Verification by abstraction as a preamble for interoperability test suite generation

Pierre de Saqui-Sannes[a,b], Jean-Pierre Courtiat[b] and René Casadessus[c]

[a] ENSICA, Département DFR/MI, 49 avenue Léon Blum, 31056 Toulouse Cédex, France

[b] LAAS du CNRS, Groupe OLC, 7 avenue du Colonel Roche, 31077 Toulouse Cédex, France

[c] EDF, Direction des Etudes et Recherches, 6 Quai Watier, 78400 Chatou, France

Abstract

In [2], we proposed a verification by abstraction approach for protocol specifications written in Estelle*, an enhanced Estelle with a rendezvous mechanism. The protocol's reachability graph is labeled with occurrences of service primitive exchanges, and minimized to a quotient automaton using observational equivalence. In this paper, we indicate how interoperability test suites can be derived from the quotient automaton. We use a novel approach introduced by Drira in [1] for Labeled Transition System. Abstract test suites are generated, categorized in Must and May tests, and translated to TTCN. A canonical tester is also derived from the quotient automaton; it anticipates on the computation graph of the real tester to be developed. The proposed methodology is illustrated on the Manufacturing Message Specification protocol (MMS - IS 8506).

Keyword Codes: C.2.4, D.2.1, D.2.5, F.4.3.
Keywords: Network protocols, Specifications, Testing and Debugging, Formal Languages.

1. INTRODUCTION

Formal Description Techniques (FDTs) have been claimed to offer a formal support for automating the entire protocol design trajectory within a unified framework. There is by a contrast a lack of FDT usage report indicating that a formal description contributed to both early detection of design errors and test sequence generation. Reasons are particularly to be sought in the separate development of verification and testing techniques.

This paper tackles the problem of generating interoperability test suites relying on a verification by abstraction approach that was proposed in [2] for protocols specifications written in Estelle* [3], an enhanced Estelle with a rendezvous mechanism. The reachability graph of the protocol specification is labeled with occurrences of service primitive exchanges, and minimized to a quotient automaton using observational equivalence [4]. The extension discussed in this paper relies on Drira's approach for deriving abstract test suites from a Labeled Transition System (LTS) [1]. We show how interoperability test suites can be derived from a service quotient automaton that is observationally equivalent to the labeled reachability graph of a protocol specification written in Estelle*. The generated abstract test cases are classified in Must and May tests, and translated to TTCN [6]. From the service quotient automaton, a canonical tester is also derived following the approach proposed in [5].

The paper is organized as follows: Section 2 gives the intuition behind the use of the verification by abstraction approach as a "preamble" for interoperability test suite generation. Section 3 outlines Drira's approach to deriving test sequences and a canonical tester from an LTS. The approach is implemented by PSTV, a Package for Symbolic Testing and Verification which has been linked with the Estelle* verification tool ESTIM [7]. Examples of test sequences and canonical tester are given in Section 4 for two elementary services of the

Manufacturing Message Specification protocol (MMS) [8][9]. Our testing approach is compared to related work in Section 5. Future work directions are finally given in Section 6.

2. A UNIFIED FRAMEWORK FOR VERIFICATION AND TESTING

Estelle has a "glass-box" description style, and models a specification as a "closed world". Conversely, FDTs such as LOTOS and LTSs model a system which can interact with its surrounding environment, and support therefore the application of "black box" verification and testing techniques. The latter have been investigated a synchronous run of the system and its environment in mind. In this section, we show that similar conditions can be obtained thanks to the rendezvous extension of Estelle*.

2.1. Labeled reachability graph

Reachability analysis has come out as the most popular verification technique in the realm of state/transition models. It works under the assumption that the modeled system has a finite behavior, and can be applied to Estelle* specifications under the following three conditions: (i) FIFO queues are arbitrarily bounded by the designer; (ii) the specification does not contain any infinite loop statement; and (iii) the configuration of module instances and linked interaction points is not modified at runtime, which means that verification works for a static specification. An extension of usual reachability analysis was proposed in [2], so the problem of verifying an Estelle* specification against its expected properties can be boiled down to the problem of observing synchronized interaction exchanges between an "observed world" and its environment.

For that purpose, a partition is defined over the module instances of an Estelle specification as a function P: M $\rightarrow$ {observed-world, environment} where M denotes the set of module types declared within the specification. For simplicity, if several instances of the same module are to be created, all of them belong to either the observed world or the environment.

Verification will thus consist in observing those interactions, referred to as observable events, which are exchanged across the boundary defined by P. By contrast, a change in the modules' internal variables or a communication between two modules instances located inside the observed world are indistinguishably handled as internal events. Transitions in the reachability graph are accordingly labeled by either an observable event identifier or τ, respectively.

