Visualisation of TTCN test cases by MSCs

Jens Grabowski[a] *and Thomas Walter*[b]

[a]*Medizinische Universität zu Lübeck, Institut für Telematik, Ratzeburger Allee 160, D-23538 Lübeck, e-mail: jens@itm.mu-luebeck.de*

[b]*Eidgenössische Technische Hochschule Zürich (ETHZ), Institut für Technische Informatik und Kommunikationsnetze, ETH Zürich/Gebäude ETZ, CH-8092 Zürich, e-mail: walter@tik.ee.ethz.ch*

Abstract

Practice has shown that concurrent TTCN is suitable to describe complex distributed conformance tests. Unfortunately, the entire test behaviour may be distributed over several TTCN tables and often it is difficult to keep the overall view of the test case when reading TTCN. A combined use of MSC and TTCN may improve the readability of test cases and make them more understandable. In this paper we discuss the use of MSC for the specification of test purposes and the generation of MSCs from TTCN test cases.

Keywords

Conformance Testing, TTCN, MSC, Test Purpose, Test Case Visualisation, TTCN Simulation

1. INTRODUCTION

The ISO Conformance Testing Methodology and Framework (CTMF) [IS9646-1:94] has been primarily developed for the test of communication protocols and services. CTMF focuses on conformance testing, i.e., black-box testing of functional requirements of an implementation under test (IUT) while performing communication tasks.

Functional requirements define the interactions between a system under test (SUT) and a test system. An SUT is a combination of an IUT, e.g., an implementation of a protocol entity, together with an underlying communication service. A test system comprises all software and hardware components necessary to run tests against the SUT. Tests define sequences of stimuli to and the foreseen responses from an SUT.

The test specification language recommended by ISO and ITU-T for the description of conformance test suites for OSI protocols is the Tree and Tabular Combined Notation (TTCN) [IS9646-3:96]. TTCN gives static, dynamic and test specific information in form of tables containing textual information. Practice has shown that real test cases may consist of several hundreds of TTCN tables. Practice has also shown TTCN is not very intuitive, even if tools are used.

A possibility for improving the readability of TTCN test suites is a combined use of the Message Sequence Chart (MSC) language and TTCN. MSC is a formal language recommended by ITU-T in Rec. Z.120 [Z.120:96], i.e., in contrast to TTCN, MSC has a formal semantics definition based on basic process algebra [MR93, Z.120B:95]. MSC can be characterised as a trace language which concentrates on message interchange of communicating entities (such as services, processes, or protocol entities) and their environment. An advantage of an MSC is its clear graphical layout which immediately gives an intuitive understanding of the described system behaviour.

Several attempts have been made to use MSC for test case specification and test case implementation. These attempts make use of the clear and intuitive understanding of MSC diagrams[1], but, have to deal with the problem that MSC is not meant to be a test specification and implementation language. In particular, concepts necessary for expressing testing purposes are not implemented. For example, there exists (a) no notion of test verdicts, (b) no possibility to specify types and constraints, i.e., instances of PDUs and ASPs, (c) no possibility to describe variables and constants for test cases or test suites, and (d) no notion of test architectures, PICS, and PIXIT. As a

[1] In the following the abbreviation MSC is also used to refer to an MSC diagram. In case of ambiguities we will use the terms MSC language and MSC diagram instead.

consequence, MSC is used often in combination with other specification languages for testing purposes(e.g., [GHN93] or [ETSI98]) which provide at least parts of the missing information or additional assumptions. Furthermore, MSC language extensions have been made in order to adapt MSC to the needs of testing (e.g., [GHNS95]).

Our intention is the other way round, we want to improve the readability of TTCN test suites and make them more understandable by proposing a combined use of MSC and TTCN. The MSCs explaining a test suite may be generated in advance of, in parallel with, or, possibly by means of tools, after the TTCN test suite production. There are two reasons for a combined use of TTCN and MSC. First, TTCN is used in standards for describing abstract conformance test suites. Products claiming to conform to a protocol standard have to pass a test based on the corresponding standardised abstract test suite. All manufacturers of such a product have to implement the corresponding standardised test suite. Manufacturers not using TTCN have to interpret the TTCN test cases and (re-)implement them in their environment. Additional MSCs may be helpful to avoid errors due to misinterpretations of the test suite. Second, the dynamic behaviour of a test case in concurrent TTCN is distributed over several TTCN tables where every table contains the dynamic behaviour of a single test component. The combined behaviour of test components is not described explicitly since concurrent TTCN does not provide means for representing the overall configuration of test components and the temporal ordering of test events graphically. Combining concurrent TTCN with MSCs seems to be a feasible approach to enhance the readability of test case specifications.

In the following the use of MSCs for the specification of test purposes (Section 2) and the generation of MSCs from TTCN (Section 0) is discussed. The paper will conclude with a summary and an outlook (Section 4).

