HyperMSC – a Graphical Representation of TTCN

Ekkart Rudolph[1], Ina Schieferdecker[2], Jens Grabowski[3]

[1]Technische Universität München, Institut für Informatik, D-80290 München
Email: rudolphe@informatik.tu-muenchen.de

[2]GMD FOKUS, Kaiserin-Augusta-Allee 31, D-10589 Berlin,
Email: Schieferdecker@fokus.gmd.de

[3]Medizinische Universität zu Lübeck. Institut für Telematik, Ratzeburger Allee 160,
D-23538 Lübeck, Email: grabowsk@itm.mu-luebeck.de

Abstract

The development of an MSC based graphical representation of TTCN is part of the ETSI project STF 156 on 'Specification of a Message Sequence Chart/UML format, including validation for TTCN-3'. The most important language constructs of TTCN can be translated into corresponding MSC constructs in a straightforward manner. However, it turns out that without certain extensions and modifications, the ITU standard language MSC is not capable to produce a sufficiently transparent and readable graphical representation of TTCN. In order to arrive at a really convincing solution, HMSCs are re-interpreted in a way, which has an analogy in hypertext-like specifications. MSC references may be shown also in an expanded manner and non-expanded MSC references may contain hypertext-like descriptions instead of pure reference names. As a result, there does not exist a strict borderline between HMSCs and BMSCs any longer. Such a generalisation of MSC is really efficient only together with a corresponding tool support, which allows the smooth transition between different levels of detailed description similarly to hypertext. Therefore, the name 'HyperMSC' is proposed for such extended HMSCs. The paper demonstrates how HyperMSCs arise from standard MSC descriptions in an evolutionary process of stepwise improvement and simplification.

Keywords: TTCN, MSC, HMSC, testing, software engineering, distributed systems, telecommunication

1 Introduction

The third edition of TTCN[1] is a textual test specification language, which looks somehow like a common programming language, e.g., C or C++ [1,2,8]. As such, it allows the use of different graphical presentation (display) formats. Apart from the tabular (conformance testing) presentation format known from TTCN-2 [9], the development of a Message Sequence Chart (MSC) format appears to be of special interest and therefore is part of the ETSI project STF 156 [10].

The ITU-standard language MSC is a widespread and popular means for the visualisation of system runs within (Tele-)communication systems [13]. A main advantage of the MSC language

[1] In the following the terms TTCN-2 and TTCN-3 are used to distinguish explicitly between the second and the third edition of TTCN.

is its clear graphical layout, which immediately gives an intuitive understanding of the described behaviour. Within the area of conformance testing, MSC is already well established for the specification of test purposes and as such for the automatic generation of TTCN-2 test cases [3]. Practice has shown that the tabular format of TTCN-2 is not very intuitive for behaviour description even if tools are used. Using MSC as presentation format for TTCN-3 may considerably improve the readability of test cases and make them more understandable. At the same time, MSC in form of Sequence Diagrams forms a central constituent of UML and is employed for the formalisation of Use Cases [5]. Therefore, an MSC format could bring the use of TTCN significantly closer to users of UML. Since there is no accepted test notation in UML, this is an ideal opportunity to bring TTCN-3 closer to the UML world.

MSC has its strong points in the behaviour description. Therefore, the MSC presentation format for TTCN-3 will concentrate on the clear and intuitive presentation of test behaviour. MSC can make use of any data type language to describe data dependencies within MSC. Therefore, data types and test data will be presented by re-using TTCN-3 textual format for data type and data definitions.

Though MSC has been used for test specifications in the past only in a fairly restricted manner, the powerful composition mechanisms contained in the present version of the MSC language make a comprehensive MSC specification of test cases feasible. For that, both MSC inline expressions and High-Level MSCs (HMSCs) [4,13] may be employed depending on the level of abstraction and the focus of representation. However, a naive translation of TTCN-3 into MSC would lead to overloaded diagrams, which are difficult to read and to handle.

The paper is structured in the following manner: A short overview of TTCN-2 and TTCN-3 is given in Chapter 2. In Chapter 3, it is demonstrated that the naive translation of TTCN specifications into nested MSC inline expressions, in general, leads to diagrams, which are less transparent than the tabular tree-like representations. As an example, a small TTCN-2 description of the test suite for the ISDN supplementary service multiple subscriber number (MSN) [7] is used. Within Chapter 4, the same example is employed in order to develop an appropriate MSC test format by extending HMSCs to HyperMSCs. Finally, in Chapter 5, the proposed MSC test format is discussed with respect to future elaboration.

2 TTCN

The Tree and Tabular Combined Notation (TTCN) is a semi-formal test notation that supports specification of abstract test suites for conformance testing [1,2]. An abstract test suite is a collection of abstract test cases. TTCN-2 is defined in part 3 of the Conformance Testing Methodology and Framework for OSI-based systems (e.g. communication protocols and services) standardised by ISO/IEC and ITU [11]. TTCN-2 is the notation to express conformance testing concepts of this framework. The T for tabular refers to the use of tables (proformas) for the graphical representation of test suites. The T for tree refers to the hierarchical organisation of a test suite as well as to the tree-like behaviour of test cases. It has been proven that TTCN-2 is applicable in a much wider scope of applications than OSI protocols and services such as for conformance testing of ODP, TINA and CORBA [6], and IP-based systems, APIs, and reactive systems in general.

Currently, the third edition TTCN (TTCN-3) is worked out by ETSI [1,2]. TTCN-3 is a text-based language for the specification of tests for reactive systems. TTCN-3 is on syntactical (and methodological) level a drastic change to TTCN-2, however, the main concepts of TTCN-2 have been retained and improved and new concepts have been included, so that TTCN-3 will be applicable for a broader class of systems. New concepts are, e.g., a test execution control to describe relations between test cases such as sequences, repetitions and dependencies on test

outcomes, dynamic concurrent test configurations and test behaviour in asynchronous and synchronous communication environments. Improved concepts are, e.g., the integration of ASN.1 [12], the module and grouping concepts to improve the test suite structure, and the test component concepts to describe concurrent test configurations.

In addition to the pure textual format, TTCN-3 will define at least two presentation formats: a tabular conformance testing presentation format that resembles the tabular form of TTCN-2 [9] and an MSC presentation format that supports the presentation but also development of TTCN-3 test cases on MSC level [10].

The top-level unit of TTCN-3 is the module, which can import definitions from other modules. A module consists of a definitions part and a control part [1,8]. The definitions part of a module covers definitions e.g., for test components, their communication interfaces (so called ports), type definitions, test data templates, functions, and test cases. The control part of a module calls the test cases and describes the test campaign. For this, control statements similar to statements in other programming languages (e.g., if-then-else and while loops) are supported. They can be used to specify the selection and execution order of individual test cases.

Test cases describe the probes during the test campaign, i.e., they specify the test behaviour. One can express a variety of test relevant behaviour within a test case such as the alternative reception of communication events, their interleaving and default behaviour to cover, e.g., unexpected reactions from the tested systems. In addition to the automatic test verdict assignment, more powerful logging mechanisms e.g., for a detailed tracing, are provided.

2 Motivation

TTCN-2 statements are written on successive lines with either successively incremented indentations to indicate subsequent statements or with equal indentations to indicate alternatives. In case of highly nested alternatives, such a notation is not very user-friendly. In this context, it should be noted that in the beginning of the standardisation of test notations (1983), Time Sequence Diagrams were considered as candidate but rejected because they seemed to be not sufficiently precise.

