18

Test case generation for ATM protocols using high-level Petri net models

R. Bechtold [a], *G. Gattung* [b], *O. Henniger* [b], *C. Paule* [b]

[a] *Deutsche Telekom, Niederlassung Wiesbaden, Projekt Roland K.-Adenauer-Ring 33, 65187 Wiesbaden, Germany*

[b] *GMD – German National Research Center for Information Technology, Rheinstr. 75, 64295 Darmstadt, Germany*

Abstract

This paper discusses an experience in computer-supported test case generation for ATM protocols, viz. the AAL3/4 protocol (ATM Adaptation Layer Type 3/4) and CLNAP (Connectionless Network Access Protocol). The test cases were generated by means of the test generation tool *GeneTest* based on protocol specifications as product nets, i.e. as a special kind of high-level Petri net.

Keywords

ATM (Asynchronous Transfer Mode), Protocol specification, Test case generation

1 INTRODUCTION

Quality and reliability are factors of growing importance in the highly competitive European market area of communication systems. Only well-tested products can claim to provide the required behaviour and can take over their tasks in a complex communication system. To reach the goal of reliability, a sequence of steps is required, from an unambiguous definition of requirements via conformance tests to interoperability and performance tests.

In the classic area of communication protocols, like X.25, X.400, FTAM, X.75 etc., conformance test suites are available and widely used by manufacturers, users, and independent test laboratories. For new protocols, like the ATM protocols, conformance test suites are hardly available and still need to be developed. Computer-supported test case generation is expected to save time and to cut costs in the development of test suites.

This paper reports about the application of the test generation method presented in [Bau90] to real protocols. The method has been implemented in a tool [Bau91] and has been applied to generate conformance test cases for ATM protocols, in a first stage for the AAL3/4 protocol [I363] and for CLNAP [I364].

The starting point for the test case generation method is a specification of the complete test architecture as a product net [Bur89], i.e. as a special kind of high-level Petri net with n-tuples as tokens and with inhibitor arcs and erase arcs beside the regular arcs. A Petri net is a mathematical model to describe distributed discrete event systems with the advantage to allow very compact descriptions of systems with a large set of states. The states of the system are described implicitly by means of the net model, the initial markings, and the firing rules.

B. Baumgarten et al. (eds.), *Testing of Communicating Systems*

Especially high-level Petri nets with mathematical objects as tokens carrying complex information allow compact system descriptions and are very suitable for specifying communication protocols.

The test case generation method is a method with explicit test purposes, i.e. beside the actual specification additional information about test purposes is required as input to the method. The test specifier has to decide on the test purposes and to translate them manually into initial markings of the net. Then the method ensures that test cases consistent with both the specification and the test purposes are generated. Methods with explicit test purposes offer much flexibility and allow to focus on relevant conformance requirements. On the other hand, they take manual efforts to determine test purposes and to ensure a systematic fault coverage.

The generated test cases are represented in the standardised test description language TTCN [ISO9646].

In Section 2, we shortly discuss the AAL3/4 protocol and CLNAP and their product net specifications that are used as starting point for the test case generation. In Section 3, the test case generation method is explained. Section 4 gives a summary and concluding remarks.

2 PROTOCOL SPECIFICATION

2.1 Functions of AAL3/4 and CLNAP

The AAL of the B-ISDN protocol reference model provides for the adaptation of services provided by the ATM layer to the requirements of higher layers. Its main function is mapping higher layer PDUs into the information fields (of 48 octets length) of ATM cells and reassembling these cells. Depending on the service required by the higher layer, different types of AAL have been defined (types 1, 2, 3/4, and 5). AAL3/4 is for the connection-oriented and the connectionless data transfer service classes. Our study is confined to the message mode service of AAL3/4. The AAL is subdivided into the Convergence Sublayer (CS) and the Segmentation and Reassembly (SAR) Sublayer. The CS in its turn is subdivided into the Common Part Convergence Sublayer (CPCS) and the Service Specific Convergence Sublayer (SSCS). The SSCS protocol depends on the specific AAL user. In our case, with CLNAP above AAL3/4, the SSCS is empty.

The CLNAP provides the connectionless data service in B-ISDN.