2. SPECIFYING TEST PURPOSES BY USING MSCS

The most common combined use of MSC and TTCN is the test purpose (TP) specification by means of MSC and the following definition of the entire test behaviour by using TTCN. The TTCN development based on MSC TP may be done manually or be supported by tools [ETSI98, GSHD97, SEGHK98]. The following two examples taken from Intelligent Networks (IN) [Thö94] and Integrated Digital Services Networks (ISDN) [Hal94, Tan89] explain the use of MSC for TP description.

2.1 IN example

The conformance test suite for the Intelligent Networks Application Part (INAP) Capability Set 2 (CS-2) [ETSI97] is currently under development within the European Telecommunication Standards Institute (ETSI). In contrast to other protocol standards, INAP CS-2 includes a normative and executable SDL specification. Therefore it was decided to generate the INAP CS-2 TTCN test suite by means of tools.

The TTCN generation is based on a test suite structure and test purpose document [ETSI98a] on which the relevant ETSI committee has to agree. The TPs are identified manually and described within an informal tabular notation and as MSC. The MSC TP descriptions were developed by means of simulation, i.e., the SDL specifiction was simulated and the simulation run was recorded in form of an MSC, or specified manually. Pre- and Postambles are referred to by means of MSC references and are also documented in form of MSCs. In order to automate the TTCN generation all message parameters have been included in the MSC diagram. The following example may clarify the procedure.

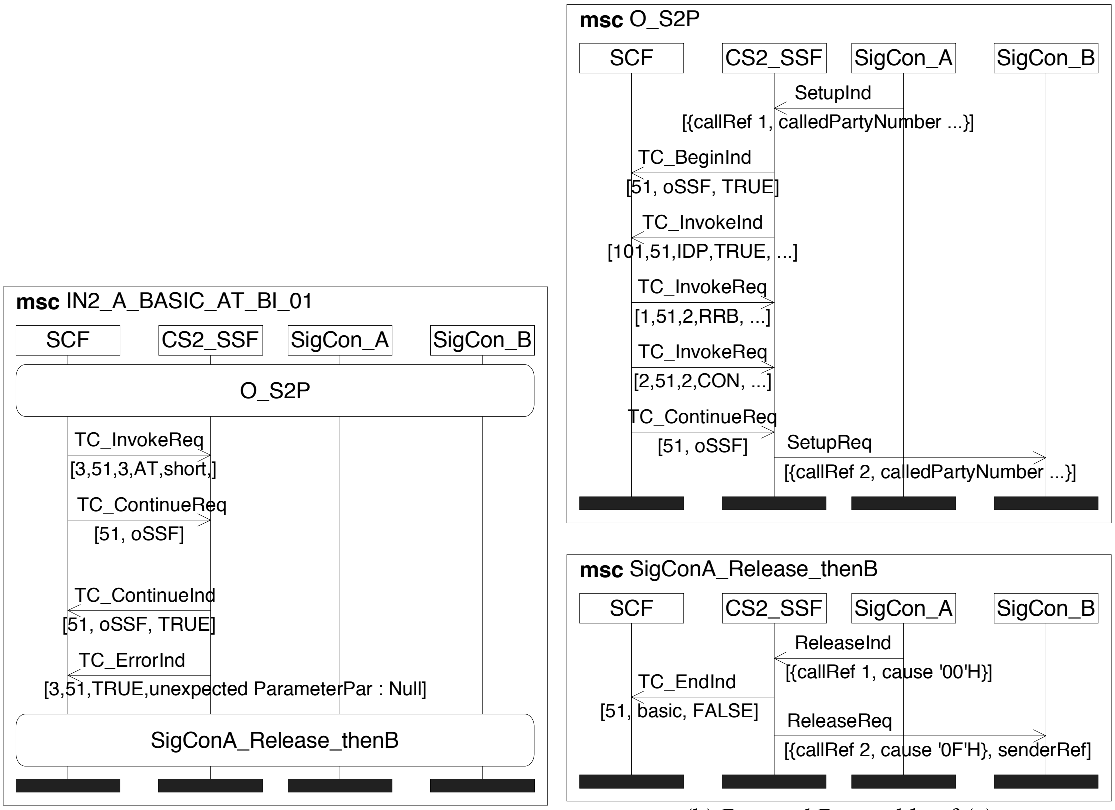
Figure 1 shows the informal tabular description of the test case IN2_A_BASIC_AT_BV_02. The purpose is to test a part of the Activity Test (AT) function of the INAP capability set 1 (CS-1) within INAP CS-2. The test case starts in the state *Originating Stable Two Party* as indicated by the preamble O_S2P. After execution of the test body (described by the entries Test description and Pass criteria) the system has to be driven back into its initial state. By performing the postamble SigConA_release_thenB all connections are released. The corresponding MSC TP representation is presented in Figure 2a[2] the referenced pre- and postamble are shown in Figure 2b. Based on the SDL specification and the presented MSCs the TTCN test case description in Figure 3 has been generated [ETSI98b]. Pre- and Postamble are referred to by using TTCN test steps (which not shown in this paper). By comparing the MSC and TTCN descriptions it can be seen that the MSC TP signal exchange leads to a PASS verdict.