TTCN-3 does not require the TTCN-2 tree-like style of describing test cases. Instead, the more obvious sequential programming style, where the sequence of statements is the dominating means for test specification, can be used. The sequential style is easier to translate into MSC, but for completeness and backwards compatibility to TTCN-2, i.e., the ability to present TTCN-2 test cases with MSC, the MSC presentation format has to cope with all specification styles. In this paper, we concentrate on the complex tree-like style and use a TTCN-2 test case as basis for out considerations.

The most obvious and straightforward way to represent TTCN-2 test cases by MSC diagrams would be to use inline operator expressions for alternatives, iterations etc. whereby depending on the test verdict, the representation of alternatives by means of the exception operator may be more suitable. Practice has shown that apart from simple cases such a 'naive' translation does not lead to diagrams, which are easy to read and understand. As a consequence, it is not clear whether such an MSC format would mean a progress with respect to the TTCN-2 notation or even a step back. In particular, inline operator expressions obscure the message flow of the 'standard' cases (PASS verdict) by mixing it with alternative parts. This is demonstrated by the following example taken from [7]. It shows the preamble of the MSN_N01_001 test case.

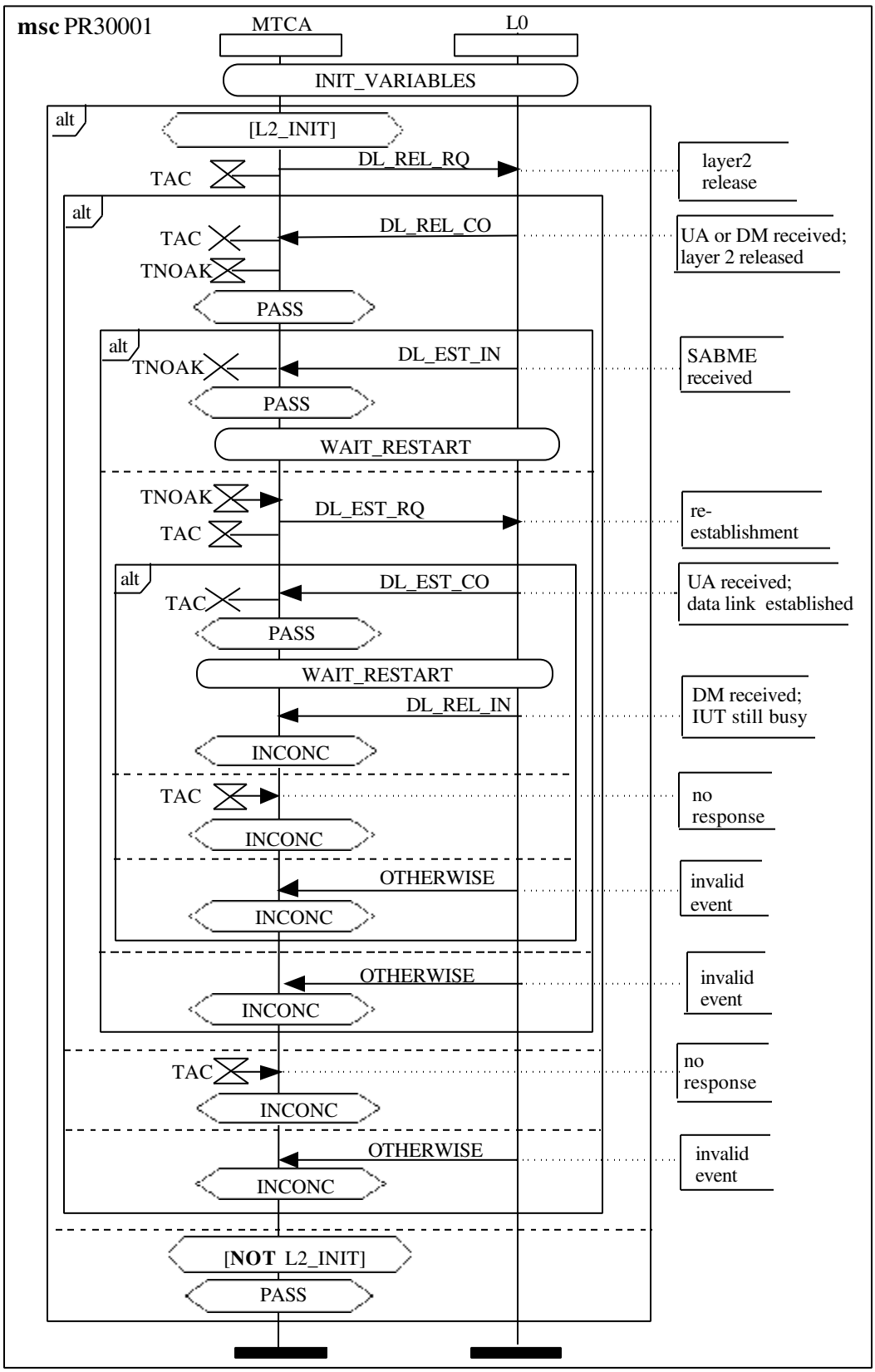

Figure 1: ‘Naive’ translation of TTCN into MSC inline expressions

Obviously, the resulting MSC (Figure 1) looks quite complicated. The MSC PR0001 in Figure 1 may serve as a strong motivation for the development of a more convincing MSC test notation. In case of few alternatives, however, such an inline representation may be still sufficiently transparent for special purposes. Therefore, an inline representation should be not ruled out completely. Within the following section, we will demonstrate the stepwise evolution of a graphical MSC representation, which appears to be highly transparent also in case of nested alternatives.

3 From TTCN to MSC – the Invention of HyperMSC

In the following, we employ a slightly smaller example of [7] for simplicity, namely the test case MSN_N01_001 (Figure 2). We start again with the MSC inline representation for the test case MSN_N01_001 (Figure 3).

Test Case Dynamic Behaviour

Test Case Name	:	MSN_N01_001
Group	:	CalledUser/
Purpose	:	Ensure that the IUT in the Null call state N00, to indicate an incoming call and only the partial ISDN number is available, sends a SETUP message with a Called party number information element with type of number coded as "unknown", numbering plan identification field coded as "unknown" or as "ISDN/telephony numbering plan", and MSN digits and enters state N06.
Configuration	:	CONFIG1
Default	:	DF69901(1)
Comments	:	Selection:IUT supports insertion of partial ISDN number in Called party number information element. PICS: SC 3.2. PIXIT: How to configure the IUT so that the user access has MSN subscribed.

Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		CREATE (PTC1:PTC1_IN)			
2		+PR30001			
3		CPA1!CP_M START TWAIT	S_SU1		
4		L0?SETUPr (CREF1 := SETUPr.mun.cr.cr_r) CANCEL TWAIT	A_SU11	(P)	valid SETUP
5		+CS59901(6,1)			
6		L0?SETUPr (CREF1 := SETUPr.mun.cr.cr_r) CANCEL TWAIT	A_SU1	(F)	(1)
7		+ PO49901(1)			postamble N0
8		?TIMEOUT TWAIT		(I)	no response
9		+END_PTC1			

Detailed Comments : (1) A SETUP not according to the test purpose is received.