2.2 Test architecture

We modelled the test architecture that is applied by Deutsche Telekom for acceptance testing of terminal adapters employed in the area of SMDS/CBDS (Switched Multi-Megabit Data Service/Connectionless Broadband Data Services). The AAL3/4 and CLNAP implementations under test reside in a terminal adapter for connecting LANs to an ATM network (using the HSSI/DXI interface, High Speed Serial Interface/Data eXchange Interface). In the test architecture, the terminal adapter is connected via an ATM network to an ATM test system acting as the lower tester. There is no upper tester in the test architecture. In order to gain some indirect controllability and observability of the upper service boundaries of the IUT, the terminal adapter's LAN output line is looped to its LAN input line.

Depending on the focus of testing, the PCO is placed at different SAPs or interfaces in the test system: immediately beneath the SAR Sublayer for SAR tests, immediately beneath the CPCS for CPCS tests, or immediately beneath the CLNAP for CLNAP tests. As an example, Figure 1 outlines the test architecture for CPCS tests.

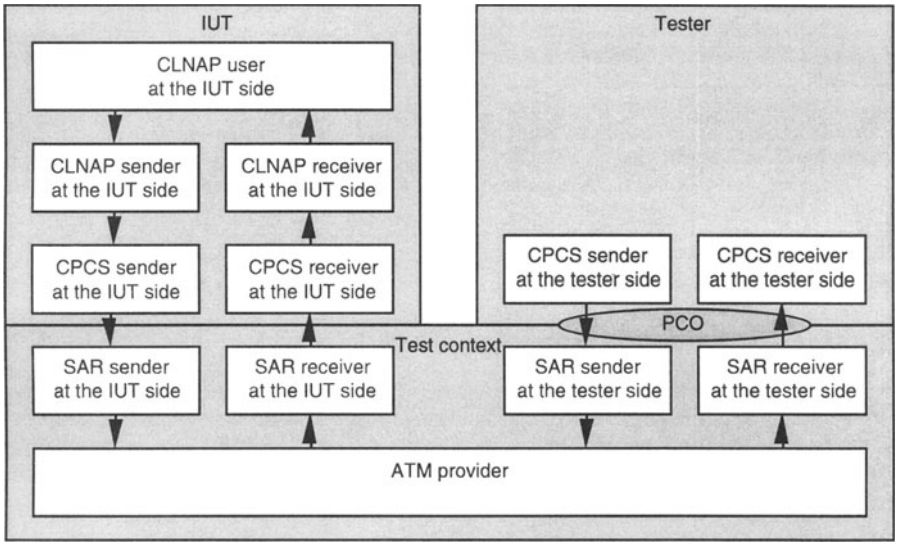

Figure 1 Test architecture for CPCS tests and division into subnets

2.3 Product net model

The complete test architecture including a correct IUT, a test context, and the tester was modelled as a product net. This net is divided into subnets as depicted by the boxes in Figure 1. A subnet is just a graphical unit showing a selected part of the overall product net. Subnets are glued together at places they have in common. All subnets together form the overall product net.

The subnets for SAR, CPCS, and CLNAP sender and receiver at the IUT side model the correct behaviour of the protocols as specified in [I363] and [I364] respectively.

In the CPCS test architecture, the behaviour of the subnets for SAR sender and receiver at the tester side is identical with that of the corresponding subnets at the IUT side, however, supplemented by modelling TIMEOUT and OTHERWISE events. The subnet for the CPCS sender at the tester side models the sending of valid as well as of invalid PDUs by the tester. In the subnet for the CPCS receiver at the tester side, the received PDUs are collected.

The subnet for the CLNAP user at the IUT side is a simplified substitute net modelling only the behaviour visible at the interface. Likewise, the subnet for the ATM provider is a simplified substitute net modelling only the behaviour visible at the ATM SAPs.

We save details of the product net model for the ATM protocols because of limited space. An introduction to product nets as well as a short product net example explaining the applied test case generation method has been given in [Bau90].