---

[2] For the sake of readability some of the message parameters have been omitted.

IN2_A_BASIC_AT_BI_01	
Purpose:	Test of invalid **Activity Test** (AT) invoke
Requirement ref.:	
Preamble:	O_S2P
Test description:	**Activity Test** invoke with argument sent by SCF to SSF during preamble
Pass criteria	SSF issues TC_error indicating unexpected Parameter
Postamble	SigConA_release_then_B

Figure 1 Informal INAP CS-2 TP description



(a) MSC TP

(b) Pre- and Postamble of (a)

Figure 2 MSC representation of the TP description in Figure 1

Test Case Dynamic Behaviour					
Test Step Name: IN2_A_BASIC_AT_BI_01 **Purpose:** **Default:** OtherwiseFail					
Nr	**L.**	**Behaviour Description**	**Constraints Ref.**	**Verdict**	**C.**
1		+O_S2P			
2		SCF!TC_InvokeReq	CIR_ActivityTest_001 (PIX_InvokeI3,Tsv_DialogId1)		
3		SCF!TC_ContinueReq	C_TC_ContinueReq (Tsv_DialogId1)		
4		SCF?TC_ContinueInd	C_TC_ContinueInd_001 (Tsv_DialogId1)		
5		SCF?TC_ErrorInd	C_TCErrorIndunexpectedParameterPar (PIX_InvokeId3,TsvDialogId1)	(PASS)	
6		+SigConA_Release_thenB			
Detailed Comments:					

Figure 3 TTCN generated from the SDL in [ETSI97] and the MSCs in Figure 2

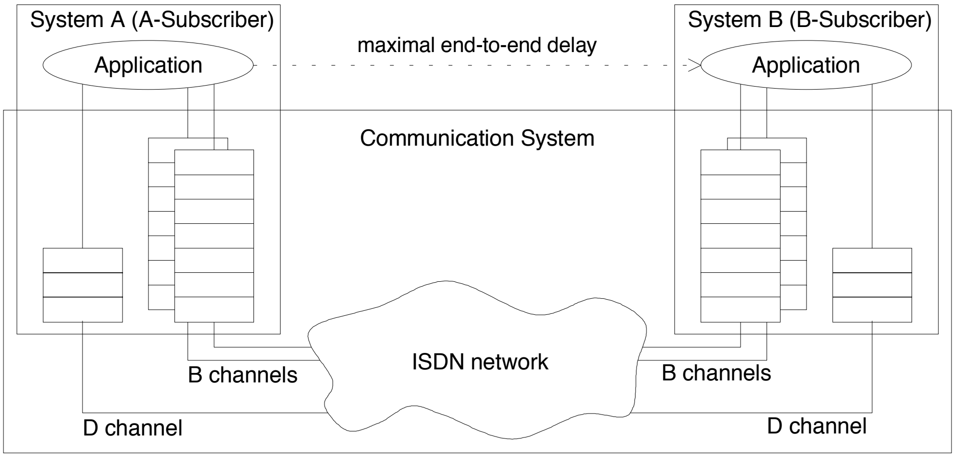


Figure 4 ISDN user interface

2.2 ISDN example

In ISDN [Hal94,Tan89] the basic subscriber interface provides two 64 kbps (kilo bits per second) B channels and one 16 kbps D channel. The B channels are used by applications for data exchange. The D channel is a signalling channel and used for the management of connections between users or application processes. Assuming that communication of application processes is bound by a maximal end-to-end delay, this quality-of-service (QoS) requirement has to be enforced within the communication system (Figure 4). Whenever the end-to-end delay becomes greater than the negotiated end-to-end delay, the communication system should indicate the violation of the QoS requirement to the applications and abort the connection (using the signalling protocol on the D channel).

A test case for testing compliance of the communication system on the B-subscriber[3] site with the above stated QoS requirement would involve three lower testers which control B and D channels on the A-subscriber site and one upper tester which replaces the application process on the B-subscriber site. Lower and upper tester are running in parallel, i.e., in concurrent TTCN they are described as test components. The test components have to perform the following functions: connection establishment on the D channel, QoS negotiation on the D channel, data transmission on the B channels, and connection abort on the D channel.

The HMSC in Figure 5a provides an overall view of the test case behaviour. The functionality is indicated by means of MSC references. Stable testing states[4] are described by means of condition symbols.

The test case starts in the stable testing state disconnected. An ISDN connection setup is performed. This is done by using the ISDN signalling protocol on the D channel. The MSC in Figure 5b shows a refinement of this procedure. After connection establishment when the system is in a connected state, the QoS parameters have to be negotiated.