Figure 2: TTCN-2 representation of the test case MSN_N01_001

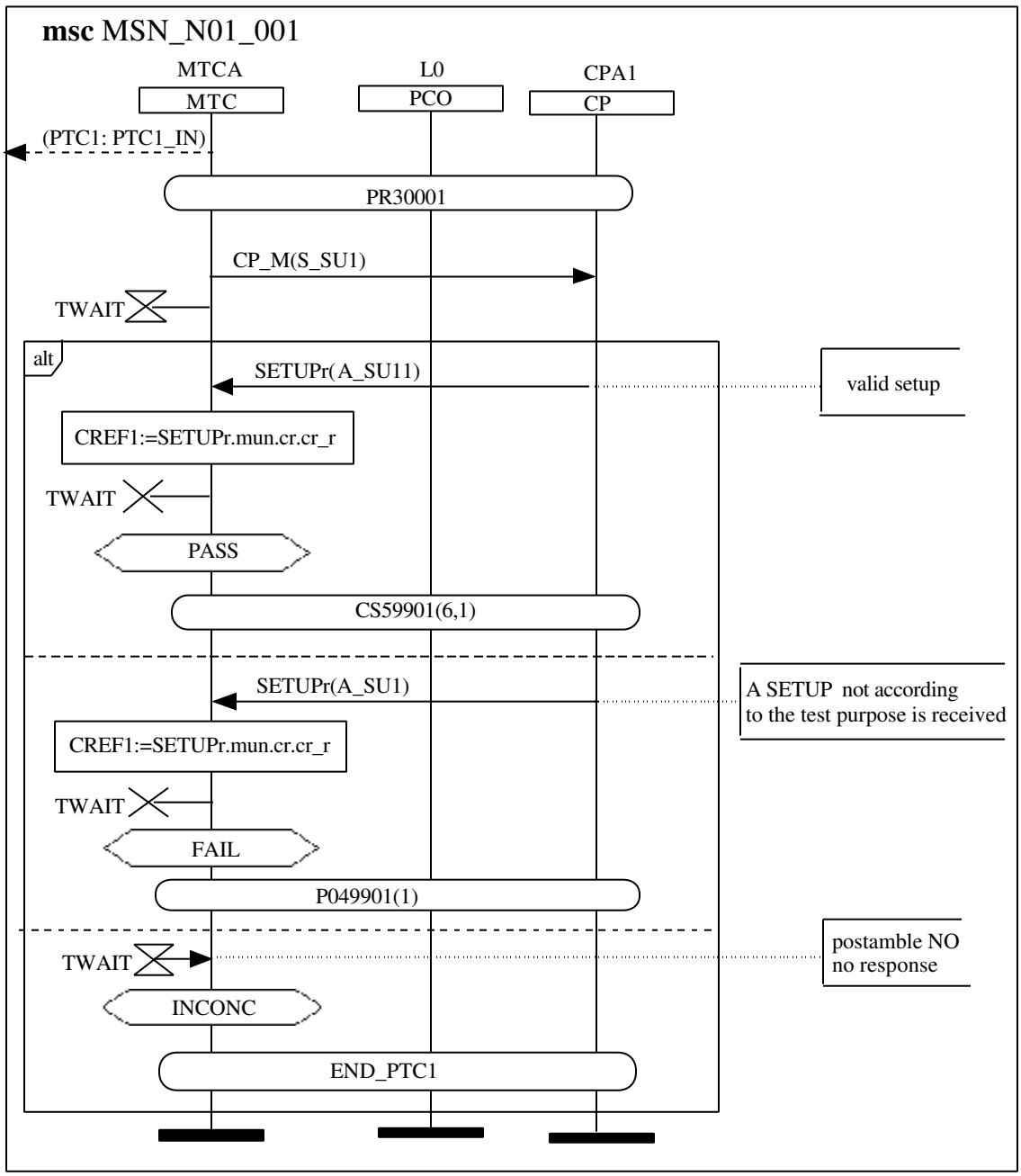

Figure 3: Representation of the test case MSN_N01_001 using inline expressions

Apart from the usage of inline expression, there are some other features to be mentioned. In comparison with the TTCN-2 representation, the comments are represented in form of MSC comments and the test verdicts are described making use of conditions.[2] L0 and the co-ordination

[2] In contrast to TTCN-2, TTCN-3 only has preliminary verdicts and no final verdicts, i.e., for the termination of test cases and test components explicit stop operations have to be used.

point CPA1 are represented explicitly in form of instances [3].[3] The create message connected to the environment contains an extension to MSC-2000 by assigning the MSC reference as behaviour description to the created instance.

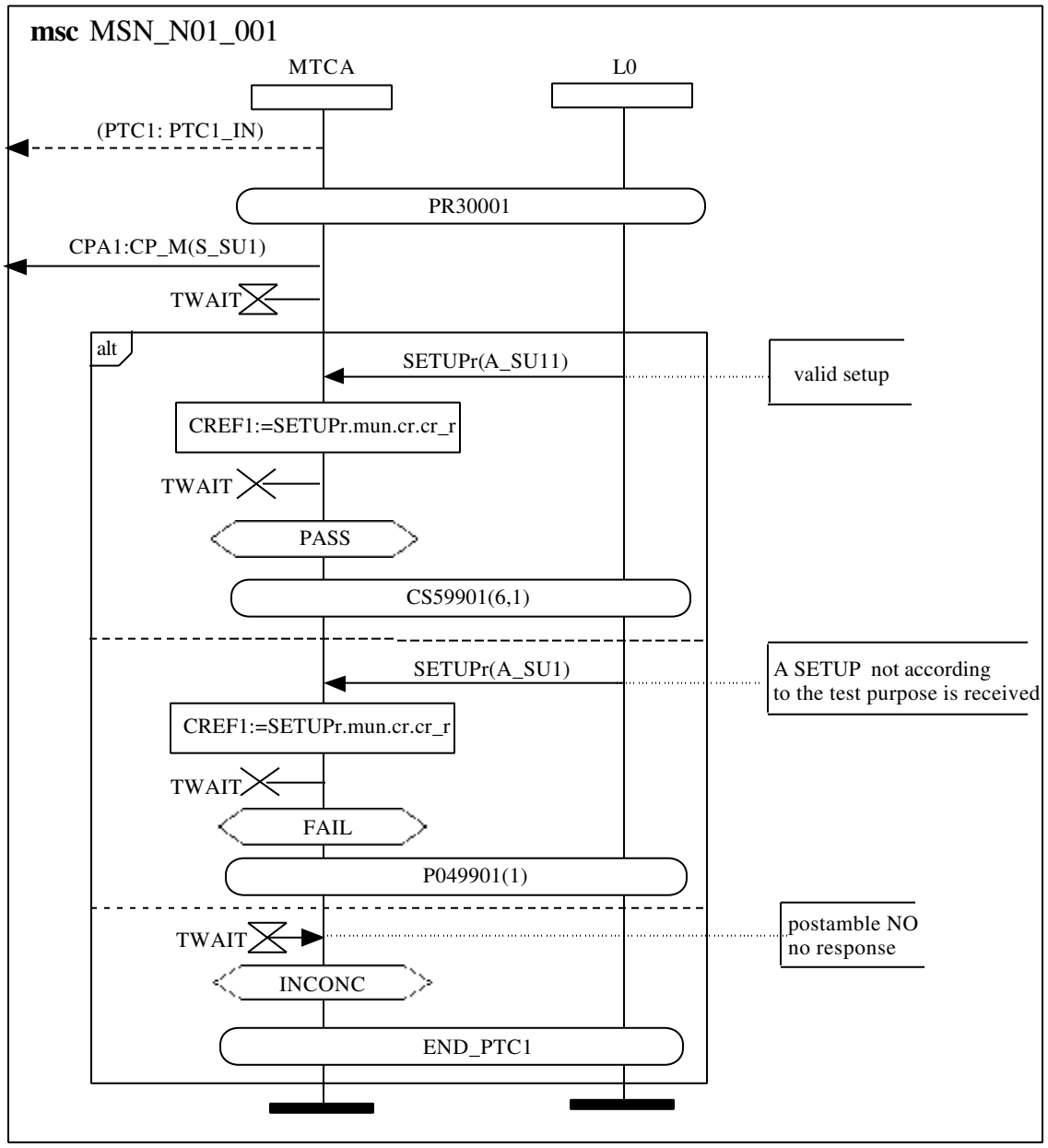

Figure 4: Representation of MSN_N01_001 with messages sent to the environment

In Figure 4, we have employed a more compact notation where also the co-ordination message CP_M to CPA1 is connected with the environment. In both cases (Figure 3 and Figure 4), a