The labeled reachability graph (LRG) of a *static* Estelle* specification is a quadruple LRG = (S, Σ, Δ, s_0) where

- S denotes the countable number of global states of the specification.
- Σ is a countable set of observable events, included in the set of interaction exchanged by pairs of module instances connected via external interaction points. Thus we have $\Sigma \subseteq X \times EIP \times \{"!", "?"\} \times INT$ where
  - X denotes the set of module instances,
  - "!" and "?" respectively denote an emission and a reception, from and by, the module instance belonging to the observed world,
  - EIP denotes the set of external interaction points of all the modules instances, and
  - INT denotes the set of interactions declared in the specification, *i.e.* the union of channel declaration parts.
- $\Delta \subseteq S \times (\Sigma \cup \{\tau\}) \times S$ denotes the set of labeled transitions of the graph; a label is either an observable event identifier of the form (module instance, interaction point, "!" or "?", interaction) or τ, the symbol which denotes any internal event.
- s_0 denotes the initial state of the Estelle* specification, *i.e.* the global state obtained at completion of the static initialization phase, that is, when each module instance executes its "initialize" part.

The intention behind the generation of a labeled reachability graph is to derive an abstract view of the Estelle* specification using minimization techniques, or "projections", that had originally been developed for LTSs [4].

2.2. Verification by abstraction

Verification by abstraction uses as input an LTS and outputs a quotient automaton reflecting an abstract view of the modeled system. The approach uses the previous distinction between an observed world and its environment. All observable events are preserved. Internal events, or τ-events, are meant to be discarded; some of them are nevertheless preserved, depending on non deterministic choices within the observed world. The theoretical background was developed for a parallel composition operator *à la* LOTOS [12]. In Estelle*, synchronous communications between the observed world and its environment can be achieved using the rendezvous extension. A LOTOS-compatible composition is obtained by connecting external interaction points of appropriate module instances.

We limit therefore the set of observable events to those interactions exchanged by two module instance which rendezvous. FIFO queued communication become *de facto* invisible. Such a limitation is not too strong in the context of protocol verification. Rendezvous interactions between adjacent protocol layers are indeed recommended in order to keep a Service Access description as abstract as possible. Conversely, a FIFO-queued interface would anticipate on a particular implementation [3].

As shown by Figure 1, protocol verification consists of three steps:

1) Observable events are limited to occurrences of service primitives exchanges by partitioning the specification modules in two groups. A typical example is given on Figures 1 and 3: two protocol entities and their underlying service are located inside the observed world whereas the service users are seen as the environment. (Note: both users are modeled as rendezvous acceptors [7]).
2) A Labeled Reachability Graph is generated using ESTIM [7] or VESAR [13] (Figure 2). All the specification components are handled, particularly major states, Pascal variables, queues, interaction parameters and "delay" clauses. The subsequent minimization of the LRG ignores the nodes content and uses transition labels only.
3) A quotient automaton, at least trace-equivalent to the LRG, is derived. Observational projection may further preserve certain internal events related to internal, non deterministic choices.

The alphabet of the quotient automaton, and consequently the characterized service, depends on the selected partition.

2.3. Partition over the specification modules

Figure 3 depicts the partition needed to obtain the global service provided by a communication protocol, such as the Network or Transport protocol. (N) U1 and (N) U2 users play a symmetric role in regards of the (N) service, and both belong to the external environment. The three other modules belong to the observed world (dashed area).

Figure 4 considers situations in which the two users play asymmetric roles. (N) C and (N) S respectively identify a Client and a Server using the (N) service. Such a distinction is common in the structure of Application layer protocols, such as FTAM (File Transfer Access and Management) and MMS (Manufacturing Messaging Specification). The environment is defined by the Client user modules. Other modules belong to the observed world. For certain Application layer protocols, such as MMS, the partition depicted by Figure 4 could be recursively generalized, so the notion of subordinated servers can be considered. The latter provide the service expected by (N) C, when (N) S requires other services from one or several of its subordinated servers. In such a situation, all the server modules would belong to the observed world.