In order to put the SUT into normal operation during a certain period of time, a normal data transfer has to be performed, i.e., all QoS parameters have to be kept. The behaviour of lower tester controlling and observing the B channels comprises the sending and reception of data packets. Sending a data packet means that a time stamp is generated and transmitted with the data packet to the SUT. This way, the SUT is able to calculate the end-to-end delay from the time stamp received and the local time of the receiver's clock[5].

In the course of exchange of data packets the test components controlling the B channels on the A-subscriber site attempts to increase the end-to-end delay. This can be done by using an additional buffer for all messages to be sent or by producing *incorrect* time stamps. Eventually, the SUT determines that the maximal end-to-end delay is not further guaranteed. Therefore the connection has to be aborted. Thus, the SUT indicates the

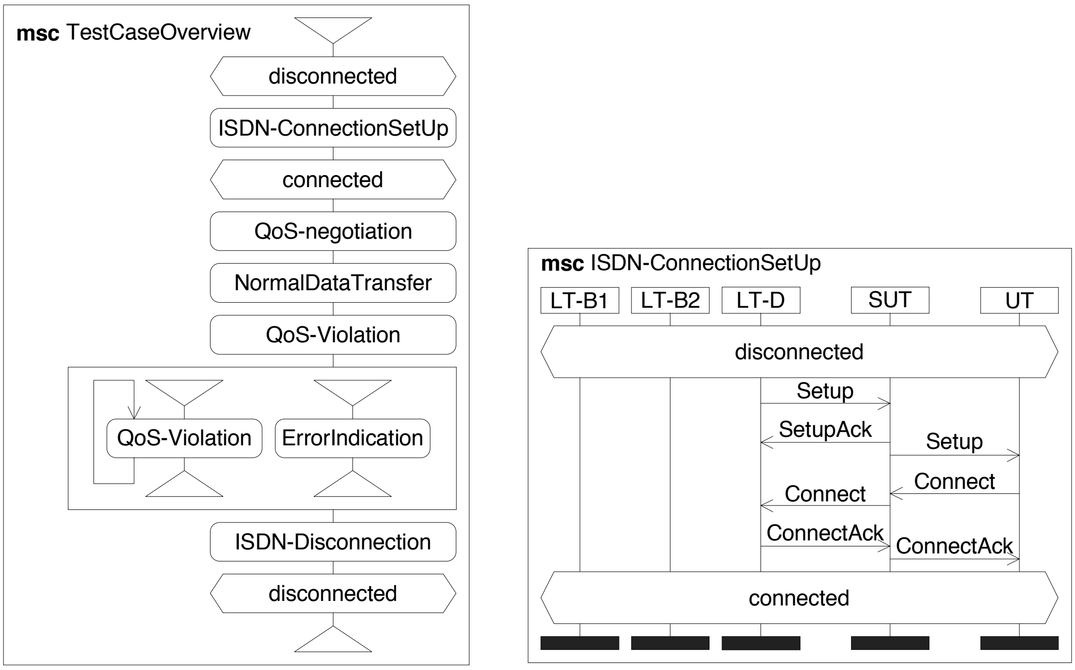
[3] In ISDN the A-subscriber is meant to be the calling party. Thus, the B-subscriber is the called party.

[4] A stable testing state denotes a global system state where the system is stable until a tester process provides the next input.

[5] For simplicity we assume that all clocks are synchronised.

disconnection to the upper tester and sends an abort indication message to the lower tester which controls the D channel. The test case ends in a disconnected state.

The dynamic behaviour of two test components for the provided test case example is partly provided in Figure 6 and Figure 7. The overall view and the communication between all system components is hidden in several tables. Each table describes just part of the behaviour of one test component.



(a) Test case overview (b) Detailed ASP exchange (partially)

Figure 5 MSC description of the ISDN test case example

Test Step Dynamic Behaviour					
Test Step Name: B-LT **Default:**					
Nr	**Label**	**Behaviour Description**	**Constraints Ref.**	**Verdict**	**Comments**
1		CP1?CM1	C_Connected		RECEIVE
2	L1	Start T1			Timer op.
3		? TIMEOUT			TIMEOUT
4		+QoSViolation1			ATTACH
5		L!DATAreq	C_Data		SEND
6		GOTO L1			GOTO
Detailed Comments:					

Figure 6 TTCN test step description for Lower Tester B Channel

Test Step Dynamic Behaviour					
Test Step Name: D-LT **Default:**					
Nr	**Label**	**Behaviour Description**	**Constraints Ref.**	**Verdict**	**Comments**
1		[disconnected]			Boolean
2		+ISDN_ConnectionSetUp			ATTACH
3		[connected]			
4		CP1!CM1	connected		SEND
5		+ISDN_Disconnection			
Detailed Comments:					

Figure 7 TTCN test step description for Lower Tester D Channel

Both, the IN and the ISDN example show that the MSC language is suitable to increase the readability of test case descriptions. HMSCs provide a graphical means for an abstract overall view of the test case behaviour and MSCs can be used to describe more detailed communication aspects, i.e., the concrete message exchange between SUT and test components.