[3] In TTCN-3, ports, which are mapped to the test system interface, represent PCOs from TTCN-2 and ports, which are connected to ports of other test components, represent CPs from TTCN-2.

second MSC PTC1_IN has to be specified describing the test events on the PTC1 side. Both MSCs have to be merged via create and co-ordination messages.

The MSC representation of MSC MSN_N01_001 in Figure 3 and Figure 4 by means of inline expressions is not extremely complicated. It is used nevertheless to demonstrate the development of other forms of representation, which are more suitable for a transparent visualisation of test sequences particularly in complex cases like Figure 1. An obvious drawback of the representation by means of inline expressions is the fact that the PASS-, INCONC- and FAIL-cases all appear in the same way. Certainly, it would be advantageous to have a means to single out the normal (PASS-) case. An immediate generalisation of this idea is to allow also representations, where INCONC- and FAIL-cases are represented in a condensed form. The MSC standard MSC-2000 contains several structuring mechanisms [13]. For this special purpose, the MSC reference mechanism is tailored. In order to highlight the PASS-case, we represent the other cases in form of MSC references. The result is shown in Figure 5.

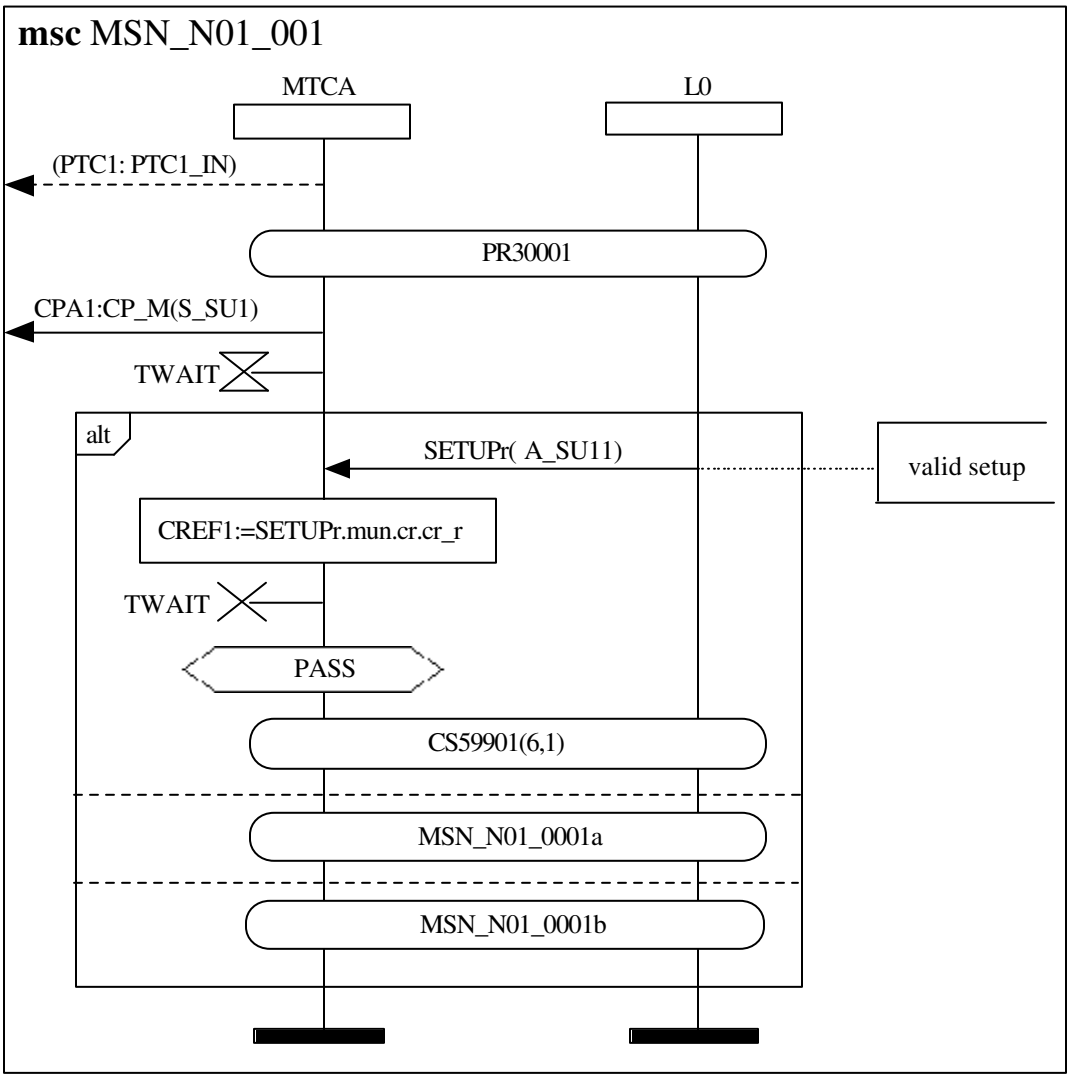

Figure 5: Using MSC references for the non-PASS cases

According to the MSC language, corresponding MSCs MSN_N01_001a and MSN_N01_001b have to be defined in the MSC document containing the events of the FAIL- and INCONC-cases, respectively. This notation has an immediate drawback: Without looking at the definitions of the referenced MSCs, there is no information in Figure 5 on the non-PASS cases. That means, apart from the PASS-case, the representation of the test case is not very intuitive. A much better presentation is provided in Figure 6.