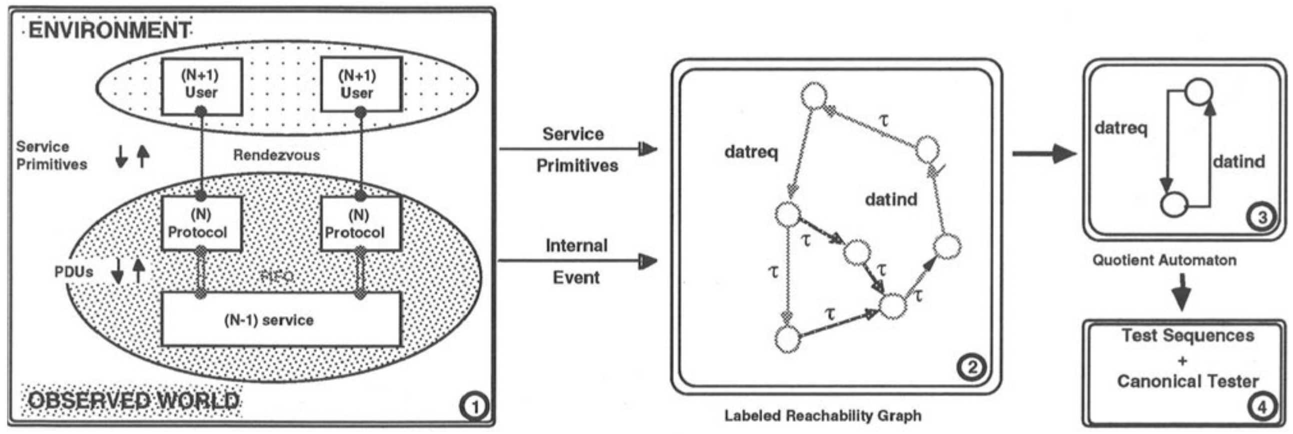

Figure 1. The Estelle* methodology

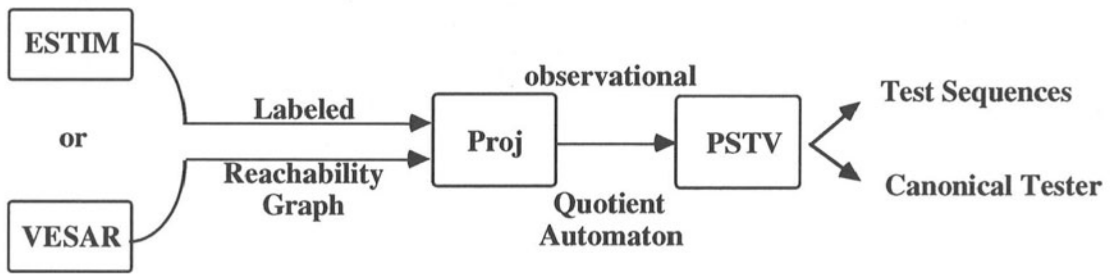

Figure 2. The toolset which implements the Estelle* methodology

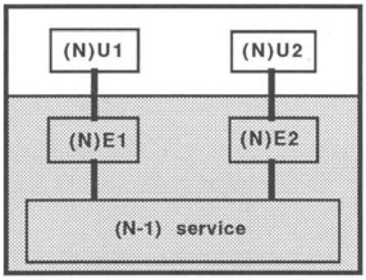

Figure 3. Partition for two (N) service users playing symmetric roles

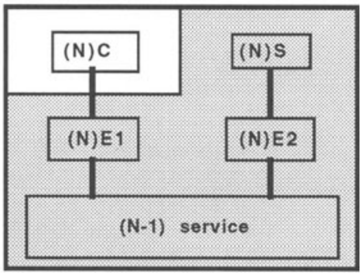

Figure 4. Partition for two (N) service users playing asymmetric roles

The partition depicted by Figure 5 may be extended to situations where a server, *e.g.* a file server, is concurrently accessed by several Client users. Only the Client user modules belong to the external environment. Similarly, this partition could be generalized to consider subordinated servers.

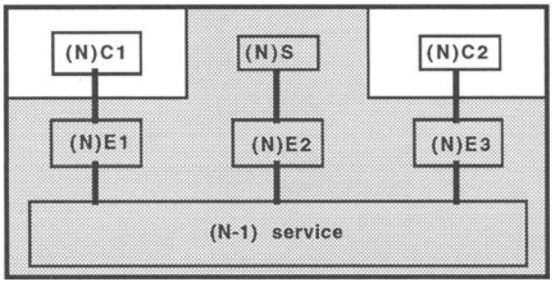

Figure 5. Asymmetric roles among $n \geq 2$ users of the (N) service

Note: Figures 4 and 5 show that a verification methodology originally developed for data transfer protocols [2] can easily be extended to Application layer protocols. The difficulty is not in verifying an elementary service, but a combination of elementary services, each being implemented by a rather simple protocol which is not worth verifying for itself [14].

3. ABSTRACT TEST SUITE GENERATION

As opposed to conformance testing methods based on a single entity model, interoperability testing needs to produce a global model of communications between two entities [17]. We thought therefore of deriving interoperability test suites from the Labeled Reachability Graph of an Estelle* specification. The testing framework defined in [1] makes it possible to use a global service quotient automaton as input in lieu of the complete LRG.

3.1. Test sequence derivation based on refusal graphs

A refusal graph [1], denoted, RG is defined as a deterministic LTS $(G,\Sigma,\Delta,g_0,Ref)$ where

Ref : $G \rightarrow P(P(\Sigma))$ is a mapping which defines for each state, the sets of actions that may be refused after the sequence leading to this state. To avoid redundancy, refusal sets must be minimal w.r.t. set inclusion. And to avoid describing imaginary systems, only refused parts of the output set are considered.

Figure 6 exemplifies the transformation from an LTS to its refusal graph. The latter was used for deriving the test cases depicted on the right part of Figure 6.