3. GENERATING MSCS FROM TTCN

The second possibility of a combined use of TTCN and MSC is the generation of MSCs from TTCN test cases.

3.1 Simulation instead of mapping

The control flow of a TTCN test case might be very complex and, in case of concurrent TTCN, be distributed over several test components. The different test components co-ordinate themselves by exchanging co-ordination messages (CMs) at co-ordination points (CPs). CPs are meant to be FIFO queues, i.e., the communication is asynchronous like the inter-process communication of SDL. In general, there is no one-to-one relation between the description of CM send events and the corresponding CM receive events. This one-to-one relation is a prerequisite for a direct mapping from TTCN to MSC. Therefore, such a direct mapping cannot be provided.

However, the visualisation of TTCN by means of MSCs can be realised by using a simulation approach. The test case to be visualised has to be simulated, the traces have to be recorded and the trace events can then be mapped onto an MSC.

The entire approach is of course to generate all test event traces leading to a final test verdict and to map these traces onto MSC diagrams. This means that the visualisation of a TTCN test case is represented by a set of MSC diagrams. If the TTCN test case includes loops, the number of traces and, thus, the set of MSCs visualising the test case may be infinite. For these cases only finite prefixes of the traces are considered for visualisation.

Within the set of MSCs visualising a test case three subsets can be identified. Each subset includes MSCs which describe test event sequences leading to one of the three TTCN test verdicts, i.e., we can distinguish MSCs leading to PASS, INCONCLUSIVE, and FAIL verdicts.

The problems related to our simulation approach are complexity due to the TTCN interleaving semantics, implicit complexity and the influence of data on the behaviour description.

Simulation based on interleaving Semantics

The TTCN simulation should be based on the standardised TTCN snapshot semantics. This semantics can be interpreted as an interleaving semantics, i.e. parallelism is described by interleaved sequences of test events.

A problem of interleaved event sequences and the MSC representation is that an MSC describes a partially ordered set of events. Thus, all interleaved sequences of events belonging to the same partial order will be mapped onto identical MSCs. An example taken from Inres [Hog91] may clarify this. Figure 8 and Figure 9 specify the dynamic part of a test case for testing the *Initiator* part of the Inres system.The expected test run for getting a PASS verdict is shown in Figure 10.[6] For readability the run is split up into three MSCs called Setup, DataTransfer and Release. DataTransfer can be interpreted as test body which uses the preamble Setup and the postamble Release. The test configuration (Figure 11) consists of an upper tester (UT) which also is the main test component (MTC), a lower tester (LT) which is realised by a parallel test component (PTC) and an SUT consisting of an Initator entity and an underlying Medium service. The points of control and observation (PCOs) used to exchange data between UT and SUT and between lower tester and SUT are ISAP and MSAP. UT and LT co-ordinate themselves by exchanging co-ordination messages (CMs) at the co-ordination point CM1.

The intention of the test case is to check if the *Initiator* buffers a data package while it waits for the acknowledgement of the data package sent previously. The expected test run is that the UT creates the LT, and after connection establishment (Figure 10b) the UT sends the first data package (DATreq[1]) to the SUT which transmits the package to the LT (DT in Figure 10a). The LT indicates the reception of the first data package via the CM RecDAT to the UT. On reception of RecDAT the UT sends the second DATreq to the SUT and the CM SendDAT to the LT.

[6] Inconclusive and fail cases of the test case are treated in the default behaviour descriptions which are not presented in this paper.

Test Case Dynamic Behaviour					
Test Step Name: InresTestCaseExample **Configuration:** ConfigOne **Default:** OtherwiseFail					
Nr	**Label**	**Behaviour Description**	**Constraints Ref.**	**Verdict**	**Comments**
1		CREATE(LowerTester,PTCDescription)			
2		ISAP!CONreq			
3		ISAP?CONconf			
4		ISAP!DATreq (DATreq.S:=1)	DATreqdef		
5		CP1?RecDAT			
6		ISAP!DATreq (DATreq.S:=2)	DATreqdef		
7		CP1!SendDAT			
8		CP1?CorDAT		(PASS)	
9		ISAP?DISind			
Detailed Comments:					

Figure 8 Inres example main test component (MTC)