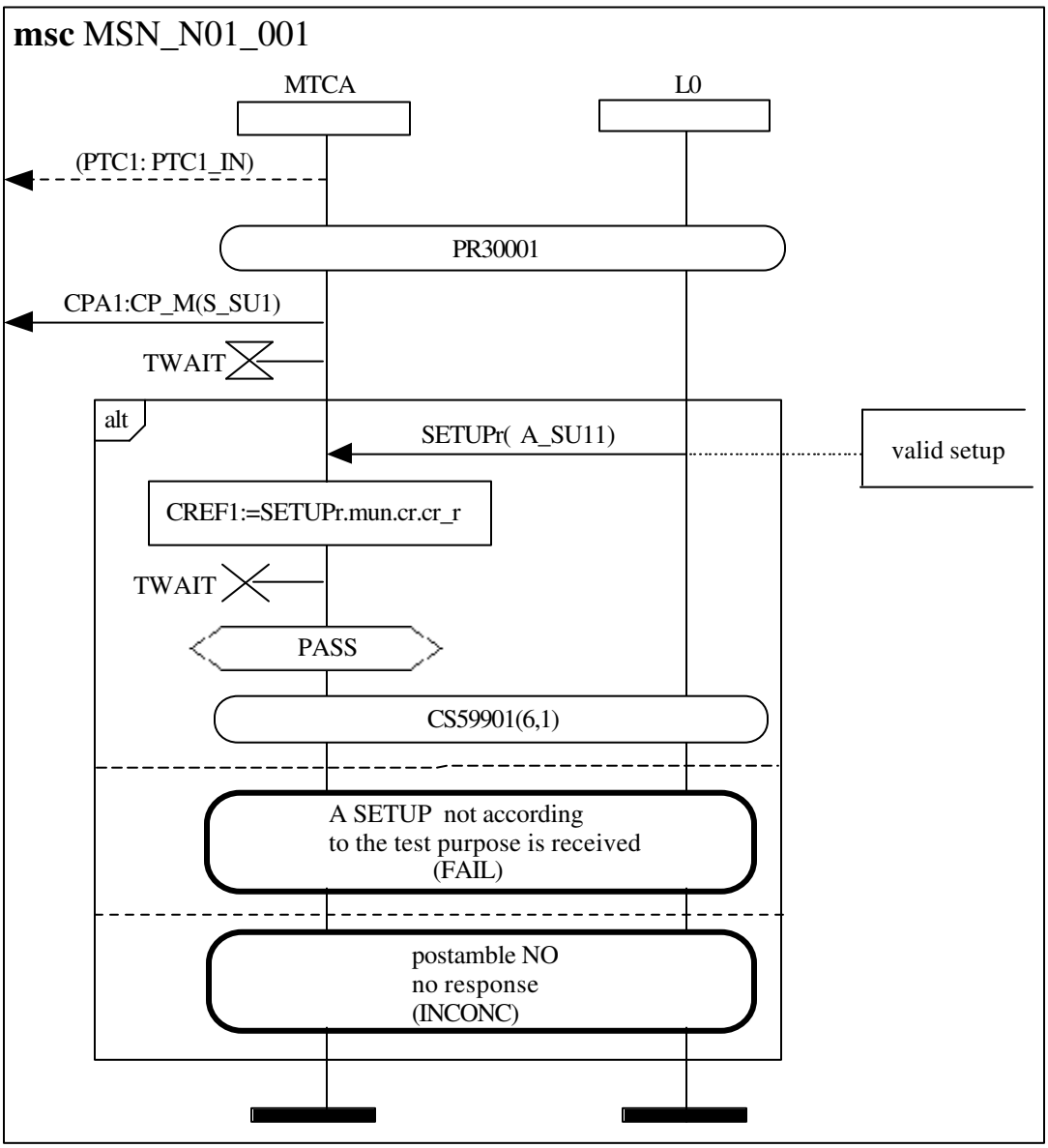

Figure 6: Using MSC references with textual inscriptions

Now, we have arrived at a notation, which is already quite satisfactory in many respects. We have included the comment text together with the test verdict in the MSC reference symbols. It should be noted, however, that we deviate slightly from the MSC standard: The MSC references do not contain a name but an arbitrary text. In a sense, this can be viewed as a generalisation of the MSC reference name convention. However, in practice it seems to be more appropriate to interpret this

description in a hypertext–like manner: We assume a corresponding tool support where the MSC references can be expanded within the embedding MSC or possibly also in a separate window. The MSC references, which can be expanded, may be indicated by underlining the text, by coloured text or by a variation of the line width of the symbol lines (within this paper, the variation of the line width is used). Such an approach is appropriate particularly in the usual case of many fairly small MSC reference definitions. Because of this analogy, we have introduced the notation 'HyperMSC', which shall indicate not only a special syntax form but also a corresponding tool support.

In the following, the idea of 'HyperMSC' will be further outlined. In case of many alternatives, the inline representation shown in Figure 6 is still not yet very transparent. As a general rule, inline expressions should be used only in a very limited manner and should be restricted to a few alternatives or loops. In more complex situations, HMSCs are much more transparent since they abstract from details and focus on the compositional structure [4,13]. However, if we translate the MSC of Figure 6 into an HMSC we are faced with the problem to represent the expanded parts whereas standard HMSC contain only non-expanded MSC reference symbols. In order to overcome this deficiency, we admit an expanded form of MSC references within HMSCs [4]. With this additional extension of the MSC language, we arrive at the HyperMSC in Figure 7.

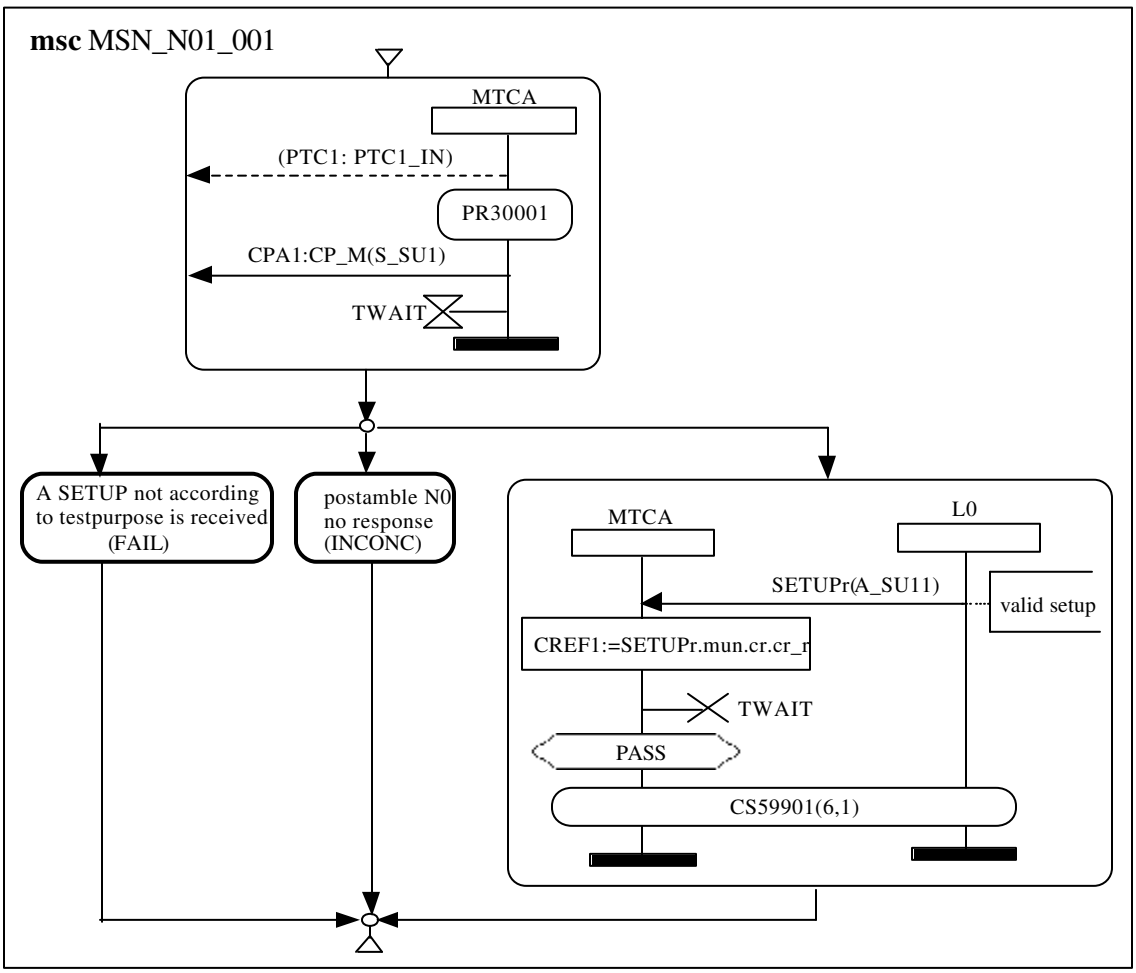

Figure 7: Representation of MSN_N01_001 in form of generalised HMSC (HyperMSC)

The other MSC references may be also expanded either inline or in a separate window depending on the special situation. The other way around, expanded MSC references may be closed. As an additional feature, complete paths in the HMSC may be expanded and shown in expanded form as a coherent MSC in a separate window.