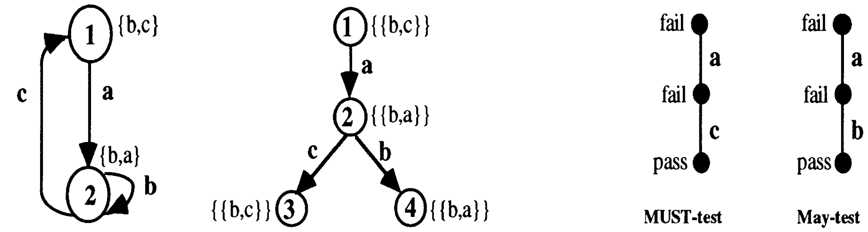

Figure 6. A sample LTS, its refusal graph and derived test cases

The benefits of introducing the refusal graph concept include an operational procedure for implementing the concept of canonical tester introduced in [12]. A canonical tester has the same traces as the original specification and its synchronous run with the IUT must not lead to a deadlock situation. In [5], a canonical tester is simplified relying on the state space of the refusal graph in lieu of the - possibly infinite - trace set of the LTS .

Drira's approach further enables the derivation of test sequences from a quotient automaton which is observationally equivalent to the original LTS. Test sequences are identified from paths of maximal length in the refusal graph, and categorized in MUST and May tests [15], respectively.

We further model test cases as particular LTSs $(S, \Sigma, \Delta, s_0, v)$ where $v: S \rightarrow \{Pass, fail\}$ is a verdict function which will be used to assign verdicts to test runs [16]. The notation is used in the two right most schemes of Figure 6.

3.2. Testing architecture

3.2.1. Interoperability testing

Previous section discussed test generation from a basic LTS, and did not address the semantics of the transition labels. From the definition in Section 2.1, we can expect that test sequences derived from an Estelle* specification reachability graph will be made up of a 4-uple (module instance, interaction point, emission or reception symbol, service primitive). The testing architecture can be related to the verification one. For example, Points of Control and Observation in the testing architecture [17] correspond to those interaction points in the Estelle* specification which are connected to a channel crossed by the partition. In particular, PCOs have SAP interaction points as counterpart in the Estelle* architecture.

The testing architecture closely depends on the selected partition P. Figure 7 depicts the testing architecture corresponding to the "global service" partition suggested by Figure 3. A global tester is assumed to be able to access both Implementations Under Test using synchronous communications. Similarly the "local service" partition depicted by Figure 4 leads to the testing architecture depicted by Figure 8.

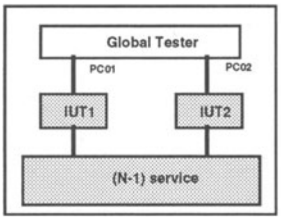

Figure 7. Global tester (interoperability)

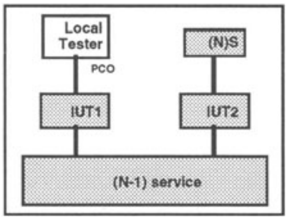

Figure 8. Local tester (interoperability)

3.2.2. Extension to conformance testing

The partitions presented so far were defined with service observation in mind. In particular, no assumption has been made on the communication mechanism used between the protocol entities and the underlying (N-1) service; PDUs exchanges are handled as "internal events" anyway. For a datagram (N-1) service, a FIFO queued communication is to be preferred.

For an extension to conformance testing (Figure 9), we need to observe both type of exchanged interactions: service primitive and PDUs. A possible partition is shown by Figure 10. Such an approach is valid if the (N-1) service is accessed via rendezvous interaction points.

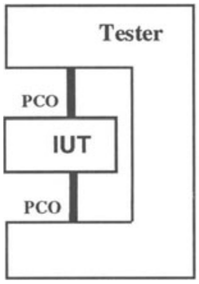

Figure 9. A conformance testing architecture

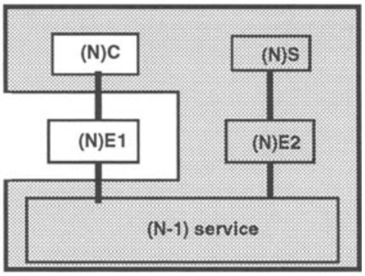

Figure 10. Partition for conformance testing (rendezvous at both the upper and lower interfaces of E1)

A generalization of the proposed approach to FIFO queued channels may for instance rely on the asynchronous testing framework discussed in [16].

3.3. Translation of interoperability test suites to TTCN

Each test case derived from either the Labeled Reachability Graph or its observationally equivalent quotient automaton contains test cases defined as a 4-uple: (module instance identifier, interaction point, "!" or "?", interaction). Emission and reception symbols have to be inverted to reflect the point of view of the tester, and not that of the module instance located inside the observed world (this was the case in the LRG). In previous sections, we suggested that interaction point identifiers can serve as PCO identifiers for the tester. We thus obtain test cases (PCO, "!", interaction) or (PCO, "?", interaction) as shown in Figure 16.

Again, the derived test cases are obtained for a synchronous tester. Their translation to the standardized notation TTCN captures an additional problem: TTCN assumes that the IUT and the tester communicate asynchronously. A similar problem is reported in [18] dealing with test sequences generated from LOTOS processes. [18] proposes to replace each test step in the generated sequence by a parameterized test step whose definition includes a new primitive "Test_accept_RDV" and two parameters, a LOTOS gate and an event identifier, respectively.