Test Step Dynamic Behaviour					
Test Step Name: PTCDescription **Default:** OtherwiseFail					
Nr	**Label**	**Behaviour Description**	**Constraints Ref.**	**Verdict**	**Comments**
1		MSAP?CR			
2		MSAP!CC			
3		MSAP?DT (S:=DT.S)	DTdef		
4		CP1!RecDAT			
5		MSAP?DT [S=DT.S]	DTdef		
6		MSAP?DT [S=DT.S]	DTdef		
7		MSAP?DT [S=DT.S]	DTdef		
8		CP1?SendDAT			
9		MSAP!AK (AK.Nr:=S)	Akdef		
10		MSAP?DT [DT.S = S + 1]	DTdef		
11		CP1!CorDAT			
12		MSAP!AK (AK.Nr:=S+1)	Akdef		
13		MSAP!DR			
Detailed Comments:					

Figure 9 Inres example parallel test component (PTC)

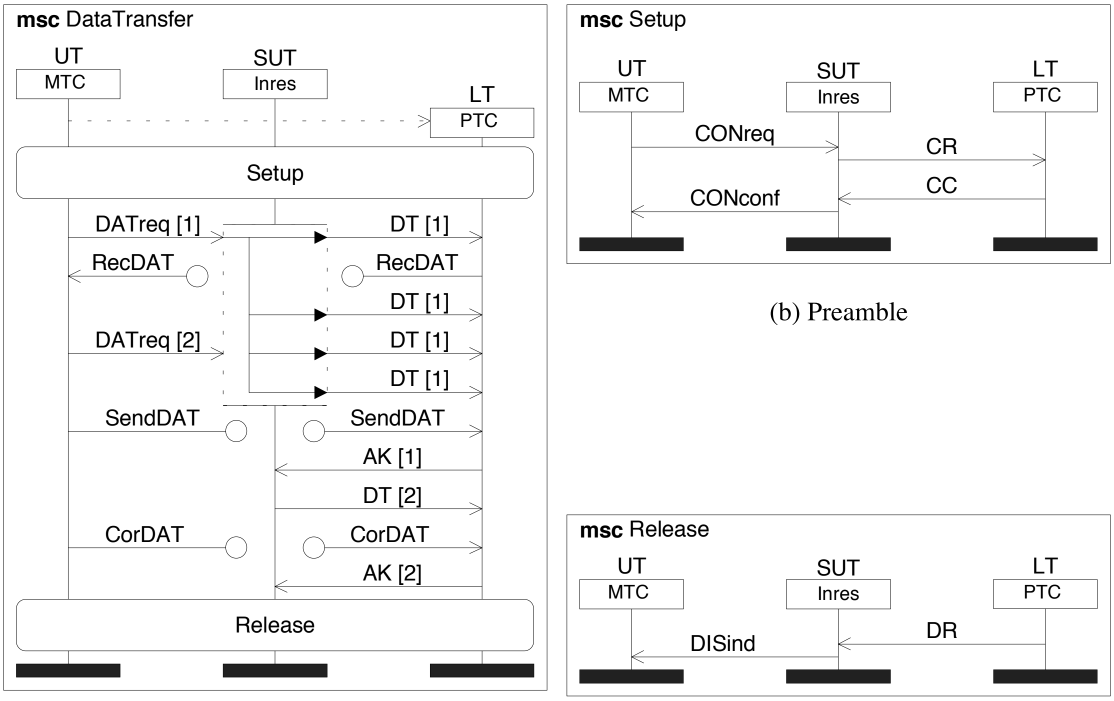
The LT waits until the third repetition of the transmission of the first data package and checks that the second data package was sent during the three repetitions. Then the LT acknowledges the first data package (AK[1]) and consumes the second data package (DT[2]). Afterwards the LT indicates the correct reception of the two data packages to the UT (CM CorDAT) and acknowledges the reception of the second data package (AK[2]). Finally the connection is released (Figure 10c).

By simulation of the presented TTCN, more than 9000 test event traces all leading to a PASS verdict can be generated. A possibility to avoid the explosion of interleaved event traces is partial order simulation as presented in [TGH95] for SDL specifications.

Implicit complexity

For test cases of the same test environment, often only a few things change. To avoid the duplication of identical parts common data types, variables, default constraints descriptions, preambles, postambles, default behavior descriptions, or even complete test components are often collected in a module and bind to the test case before its compilation and execution.[7] The imported definitions and behaviour may be very complex and, therefore, a test case which looks very simple may have a very complex behaviour.

[7] The latest version of the TTCN definition takes this fact into account by allowing definition, import and export of TTCN modules.



(a) Testbody

(b) Preamble

(c) Postamble

Figure 10 Expected test behaviour as MSC

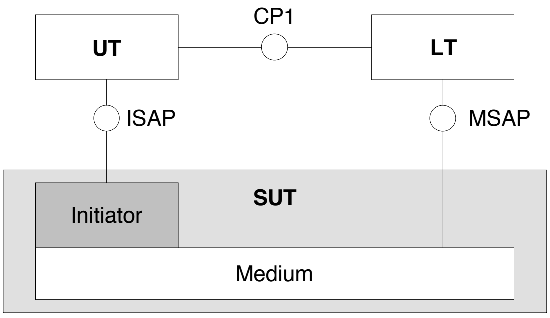


Figure 11 Inres example test configuration

Data influence

TTCN supports the definition and use of test suite and test case variables. Variable values and the values of message parameter[8] may influence the behaviour of test cases. For the simulation of a TTCN test case the values provided by the SUT cannot be predicted. This problem can be tackled by providing values manually or by using a simulator which is able to handle symbolic values.