Eventually, one would like to have such a coherent expanded representation of a whole path not only in a separate window but also in inline-form within the HMSC itself. This is advantageous, in particular, to show the PASS-case in a coherent manner. In comparison with the tree and tabular notation, the MSC in Figure 7 still has the drawback that the main event flow (PASS-case) is split into separate parts. In case of many alternatives, this splitting is very disturbing. As a consequence, this MSC test format still may not appear as a progress in every respect in comparison with the traditional TTCN-2 representation. We therefore suggest a further extension of HMSCs, which somehow may be viewed also as a unification of HMSC and BMSC (basic MSC). We combine the expanded MSC references in Figure 7 to one coherent expanded MSC reference. As a consequence we have to shift the connection point to the borderline of the resulting MSC reference. This procedure is illustrated in Figure 8. The marked MSC references may be shown in expanded form (Figure 9).

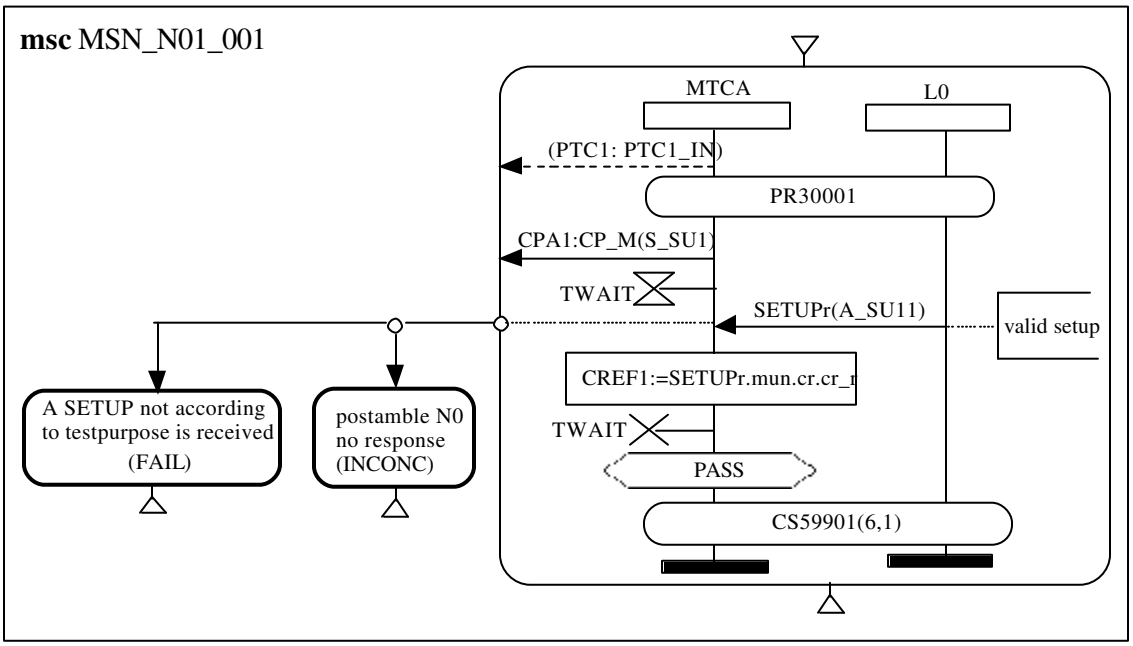

Figure 8: HyperMSC with generalised graphical branching construct

With suitable tool support, the most attractive alternative representation, however, would be to change the roles of the PASS-case and the INCONC- or FAIL-case (Figure 10), i.e., the test engineer selects the information he needs to examine by choosing an appropriate view.

In order to prove the power of this HyperMSC we come back to our first example in Figure 1. Using the final HyperMSC representation, also the preamble PR0001 can be put into a form, which looks simpler and much more transparent (Figure 11).