A similar solution can be applied to Estelle* having interaction points instead of gates, and interactions in lieu of events. A test step PCO?int or PCO!int is then translated as follows:

```
PCO ! begin_test_rendezvous (interaction)
  START rv_timer
    PCO ? end_test_rendezvous
      CANCEL rv_timer
      ? TIMEOUT
```

4. CASE STUDY: THE MMS PROTOCOL

The Manufacturing Message Standard protocol offers 84 communication services for manufacturing floors. MMS services include down- and uploading of programs and data, remote control, semaphores and status report generation. The standard was developed as part of the MAP/TOP architecture.

Robots and manufacturing equipments in general have a slow time response compared with communication delays supported by today's networks. Therefore MMS has a special feature not usually available with other Application protocols: a service can be canceled upon user request. The associated *Cancel* service was shown to be incorrectly specified by the IS 8506 standard: a unspecified reception error was detected using reachability analysis. A proven correct protocol machine is proposed in [30]. In this paper, the Estelle* methodology is illustrated on an isolated confirmed service, and the testing phase is detailed for the *Reject* service which handles protocol error situations.

4.1. Verification architecture

An isolated MMS confirmed service can be modeled within the Estelle* architecture depicted by Figure 11. The protocol machines depicted by Figure 13 lie in the *Requester* and *Responder* module instances, respectively. Their respective users *Client* and *Server* are modeled as rendezvous acceptors (for both emission and reception).

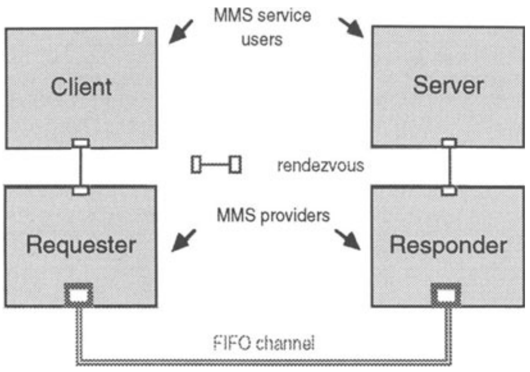

Figure 11. Architecture of the MMS specification

The Estelle* specification relies on a client/server architecture. Rendezvous communication is used at the service access interface. The Presentation service is presumably reliable, and consequently it is modeled by a FIFO queued channel.

In practice, several elementary MMS services, such as *Write Variable* and *Information Report*, can be used in parallel. Both the Requester and Responder modules have then to be refined in two or several submodules, each being dedicated to a particular elementary service [14]. Interactions between MMS services are further discussed in [30].

4.2. Testing Architecture

There is no standardized interoperability testing methodology. In Figure 12, we propose an architecture which assumes that a global tester can access both the MMS requester and Responder implementations, using two Point of Control and Observation, PCOc and PCOs, on the client and server side, respectively.

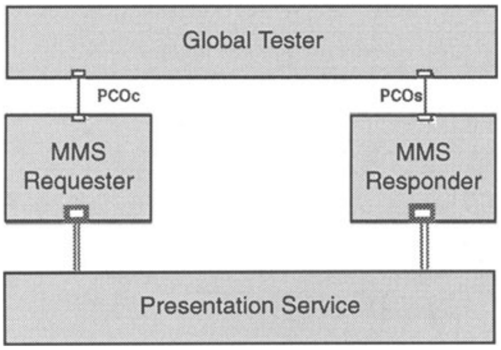

Figure 12. Testing architecture

In the following sections, we present test sequences that have been obtained for the global tester architecture depicted by Figure 12, and from a quotient automata computed using a "global service" service partition.

4.3. Isolated confirmed service

The MMS standard defines 81 confirmed services among 84. The protocol machines in Figure 13 refer to a generic service x, such as *Read Variable* and *Write Variable*. We use "req", "ind", "rsp" and "cnf" to denote a request, an indication, a response and a confirm, respectively. Suffixes + and - denote a positive and negative response or confirm, respectively. Symbols "?" and "!" respectively denote an emission by / a reception from one of the two MMS service providers.

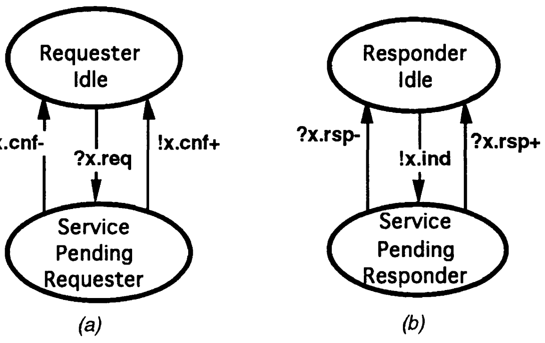

Figure 13 - MMS protocol machines (a) Requester and (b) Responder

The verification by abstraction approach has proven useful for teaching. Indeed it helps clarifying the difference between the protocol machines (Figure 13) and the provided service (left part of Figure 14).

For an elementary confirmed service, there are two MUST sequences derived from the quotient automaton.

