writers are involved in a project, it will be necessary to maintain a single source of vocabulary extensions. For this purpose, the notation allows a TSS to include other TPLan files by means of a **`#include`** statement as shown in Figure 5. This mechanism uses a simple replacement method so that the content of the identified file is inserted into the file in place of the **`#include`** statement. Additionally, the inclusion mechanism can be used to construct a complete TSS from separate group files developed by multiple test purpose writers.

```
TSS       : COR_IOP -- identifier for all test purposes
Title     : 'RFC2460 IPv6 Core Specification'
Version   : 1.0.1
Date      : 05.10.2006
Author    : 'Steve Randall (ETSI TC-MTS)'

Group 1 'Initialization functions'

Group 1.1 'System startup'

Group 1.1.1 'Memory check'
…
<test purpose definitions>
…
End Group 1.1.1

Group 1.1.2 'Media check'
…
<test purpose definitions>
…
End Group 1.1.2

End Group 1.1

End Group 1

…
```

Fig. 4. Example test purpose structuring using TSS header and grouping

```
TSS       : COR_IOP -- identifier for all test purposes
Title     : 'RFC2460 IPv6 Core Specification'
Version   : 1.0.1
Date      : 05.10.2006
Author    : 'Steve Randall (ETSI TC-MTS)'

#include c:\include\SIUnitDefs.tplan
#include c:\include\IOPDefs.tplan
#include c:\include\IPv6Defs.tplan

#include c:\include\IPv6Group1.tplan -- Initialization

#include c:\include\IPv6Group2.tplan -- Outgoing call

#include c:\include\IPv6Group3.tplan -- Incoming call
```

Fig. 5. TPLan specification constructed from `#include` statements

4 First experiences

TPLan has been used by ETSI for the specification of test purposes in its IPv6 and dPMR test development projects. In both cases test purposes have been specified for two types of testing, conformance and formalized interoperability testing [17].

Two examples, a dPMR conformance and an IPv6 interoperability test purpose, are shown in Figures 6 and 7. These examples illustrate how TPLan vocabulary can be customized for these specific application domains and adapted to different types of testing. Note that required TPLan user definitions have been omitted from the figures. Also, almost all test purpose header lines are optional. The ones chosen in these examples provide further information about a test purpose summary, the type of test purpose, a reference to the catalogued requirement that the test purpose pertains to, the role or type of equipment being the subject of test, as well as a reference to the test architecture or configuration in which the IUT or Equipment Under Test (EUT) is embedded.

```
TP id      : TP_PMR_0406_01
summary    : 'Header frame acknowledges connect request'
TP type    : conformance
RQ ref     : RQ_001_0406
IUT Role   : CSF    -- Configured Service Function (CSF)
config ref: CF_dPMR_CSF_01 -- CSF Implementation Under
                           -- Test (IUT) and TESTER

with   { IUT in standby }

ensure that {
  when { IUT receives a Connection_Request }
  then { IUT sends an Acknowledgement_Frame }
}
```

Fig. 6. Example dPMR conformance test purpose

```
TP id      : TP_COR_8231_01
summary    : 'EUT uses at least two of the connected
              routers as its default routers '
TP type    : interoperability
RQ ref     : RQ_COR_8231
EUT role   : Host, Router -- = either Host or Router
config ref: CF_033_I -- 2 Routers and 1 Node as
                     -- Qualified Equipment (QE1/2/3) +
                     -- Equipment Under Test (EUT)
                     -- connected via 2 links
TD ref     : TD_COR_8231_01

with { QE1 having 1 unique unicast_address on each link

   and QE2 having 1 unique unicast_address on each link

   and EUT and QE3 able to communicate

   and QE1 'having disabled one of its interfaces' }
```

```
ensure that {
  when { (    QE1 disables 1 interface
           or QE2 disables 1 interface )
          and EUT is requested to send a packet to QE3 }
   then {     EUT sends the packet to QE3 }
}
```

Fig. 7. Example Ipv6 core interoperability test purpose

Our experiences with the first prototype version of TPLan (which allowed the use of a non extensible pre-defined set of keywords and quoted strings) were that writers felt limited in their ability to fully express test purposes. The language used by the standard document differed too much from the language that could be constructed from pre-defined TPLan keywords. Consequently, writers frequently requested new keywords to be added to the notation and made heavy use of quoted strings in test purpose specification. That, in turn, reduced the ability of project managers to ensure the quality and consistency of test purposes.