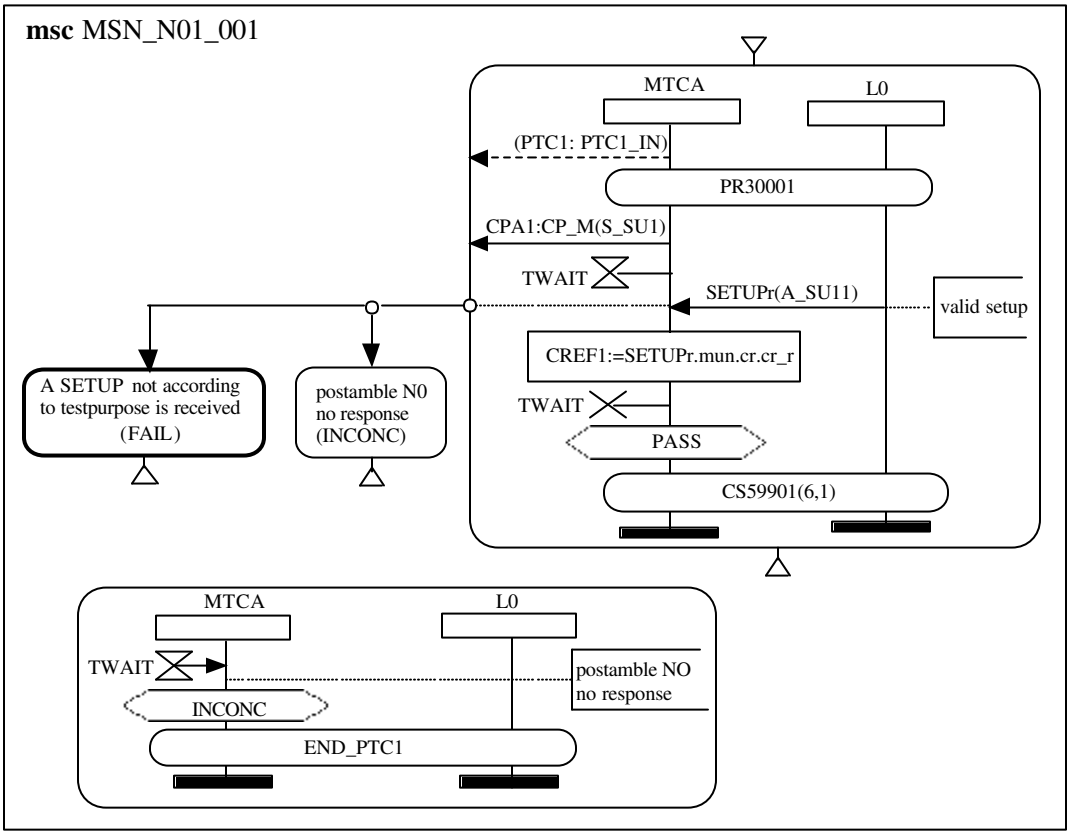

Figure 9: Expansion of an MSC reference in a separate representation

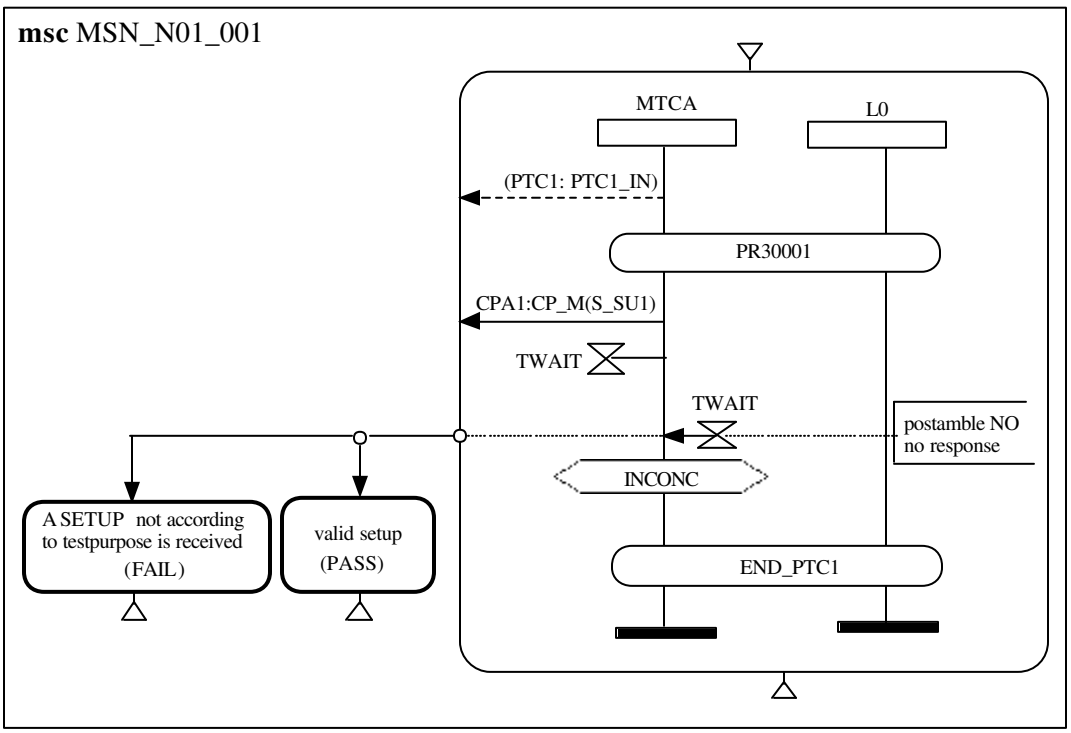

Figure 10: Interchange of the roles of the PASS- and INCONC-cases

In Figure 11, the diagram has been condensed by introducing a default construct [1,3] for *invalid events*. This default construct is a real extension of the MSC language and certainly needs further studies. Beyond that, the reference re-establishment comprises a nested alternative, which may be expanded in a separate window as is shown in Figure 12.