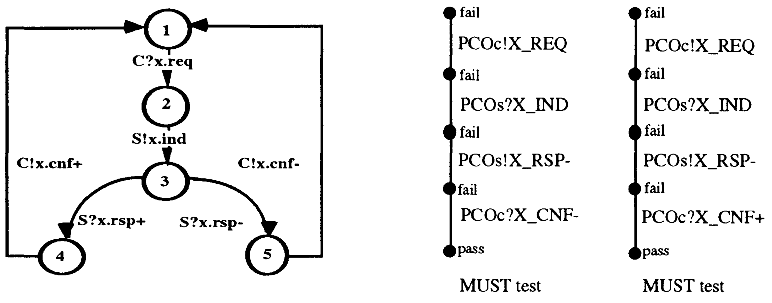

Figure 14. Quotient automaton and test sequences for an isolated confirmed service

4.4. Confirmed service with Reject

4.4.1. Protocol machines

A *Reject* service can be used by each of MMS providers which needs to report a protocol error. A Reject_indication primitive is transmitted to the relevant user (Client or Server). There are two situations in which the *Reject* service can be invoked.

(1) Requester (Figure 15a) receives an erroneous service request after entering the "Service Pending Requester" state, and
(2) Responder (Figure 15b) is in "Responder Idle", and receives an erroneous response via a service primitive which may be either negative or positive.

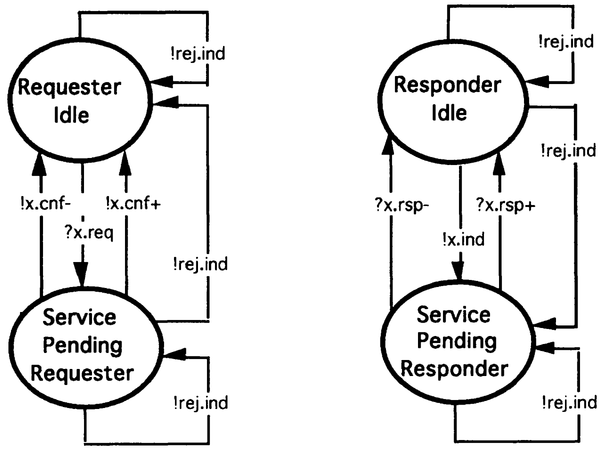

Figure 15. MMS Requester (a) and MMS Responder (b) with Reject

The reachability graph of the Estelle* specification has 8 states and 13 edges and was labeled using a "global service" partition that puts the Client and Server modules into the environment whereas the MMS Requester and Responder lie in the observed world. The trace-equivalent quotient automaton has 5 states and 9 arcs. The observation-equivalent quotient automaton has 7 states and 12 arcs. The difference between the two quotient automata stems from internal events that corresponds to protocol error situations which are modeled relying on non determinism.

4.4.2. Interoperability test suites

For a confirmed service with *Reject*, we obtained a test suite made up of five test cases. PCOc and PCOs are the Point of Control and Observations shown by Picture 12. A Reject Indication is denoted REJ_IND.

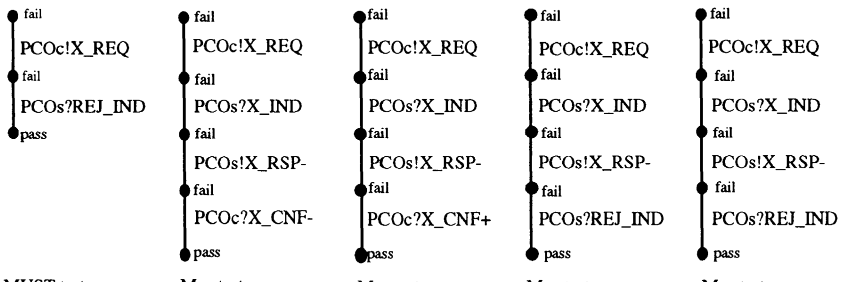

Figure 16. Synchronous test cases for a confirmed service with reject

The transformation discussed in Section 3.3. can be used to translate the above synchronous test cases to TTCN.

4.4.3. Canonical tester

Besides the test sequences depicted by Figure 16, we also derived the canonical tester depicted by Figure 17.

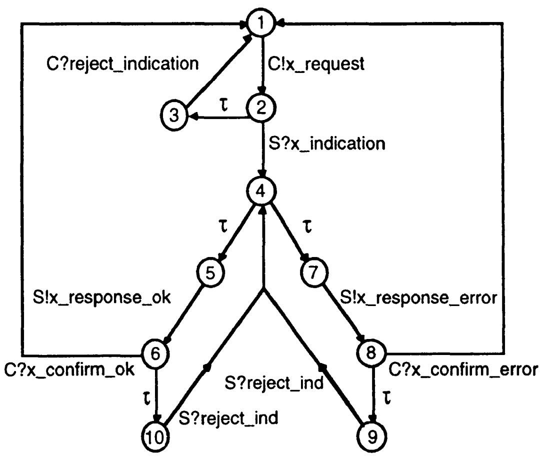

Figure 17. Canonical tester for a confirmed service with Reject

In Figure 17, events noted as τ are in fact associated with non deterministic choices (see, *e.g.*, states 2, 4, 6 and 8 in Figure 17). A possible interpretation of τ-arcs is as follows:

- From State 2, the tester executes an internal action (τ-arc) before accepting a reject_indication primitive. That τ-arc can be interpreted as follows: if the tester does not receive a x_indication primitive within a "prefixed time interval" then it must receive a reject_indication primitive. Internal event τ can thus be interpreted as a TIMEOUT clause of TTCN.