3.2 MSC representation

Normally, when specifying systems or defining test purposes, the MSC description focuses on the system to be specified or the system to be tested, i.e., system or SUT axes are placed in the middle and environment or test component axes can be found at the borders of the diagram. For the visualisation of TTCN test cases, this view

---

[8] TTCN distinguishes between three types of messages: co-ordination messages (CMs), abstract service primitives (ASPs) and protocol data units (PDUs). CMs are exchanged among test components, ASPs and PDUs are exchanged between test components and SUT.

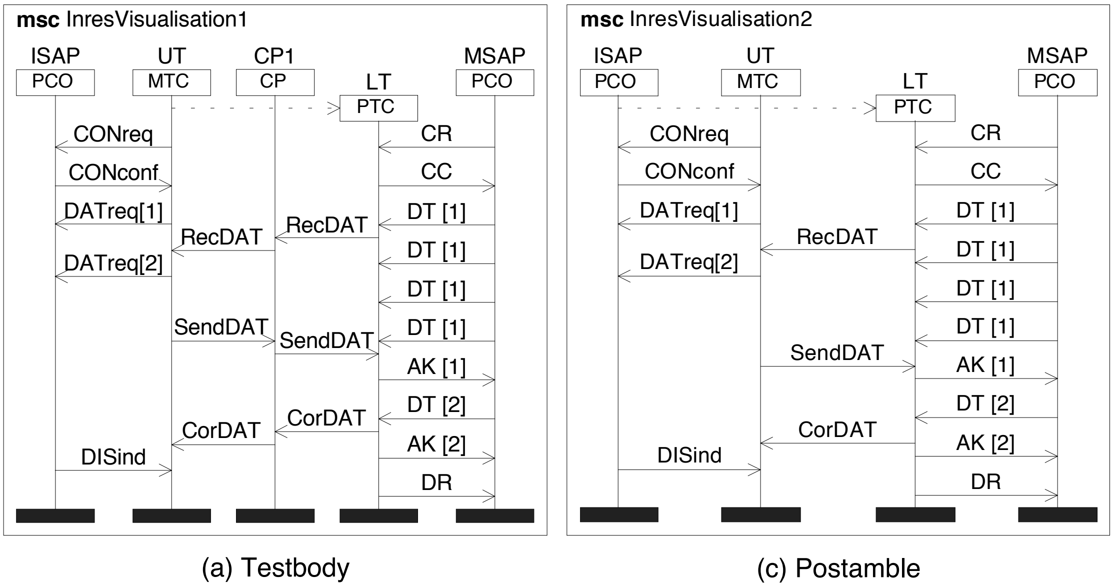


Figure 12 MSC visualisations of the Inres example test case leading to a PASS verdict

has to be changed. A TTCN test case describes the communication among test components at CPs, and between test components and their environment at points of control and observation (PCOs). A test case provides no information about internal structure and communication of the SUT.

Within an MSC visualising a test case, we may represent test components, PCOs and CPs. An MSC leading to a PASS verdict with these components is shown in Figure 12a. The mapping of test event sequence generated by means of simulation onto an MSC is straightforward:

- TTCN CREATE events are mapped onto MSC CREATE
- TTCN send events are mapped onto MSC message outputs (start of a message arrow)
- TTCN receive events are mapped onto MSC message inputs (head of a message arrow)
- Missing MSC message inputs and outputs describe the communication with the test case environment and can be added automatically, because the involved PCOs are mentioned in TTCN send and receive events.
- TTCN assignments are mapped onto MSC tasks.[9]
- TTCN timer operations are mapped onto MSC timer operations (the READTIMER operation has to be mapped onto an MSC task)
- TTCN qualifiers may be described by means of MSC conditions or comments,[9] and,
- Entry and exit points of TTCN tree attachments may be indicated by MSC conditions or comments

The representation of a CP within an MSC might be a little confusing because a CP is just a FIFO queue between two test components and does not provide much new information. For the visualisation we may abstract from CPs as shown in Figure 12b. With the help of tools other views which abstract from certain aspects of a test case description may be generated with the help of tools. An example of such a view might be an MSC which include only axes for the test components (communication with PCOs may be indicated by using tasks), i.e., concentrates on the communication among test components.

4. SUMMARY AND OUTLOOK

We discussed the use of MSCs for improving the readability TTCN test cases. If MSCs are used for test purpose description, i.e., before the TTCN production, commercial tools already exist which support or even automate the generation of TTCN.