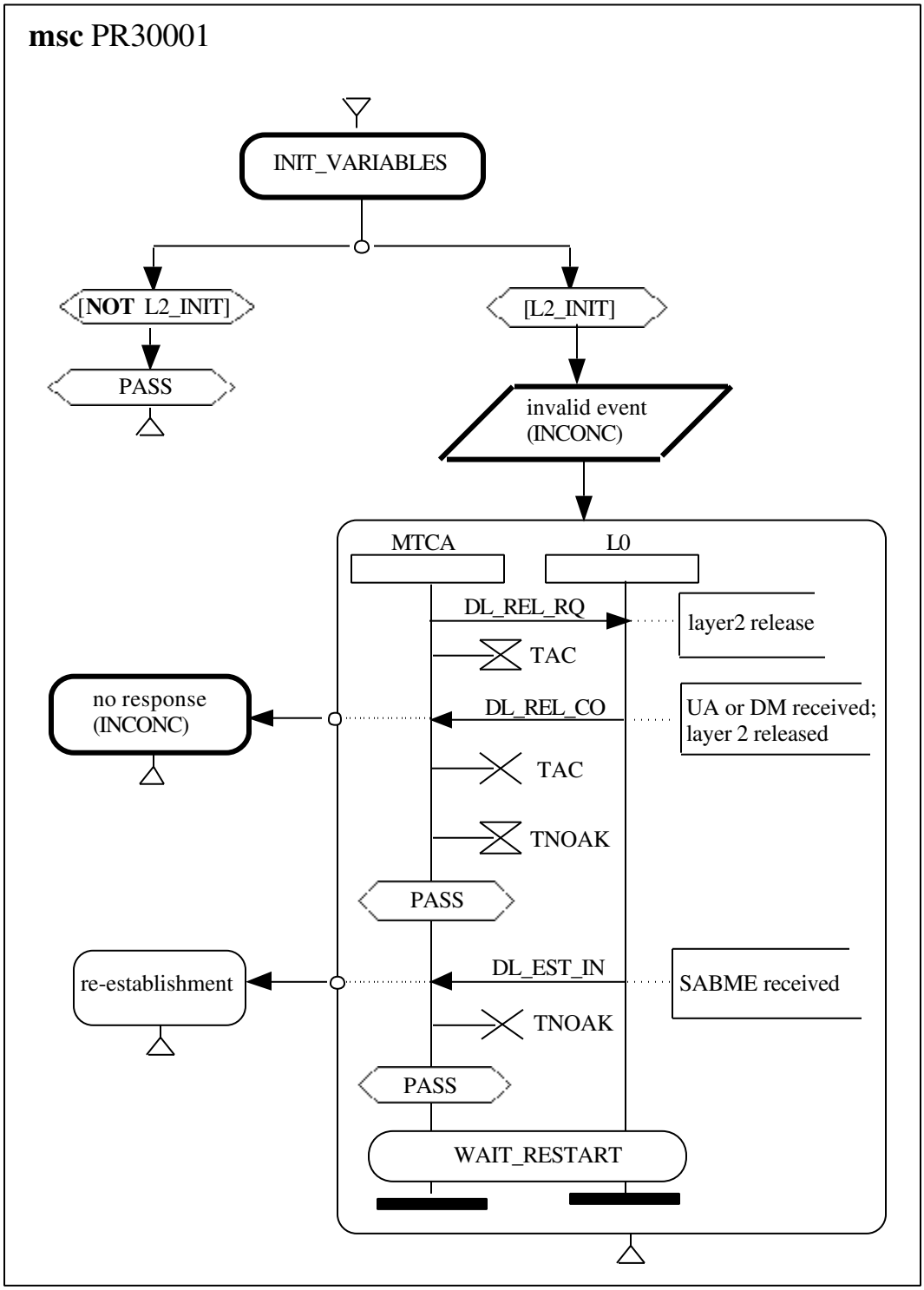

Figure 11: HyperMSC representation of the preamble PR0001

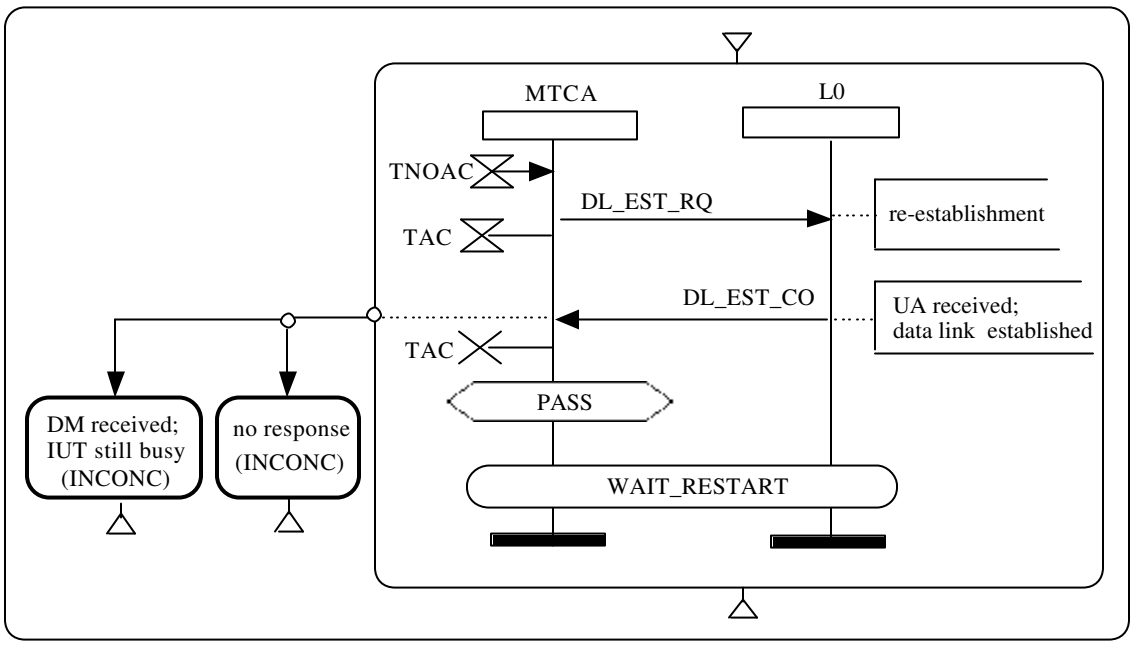

Figure 12: Expanded form of the MSC reference re-establishment.

Note, that Figure 12 shows an expanded MSC reference but not an MSC. The HyperMSCs in Figure 11 and 12 exhibit the main paths (PASS-cases) in a coherent way in form of an expanded MSC diagram while the side cases (INCONC-cases) are indicated by branching with non-expanded MSC references containing textual descriptions. This suggests another attractive HyperMSC application, which immediately combines the MSC format with the TTCN representation: The MSC references describing the side cases may contain even TTCN language descriptions instead of comments. Such a hybrid representation may be particularly useful for documentation purposes since it provides a complete test case description with a visualisation of the main paths in form of the MSC format.

As a consequence, even more than two levels of MSC reference representations may be distinguished. There are four possible ways to present a reference: (a) by its proper name, (b) by a textual description (comments and text verdict), (c) by means of TTCN program descriptions, (d) by its full blown MSC diagram.

4 Conclusions and outlook

Obviously, the MSC standard needs certain extensions in order to be applicable for a graphical representation of test cases. Neither MSC inline expressions nor the standard HMSCs can be employed immediately for the development of an MSC test format, which is sufficiently transparent and readable. Though the tree-like representation of TTCN-2 test cases is far from being ideal, at least, it describes the normal cases in a coherent manner. An MSC test format, which really can be looked at as a progress with respect to the tree-like representation demands a different handling of HMSCs together with an appropriate tool support. The proposed new features leading to the concept of 'HyperMSC' merely concern the layout and the handling but do not impose any substantial changes. In particular, there are no semantics changes requested. Beyond that, most of the changes like expanding MSC references within HMSCs or generalising MSC reference name conventions to comment texts can be looked at as variants, which are already used in practice. That shows that the strict borderline between HMSCs and plain MSCs is

of no practical value and should be removed. The proposal of HyperMSC, therefore, can be looked at also as a unification of the MSC language. Though the motivation for HyperMSC was the development of a MSC test format within ETSI its possible application range is much larger. In particular, HyperMSC may be useful for the modelling and formalisation of Use Cases.

The presented proposal for a MSC test format appears to be a major step towards a more user-friendly graphical representation of TTCN-3. However, there are still several open questions in the development of the MSC test format. While most of the main TTCN-3 language constructs for test behaviour description - communication operations, program statements, functions - allow a straightforward translation into the MSC language, for some TTCN-3 constructs, e.g., the default behaviour or the test configuration [1,8], there are no immediate MSC counterparts. The elaboration of these concepts is part of the ETSI project STF 156.

References

[1] Jens Grabowski: 'TTCN-3 - A new Test Specification Language for Black-Box Testing of Distributed Systems'. Proceedings of the "17th International Conference and Exposition on Testing Computer Software (TCS'2000), Theme: Testing Technology vs.Testers' Requirements", Washington D.C., June 2000.

[2] J. Grabowski, D. Hogrefe: An Introduction to TTCN-3. In: G. Csopaki, S. Dibuz, K. Tarnay, editors, , 'Testing of Communicating Systems – Methods and Applications', Kluwer Academic Publishers, September 1999.

[3] J. Grabowski, T. Walter: Visualisation of TTCN test cases by MSCs. In: Proceedings of the 1st Workshop of the SDL Forum Society on SDL and MSC - SAM'98 (editors: Y. Lahav, A. Wolisz, J. Fischer, E. Holz), Informatik-Berichte Humboldt-Universität zu Berlin, June 1998.

[4] S. Mauw, M. A. Reniers: High Level Message Sequence Charts. In SDL'97 Time for Testing-SDL, MSC and Trends, Proceedings of the 8th SDL Forum in Evry France (A. Cavalli and A. Sarma editors), North Holland, Sept. 1997.

[5] E. Rudolph, J. Grabowski, P. Graubmann: Towards a Harmonization of UML Sequence Diagrams and MSC. In SDL'99, The Next Millenium, Proceedings of the 9th SDL Forum in Montreal Canada (Yair Lahav and R. Dssouli editors), North Holland, June 1999.

[6] I. Schieferdecker, M. Li, A. Hoffmann: 'Conformance Testing of TINA Service Components - the TTCN/CORBA Gateway'. In: Proceedings of the 5th International Conference on Intelligence in Services and Networks, Antwerp, Belgium, May 1998.

[7] ETSI. Integrated Services Digital Network (ISDN); Multiple Subscriber Number (MSN) supplementary service; Digital Subscriber Signalling System No. one (DSS1) protocol; Part 5: Test Suite Structure and Test Purposes (TSS & TP) specification for the network. EN 300 052-5, 1998.

[8] ETSI TC MTS. TTCN-3 - Core Language. European Norm (EN) 00063-1 (provisional)[4], 2000.

[9] ETSI TC MTS. TTCN-3 - Tabular Presentation Format. EN00063-2 (provisional)[4], 2000.

[10] ETSI TC MTS. TTCN-3 - MSC Presentation Format. EN00063-3 (provisional)[4], 2000.

[4] NOTE: The EN-00063 numbers are only provisional ETSI Work Item numbers. The actual EN numbers will not be the same.

[11] International Standardisation Organisation. 'Information Technology - OSI - Conformance Testing Methodology and Framework - Parts 1-7'. ISO, International Standard 9646, 1994 - 1997.

[12] ITU-T Recommendations X.680-683. 'Information Technology - Abstract Syntax Notation One (ASN.1)', 1994.

[13] ITU-T SG 10. Message Sequence Chart (MSC). Rec. Z.120, Geneva 2000.