The introduction of user defined extensions to TPLan radically changed this situation. The ability to define a domain specific vocabulary not only gave writers more freedom in specifying test purposes but also made it easier to detect the misuse or misspelling of significant words. We noticed that, independent of the project type, the user defined vocabulary initially grows quite rapidly during the specification of first test purposes. After that, however, the need for new definitions levels off quickly. The test purpose writers also found it useful that user defined keywords could be explained or clarified with comments at one central place, i.e., their definition.

The extra effort spent in structured writing helped to reveal many problems or inconsistencies in the base specification prior to any test case specification or execution. This property of the notation became especially apparent during dPMR test purpose specification where writers were experts in the technology but novices in testing. Automated syntax checking with a simple parser [18] gave a first level of assurance on test purpose quality. A manual check of test purposes was nevertheless still required to assure their correctness. For a proficient English speaker it was easy to identify incorrect or badly written test purposes as these were not minor grammatical or spelling errors, the test purposes just obviously read incorrectly. It is not clear at this point if an improved syntax checker or further tool support could eliminate the need for this second grammar check.

5 Improving of test specification efficiency

In this section we want to show how TPLan offers a foundation to build on. Remember that it has only recently been standardized and is still in its infancy. So far there is much interest in it but only limited tool support. Based on our early experience with TPLan we believe that additional tool support could help to further improve the speed and quality of test purpose as well as test specification development as a whole.

Sophisticated editor support is probably one of the more important issues. Context sensitive editors could assist, extend and manage TPLan user definitions and provide features such as syntax highlighting, keyword completion and other forms of vocabulary management. This kind of tool would help users to avoid writing incorrect TPLan test purposes to begin with. More advanced parsers which go beyond simple syntax checks are needed to help pinpoint incorrect test purposes early on in the specification process. It may also be possible to extend the analysis of test purposes by incorporating some of the English grammar checking technologies used by modern word processing software.

Test purposes that have been checked for correctness can be used as input to other forms of processing. One of these is the identification of recurring patterns in preconditions and the interactions between entities. Such patterns can be used in a variety of ways:

1. the identification of potentially reusable segments of test case specifications derived from the test purposes;
2. an assessment of test purpose variation;
3. an estimation of the possible complexity of the eventual test case specification;
4. an estimation of the effort likely to be required for implementing the eventual test specifications.

When used with suite-based test technologies TPLan test purposes can serve as the basis for test purpose publication or other presentation formats. TPLan test purposes seem especially attractive for generation of test specification stubs. They contain a considerable amount of information regarding initial conditions, verdict criteria and interaction of entities. Nevertheless there are many details which are not specified in test purposes but which are required for test case specification such as preamble implementation, complete message values, guarding against unexpected behaviour, postamble implementation, etc. To make test case generation as complete as possible we expect it to almost certainly be based on and driven by domain-specific TPLan vocabulary and semantics as well as other external sources of input. But once a clever approach is found it will be possible to develop code generators for many different testing languages since TPLan is independent of a specific test case specification language.

Another interesting application for future TPLan tools is the automatic validation of manually written test cases against TPLan test purposes to determine whether or not a test case implementation fulfils the criteria specified in the associated test purposes. This could be an interesting idea, e.g., for companies that define test purposes but subcontract test case specification.

When used in a model-based testing context, TPLan test purposes can be used in the definition of coverage criteria or testing directives for test generation from formal executable models. Here, the test purpose would define a path through model behaviour. Similarly as in the case of test case stub generation, the abstract nature of test purposes has to be again taken into account in model-based test generation. A stimuli or response of a test purpose specification may correspond to only one state transition but also to a path or even multiple paths in the model (see our discussion of "`accepts`" in Section 3.3). Secondly, the faithful use of data specified in a test purpose is non-trivial to handle in test case generation. Data is often not hard-coded but computed during the execution of models. Therefore, for example, it has to be ensured that the event parameter values specified in a test purpose are truly sent or expected by the generated test case.

6 Conclusions

In this paper we have introduced the new notation TPLan which has been developed and standardized by ETSI for expressing test purposes. TPLan attempts to formalize the specification of test purposes by requiring a certain structure and composition based on a set of well defined concepts, i.e., entities, event, value, units, conditions and words. It is independent of a specific testing technology or application domain. A key concept in TPLan is that the pre-defined vocabulary can be extended and customized by users for specific application domains. User definitions make it possible to add more meaning to test purposes and to customize the notation for a specific application domain.