3 TEST CASE GENERATION

3.1 Overview

The starting point of the test generation method *GeneTest* is the product net comprising the complete specification of the test architecture, i.e. the protocol to be tested, the test context, and the tester. To generate a test case, the following steps are carried out :

1. The transitions of the product net that are controllable or observable at the PCOs of the chosen test architecture are determined, and the extra information needed for representing these transitions in TTCN is provided.
2. An initial marking for the product net is determined according to a given test purpose.
3. Starting with the initial marking, a complete reachability analysis of the product net is carried out by means of the product net analysis tool *PNM* [GMD94].
4. By means of the test generation tool *GeneTest* [Bau91] the reachability graph is restricted to the occurrences of the controllable or observable transitions. This restriction results in an automaton describing only the behaviour visible at the PCOs. This automaton is converted into TTCN format. At the same time, *GeneTest* generates most of the relevant parts of the declarations and constraints parts of the resulting test suite in TTCN format.
5. The generated test case is edited in order to add verdicts, which are not yet generated automatically. The determination of appropriate verdicts is not defined clearly in [ISO9646]. [Bau95] deals with the formalisation of the verdict assignment.

Figure 2 illustrates the steps of the applied test case generation method. Though generally leading to combinatorial explosion, reachability analysis is feasible in most cases since a separate reachability graph for each initial marking, not a single reachability graph comprising every possible behaviour of the whole system, is computed. However, for some initial markings modelling transmission of many PDUs reachability analysis may take a rather long time.

3.2 Extra information for TTCN representation

According to the chosen test architecture, the test specifier has to determine the transitions that are visible at the PCOs. Occurrences of these transitions are mapped to TTCN behaviour lines in the generated test cases. In *GeneTest*, each transition visible at a PCO is annotated by information necessary for its representation in TTCN:

- the type of TTCN events to which the transition occurrences correspond (SEND, RECEIVE, TIMEOUT, or OTHERWISE events);
- in case of SEND or RECEIVE events: the PCO identifier (if any), the ASP or PDU identifier to be used in the TTCN test suite, a prefix for the TTCN constraint identifiers, the ASP parameter or PDU field identifiers and the way their values are derived from the occurrence variables of this transition (if any);
- in case of TIMEOUT events: the timer identifier;
- in case of OTHERWISE events: the PCO identifier (if any).

3.3 Test suite structure, test purposes, and initial markings

The behaviour of the product net depends on the initial marking of the net. In the sender subnet of the tester, an initial marking models the data to be sent by the tester. In other subnets, some places modelling options are set to an appropriate initial marking.

We manually developed the test suite structure and test purposes for the protocols under consideration according to the guidelines given in [ISO9646]. Care was taken to ensure an adequate coverage of the conformance requirements. Then the test purposes were manually translated into the appropriate initial markings of the product net. The need to define test purposes manually is a weak point of the test case generation method. However, so far methods for automatic generation of test purposes seem to be of limited applicability in practice.

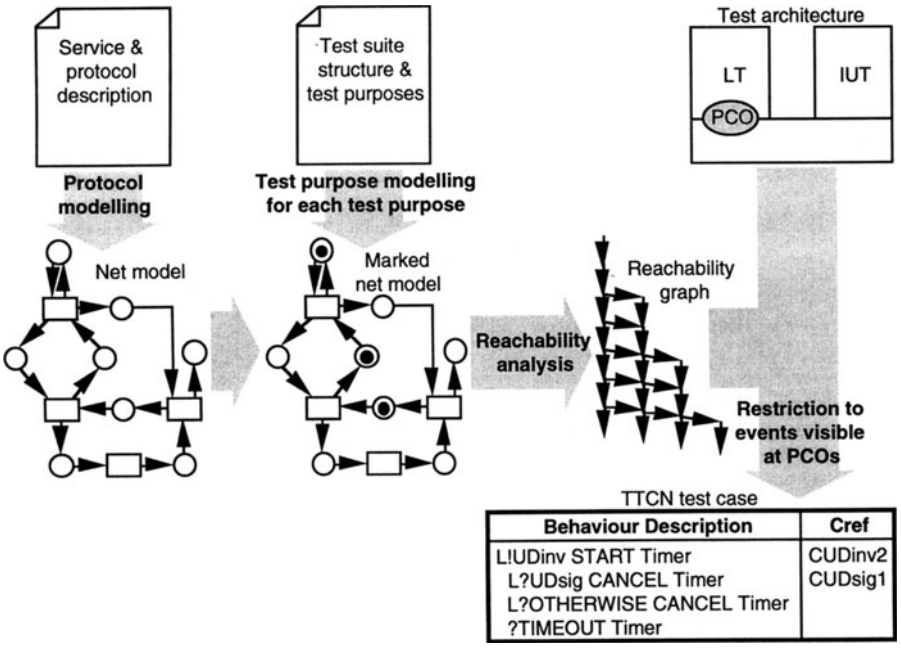

Figure 2 Test case generation from net models

3.4 Manual revision of the test suites

Not all parts of a test suite can be generated automatically from the formal specification, either because relevant information is not contained in the net model or because of limitations of the test generation tool. Therefore, the generated test cases have to be revised manually. The following steps have to be carried out, preferably by means of a TTCN editor tool:

- addition of test suite type definitions, test suite parameter declarations, test case selection expression definitions, test suite constant declarations, test case variable declarations, PCO declarations, timer declarations, test suite operations definitions, and test suite variable declarations (if any) to the declarations part;
- transformation of some constraint values to the correct TTCN format (due to limitations of the current version of the test generation tool);
- assignment of verdicts to the leaf nodes of the TTCN behaviour trees.

3.5 Resulting test suites

The SAR part of the AAL3/4 test suite we produced consists of 104 test cases; the CPCS part has 38 test cases. The CLNAP test suite consists of 76 test cases. Because of the well-known limitations of manually defined test purposes, we cannot declare a certain fault coverage.

To judge the resulting test suites, we compared them with the test suites for the AAL3/4 protocol and for CLNAP that had been developed before by the Roland team of Deutsche Telekom in the classic, manual way. The results of the computer-supported test case generation turned out to exceed our expectations. The resulting test suites meet all requirements

applied by Deutsche Telekom and revealed some flaws in the manually developed test suites. The resulting test suites are available in TTCN format; thus, they allow processing by other tools to create executable test suites.

4 CONCLUSIONS

By means of the test generation tool *GeneTest*, which operates on product net specifications, test suites have been generated for the AAL3/4 protocol and for CLNAP. While still requiring considerable manual efforts to develop the product net specification, to determine initial markings for the net, and to manually revise the generated test suite, the test generation method effectively supports the generation of correct test cases for realistic protocols. The tool-supported test generation takes less time compared to the classic, manual test suite development, delivers correct test cases provably consistent with the specification and the specified test purposes, and thus helps to cut the development costs for test suites.

In the course of the AAL3/4 and CLNAP test generation project, some desirable improvements and extensions have been identified, which shall be taken into consideration for a new version of *GeneTest*, such as parameterisation of test cases and constraints, assignment of verdicts, incorporation of ASN.1 definitions, and TTCN presentation of data values.

It is planned to apply the *GeneTest* tool to other ATM protocols in subsequent projects. The generated test suites will be fed into the standardisation area, as a first step within the ATM Forum.

ACKNOWLEDGEMENTS

The authors would like thank the other participants in the test case generation project, B. Baumgarten, A. Giessler, K. Stöttinger, and H. Wiland, for their helpful comments on this experience report.

REFERENCES

[Bau90] B. Baumgarten, A. Giessler, R. Platten. Test derivation from net models. In J. de Meer, L. Mackert, and W. Effelsberg (editors), *Protocol Test Systems, II*, pp. 141-159, Berlin, Germany, 1989. Elsevier Science Publishers B.V., 1990.

[Bau91] B. Baumgarten, A. Giessler, C. Paule. Testgenerierung mit Produktnetzen – Algorithmus, Implementierung und Benutzeranleitung. Arbeitspapiere der GMD, Nr. 605, GMD, Germany, 1991. The user-manual part is updated regularly.

[Bau95] B. Baumgarten. Timed systems behaviour and conformance testing – a mathematical framework. In A.R. Cavalli and S. Budkowski (editors), *Protocol Test Systems, VIII*, Evry, France, 1995. Participants' proceedings.

[Bur89] H.J. Burkhardt, P. Ochsenschläger, R. Prinoth. Product nets: a formal description technique for cooperating systems. GMD-Studien, Nr. 165, GMD, Germany, 1989

[GMD94] Produktnetzmaschine. User manual, GMD, Germany, 1994.

[ISO9646] International Standard ISO/IEC 9646: Information technology – Open Systems Interconnection – Conformance testing methodology and framework. 1991.

[I363] Revised ITU-T Recommendation I.363: B-ISDN ATM Adaptation Layer (AAL) Specification. 1993.

[I364] ITU-T Recommendation I.364: Support of broadband connectionless data service on B-ISDN. 1994.