- State 4 is followed by two τ-arcs preceding a non deterministic choice between two possible outputs, a positive and a negative confirm, respectively.

5. RELATED WORK

Test sequence derivation from an Estelle specification has been the subject of several papers cited in reference, and usually relies on either of the three following approaches:

(1) A simplified model is extracted from an Estelle module in order to apply various techniques developed for test sequence derivation from FSM models [19] [20] [21].

(2) An Estelle specification is translated into an intermediate formalism, such as state-charts [22], or rewritten in either a normal form [28] or a well-formed fashion [23] in order to permit application of a particular test generation procedure.

(3) An Estelle specification is directly processed by a tool built upon a simulator. For example TVEDA [24] implements an expert conception based on ad-hoc procedures in use within testing standardization bodies, and has been extended to generate both conformance and interoperability test suites [25].

Test generation methods based on either (1) or (2) usually separate the control flow of the protocol machine from the data part. This usually leads to undesirable sequences that need to be discarded [26]. This may happen if test sequences are generated based on "when" clause receptions, and ignoring parameter values tested by "provided" clauses. Conversely, such a situation is unlikely to happen if test derivation relies on simulation or reachability analysis of the complete specification, as it is suggested in the paper.

Using reachability analysis as the starting point for test generation, we overcome a severe limitation mentioned for example in [31]: the ESTIM-PSTV tool chain *is not* limited to testing a single module.

Conversely, the TESTGEN tool described in [31] has an advantage from its support of an enhanced Estelle with ASN.1 data types. Pascal data structures used in ISO Estelle and Estelle* are too restrictive for Application layer protocol testing in general. For the MMS protocol discussed in the paper, service primitives have simple parameters, such as a variable identifier, that can be modeled as characters or strings. To include parameter testing into our approach, we suggest the following solution: each abstract test case is duplicated as many times as requested for each service primitive parameter to be tested at its minimum and maximum values (obtained from type definitions).

6. CONCLUSIONS AND FUTURE WORK DIRECTIONS

Starting from an enhanced Estelle with a rendezvous mechanism, we proposed to create an environment inside an Estelle* specification so as to reuse verification results originally obtained in the realm of Labeled Transition Systems and LOTOS. The verification by abstraction approach first experienced with the ESTIM tool has been adapted to the industrial tool VESAR [13]. It can be categorized as a "black box" approach which has been proven of high interest for verifying a protocol layer specification against its expected service.

The paper discussed an extension of verification by abstraction for deriving abstract test suites. Service primitive exchanges are considered as the one observable events in the Labeled Reachability Graph of an Estelle* specification. The quotient automaton obtained using observational equivalence therefore characterizes the provided service and contains the information requested to derive abstract interoperability test suites. The proposed testing approach relies on Drira's work on deriving test sequences and a canonical tester from an LTS model [29]. MUST and May tests are separated, which can be the starting point for test selection w.r.t. economical constraints.

Future work directions include an application of the proposed approach to conformance testing, which implies to investigate equivalence relations compatible with FIFO queued communications. We also consider the inclusion of fault models into the MMS specifications.

ACKNOWLEDGEMENTS

We would like to thank Khalil Drira for fruitful discussions on testing theory and for making available the tool, PSTV, to us for use. Contributions from Vitorio B. Mazzola and Luiz F. Rust da Costa Carmo are also acknowledged. This research has been supported by the Direction des Etudes et Recherches of Electricité de France.

REFERENCES

1. K. Drira, “The refusal Graph: a Tradeoff between Verification and Test", Proceedings of the 6th International Workshop on Protocol Test Systems (IWPTS'93), O. Rafiq (ed.), Pau (France), September 1993, pp.301-316.
2. P. de Saqui-Sannes and J.-P. Courtiat, "From the Simulation to the Verification of Estelle* specifications" in S.T. Vuong (ed.), Formal Description Techniques II (FORTE'89), North Holland, 1989.
3. J-P. Courtiat, "Estelle*: A powerful dialect of Estelle for OSI protocol descriptions" in S. Aggrawal and K. Sabnani (eds), Protocol Specification, Verification and Testing VIII, Atlantic City, June 1988, North-Holland.