This notation has already been used extensively by ETSI in its IPv6 and dPMR test specification developments. Experiences have been positive in that the quality of test purpose specifications was easier to monitor and affect. Further study is however still required to investigate other impacts of TPLan use such as its effect on overall test specification process. We expect that TPLan will have an even bigger impact on test specification development once more sophisticated tools for handling test purposes become available. Most interest seems to be in the generation of test case specification stubs from test purposes as well as the use of test purposes as a driver for model-based test generation. Some tools, such as a free simple parser and syntax highlighter, are already publicly available at [18].

In the future we see that TPLan standardization is likely to be extended. Currently the creation of TPLan profiles for specific application areas is under discussion, for example, for communicating systems. Such profiles will essentially just extend the pre-defined vocabulary and define semantics for the later. In addition, we are planning to study further existing work on requirements definition languages which are closely related to definition of test purposes.

Acknowledgments

We would like to thank all experts who took part in TPLan test purpose specification in the ETSI IPv6 and dPMR projects for their constructive comments and feedback on the notation itself. In addition, we would like to recognize Dr. Thomas Deiß for his extensive reviews of TPLan standard drafts.

References

[1] ETSI ES 202 553: “Methods for Testing and Specification (MTS); TPLan: A Notation for expressing Test Purposes”, European Telecommunications Standards Institute, Sophia Antipolis, 2007.

[2] C. Willcock et al., *An Introduction to TTCN-3*, Wiley & Sons, 2005.

[3] S. Moseley, S. Randall, and A. Wiles, “Experience within ETSI of the combined roles of conformance testing and interoperability testing”, in *Proceedings of 3rd Conference on Standardization and Innovation in Information Technology (SIIT)*, Delft, The Netherlands, October 2003, 177-89.

[4] S. Randall, “Descriptive SDL”, *Telektronikk*, no. 4, Telenor AS, 2000, p. 107-12.

[5] H. Neukirchen, Z.R. Dai, J. Grabowski, “Communication Patterns for Expressing Real-Time Requirements Using MSC and their Application to Testing”, in *Proceedings of 16th Conference on Testing of Communicating Systems (TestCom 2004)*, Oxford, UK, March 2004, LNCS 2978, Springer, pp. 144-159.

[6] A. Hartma and K. Nagin, “The AGEDIS tools for model based testing”, in *Proceedings of the 2004 ACM SIGSOFT international Symposium on Software Testing and Analysis (ISSTA ’04)*, Boston, MA, ACM Press 2004, pp. 129-32.

[7] ISO/IEC 9646-1: “Information Technology - Open Systems Interconnection - Conformance Testing Methodology and Framework - Part 1: General concepts”, Geneva, 1994.

[8] ISO/IEC 9646-2: “Information Technology - Open Systems Interconnection - Conformance Testing Methodology and Framework - Part 2: Abstract Test Suite Specification”, Geneva, 1994.

[9] C. Jard and T. Jeron, “TGV: theory, principles and algorithms”, in *Proceedings of 6th World Conference on Integrated Design and Process Technology* (IDPT 2000), Pasadena, California, USA, June 2002.

[10] A. Desmoulin and C. Viho, “Formalizing Interoperability for Test Case Generation Purpose”, in *Proceedings of IEEE Nasa ISoLA Workshop on Leveraging Applications of Formal Methods, Verification, and Validation*, Columbia, MD, USA, September 2005.

[11] J. Tretmans, *A Formal Approach to Conformance Testing*, Ph.D. Thesis, University of Twente, The Netherlands, 1992.

[12] P. Deussen and S. Tobies, “Formal Test Purposes and The Validity of Test Cases”, in the *Proceedings of the 22nd International Conference on Formal Techniques for Networked and Distributed Systems (FORTE 2002)*, Houston, Texas, USA, November 2002, LNCS 2529, Springer.

[13] J. Grabowski, D. Hogrefe, and R. Nahm, “Test Case Generation with Test Purpose Specification by MSCs”, in *SDL'93 - Using Objects* (Eds: O. Faergemand, A. Sarma), North-Holland, October 1993.

[14] Object Management Group: *UML 2.0 Testing Profile Specification*, 2003.

[15] S. Deering and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification”, *IETF RFC 2460*, December 1998.

[16] ETSI TS 102 490 (V1.3.1): “Electromagnetic compatibility and Radio spectrum Matters (ERM); Peer-to-Peer Digital Private Mobile Radio using FDMA with a channel spacing of 6,25 kHz with e.r.p of up to 500 mW”, European Telecommunications Standards Institute, Sophia Antipolis, 2006.

[17] ETSI TS 102 237-1 (V4.1.1):” 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, Sophia Antipolis, 2003.

[18] http://www.tplan.info