Our approach for the generation of MSCs from TTCN test cases is based on the simulation of the test cases. The simulation runs are recorded and can then be mapped onto MSCs. This means, that in general, a TTCN test

---

[9] Due to lack of space assignments and qualifiers are not visualised in Figure 12.

case is visualised by a whole set of MSCs. The problems of our simulation approach are complexity due to the TTCN interleaving semantics, implicit complexity inherited from modules which are common for an entire test suite, and the influence of data and message parameter values on the test case behaviour. Some of these problems can be tackled by using partial order simulation and symbolic values for simulation. The mapping of the simulation runs onto basic MSCs is straight forward. A part of the simulation approach has been implemented in a student thesis [Heg95]. Experiments with the prototype were promising, but also emphasise the complexity of test cases written in concurrent TTCN.

Our future work will focus on solving the problems mentioned above and also on the generation of MSCs with more complex constructs (e.g., inline expressions, MSC references) and HMSCs.

Acknowledgements
The authors would like to thank Beat Koch for proof-reading and valuable suggestions which helped to improve this paper.

5. REFERENCES

[ETSI97] ETSI. Core INAP CS-2; Part 1: Protocol Specification. DEN 03038-1, European Telecommunications Standards Institute, 1997.

[ETSI98] ETSI TC MTS STF 99. *CATG Handbook. DTR/MTS-00030-3*. Temporary document for the ETSI TC MTS meeting in Sophia-Antipolis (France), March 1998, to be published as European Guide in 1998.

[ETSI98a] ETSI. *Core INAP CS-2; Part 3: Test Suite Structure and Test Purposes specification for Service Switching Function (SSF), Specialized Resource Function (SRF) and Service Control Function (SCF)*. DEN 03038-3, European Telecommunications Standards Institute, 1998.

[ETSI98b] ETSI. *Core INAP CS-2; Part 4: Abstract Test Suite (ATS) for Service Switching Function (SSF), Specialized Resource Function (SRF) and Service Control Function (SCF)*. DEN 03038-, European Telecommunications Standards Institute, 1998.

[GHN93] J. Grabowski, D. Hogrefe, R. Nahm. *Test Case Generation with Test Purpose Specification by MSCs*. In: SDL'93 - Using Objects. North-Holland, October 1993.

[GHNS95] J. Grabowski, D. Hogrefe, I. Nussbaumer, A. Spichiger. *Test Case Specification Based on MSCs and ASN.1*. In: SDL'95 - Proceedings of the 7th SDL Forum, Sept. 25 - 29, 1995, Oslo, Norway, North-Holland, September 1995.

[GSHD97] J. Grabowski, R. Scheurer, D. Hogrefe, Z. Dai. *Applying SAMSTAG to the B-ISDN Protocol SSCOP*. In: Testing of Communicating Systems, vol. 10, Chapman & Hall, 1997.

[Hal94] F. Halsall. *Data Communications, Computer Networks and Open Systems*. Addison-Wesley, 1994.

[Heg95] R. Hegi. Visualisierung von Testfällen. Diploma thesis (available only in German), University of Bern, Bern, Switzerland, 1995.

[Hog91] D. Hogrefe. OSI Formal Specification Case Study: the Inres Protocol and Service. Technical Report IAM-91-012, University of Bern, Institute for Informatics, Bern (Switzerland), May 1991.

[IS9646-1:94] ISO/IEC. *Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General concepts*. International Standard 9646-1, ISO/IEC, 1994.

[IS9646-3:96] ISO/IEC. *Information Technology - Open Systems Interconnection - Conformance Testing Methodology and Framework - Part 3: The Tree and Tabular Combined Notation (TTCN)*. International Standard 9646-3, ISO/IEC, 1996.

[MR93] S. Mauw, M.A. Reniers. *An Algebraic Semantics of Basic Message Sequence Charts*. In: The Computer Journal (Special Issue on Process Algebra), 36(5), 1993.

[SEGHK98] M. Schmitt, A. Ek, J. Grabowski, D. Hogrefe, B. Koch. *Autolink - Putting SDL-based Test Generation into Practice*. In: Proceedings of the 11th. International Workshop on Testing of Communicating Systems (IWTCS'98). To appear in September 1998.

[Tan89] A. Tanenbaum. *Computer Networks*. Prentice-Hall, 1989.

[TGH95] D. Toggweiler, J. Grabowski, D. Hogrefe. *Partial Order Simulation of SDL Specifications*. In: SDL'95 with MSC in CASE. Elsevier, 1995.

[Thö94] J. Thörner. *Intelligent Networks*. Artech House, 1994.

[Z.120:96] ITU-T. *Message Sequence Chart (MSC)*. ITU-T Recommendation Z.120, 1996.

[Z.120B:95] ITU-T. *Message Sequence Chart - Algebraic Semantics*. ITU-T Rec. Z.120 Annex B, 1995.