4. K. Drira and P. Azéma, Verifying Communication Protocols via Testing Projections" in Proc. of AMAST'93, Enschede (The Netherlands), June 1993.
5. K. Drira, P. Azéma and F. Vernadat, "Refusal Graph for Conformance Tester Generation and Simplification" in A. Danthine, G. Leduc and P. Wolper (eds.), Protocol Specification Testing and Verification XIII, North Holland, pp.257-272.
6. ISO IS 9646, "OSI Conformance Testing Methodology and Framework, Part 3: The Tree and Table Combined Notation".
7. J. P. Courtiat, P. de Saqui-Sannes, "ESTIM: An Integrated Environment for the Simulation and Verification of OSI protocols specified in Estelle", Computer Networks and ISDN Systems, Vol.25, N°1, July 1992, pp.83-98.
8. ISO/IS 9506 - Part 1. "Manufacturing Message Specification - Part 1 - Service Definition", International Standard, August 1988.
9. ISO/IS 9506 - Part 2. "Manufacturing Message Specification - Part 2 - Protocol Specification", International Standard, August 1988.
10. IS 9074, "Estelle, a formal description technique based on an extended state transition model", December 1988.
11. Son T. Vuong, "FDT Support Tools for Protocol Development: a Survey", in M. Diaz and R. Groz (eds.), tutorials of FORTE'92, Peirros-Guirrec (France), 1992.
12. E. Brinksma, "A theory for the derivation of tests" in S. Aggrawal and K. Sabnani (eds), Protocol Specification Testing and Verification III, North-Holland, 1988.
13. B.Algayres, V. Coelho, L. Doldi, H. Garavel, Y. Lejeune and C. Rodriguez, "VESAR: a Pragmatic Approach to Formal Specification and Verification", Computer Networks and ISDN Systems, Vol.25, February 1993, pp.779-790.
14. P. de Saqui-Sannes, J.-P. Courtiat, "Evaluation de la méthodologie Estelle* pour le test d'applications réparties", Rapport de contrat, Convention de recherche EDF-LAAS-ENSICA, décembre 1993 (in French).
15. R. de Nicola and M.C.B. Hennessy, "Testing Equivalences for Processes", Theoretical Computer Science, 34:83-133, 1984.
16. L. Verhaard, J. Tretmans, P. Kars and E. Brinksma, "On Asynchronous Testing", in G.von Bochmann, R. Dssouli and A. Das (eds.), Proceedings of IWPTS'92, Montreal, September 1992.
17. O. Rafiq, "Le test d'interopérabilité des protocoles", in O. Rafiq (réd.), CFIP'91, Pau (France), Hermès, 1991 (in French).
18. A. Cavali, S.-U. Kim, S.-O. Loui, P. Maigron, "Environnement orienté vers la vérification et le test de protocoles", Institut National des Télécommunications, Rapport de Contrat de Recherche n°C4609, Octobre 1993 (in French).
19. L. Kahn et al., "State of Research in the area of Formal Test Specification Methods", Draft Technical Report ATM-1006-1, October 1991.
20 S. T. Chanson, H. Dany, M.C. Kim, Q. Li, Y. Lu, S.T. Vuong and S. Zhang, "The UBC Protocol Testing Environment" in O. Rafiq (Ed.), IWPTS'93, Pau (France), September 1993, pp.69-88.
21 S.T. Vuong, W.Y.L Chan and M.R. Ito, "One Chain of Techniques for Generating Protocol Test Sequences" in O. Rafiq (ed.), Réseaux et Informatique Répartie, Vol.3, NO.2, 1993, pp.201-220.
22. K. Naik, B. Sarikaya, "Testing Communication Protocols", IEEE Software, January 92, pp. 27-37.
23. D.Y. Lee and J.Y. Lee, "A well-defined Estelle specification for the automatic test generation", IEEE Transactions on Computers, April 1991.

24. M. Phalippou and R. Groz, "From Estelle specifications to Industrial Test suites" in J. Quemada, J. Mañas and E. Vásquez (eds.), Formal Description Techniques III, North-Holland, 1991.
25. N. Arakawa, M. Phalippou, N. Risser and T. Soneoka, "Combination of Conformance and Interoperability Testing" in M. Diaz and R. Groz (eds.), FORTE'92, Formal Description Techniques V, North-Holland (1993), pp.397-412.
26. W. Chun and P.D. Amer, "Test Case Generation for Protocols Specified in Estelle" in J.Quemada, J. Mañas and E. Vásquez (eds.), Formal Description Techniques III North-Holland (1991), pp.191-206.
27. O. Henniger, B. Sarikaya and S. Biedlingmaier, "Test Suite Generation for Application Layer Protocols from Formal Specifications in Estelle" in O. Rafiq (Ed.), IWPTS'93, Pau (France), September 1993, pp.69-88.
28. B. Sarikaya, B. Forghani and S. Eswara, "Estelle-based Test Generation Tool", Computer Communication, Vol.14, NO9, November 1991.
29. P. de Saqui-Sannes, K. Drira, J.-P. Courtiat, P. Azéma, "Dérivation de séquences de test et testeurs canoniques à partir de spécifications Estelle*: une expérience avec le protocole MMS" in R. Dssouli, G. von Bochmann (réd.), CFIP'93, Hermès, 1993, pp. 474-489 (in French).
30. P. de Saqui-Sannes, J.-P. Courtiat and L.F. Rust da Costa Carmo, "Verification and interoperability testing of MMS services", submitted to publication, May 1994.
31. S.T. Vuong, H. Janssen, Y. Lu, C. Mathieson and B. Do, "TESTGEN: An Environment for Protocol Test Suite Generation and Selection", Computer Communications, Vol.17, No.4, April 1994.