A Practical Effort to Generate Conformance Tests for MIL-STD 188-220B

Umakanth Puppala, Guangmeng Meng, Paul D Amer
Dept. of Computer and Information Sciences, University of Delaware
Newark, DE 19716, USA

Abstract

This paper describes the exercise of generating conformance tests for Type 2 Connection Oriented Data Link Layer Service for the protocol MIL-STD 188-220B. This effort is a continuum of work done at the University of Delaware's Protocol Engineering Lab. MIL-STD 188-220B has been specified in the Formal Description Language Estelle. The practical problem of creating test cases automatically from the Estelle specifications has been a much-researched issue at Protocol Engineering Lab. Due to the complexity of Type 2 Service it was necessary to combine a manual method with the Estelle Specification to generate conformance tests.

Keywords: conformance testing, test case generation, protocol specification and testing, Estelle, MIL-STD 188-220.

1. Introduction

Researchers in the Protocol Engineering Lab at the University of Delaware have been supported by the U.S. Army's (ATIRP) Advanced Telecommunications / Information Distribution Research Program in the specification and testing of Army communication protocols. Estelle is an ISO International Standard Formal Description Technique (FDT) for the specification of computer communication protocols [1, 2]. Estelle is capable of specifying distributed, concurrent information processing systems and is an expressive, well-defined and well structured language that is capable of specifying distributed, concurrent information processing systems in a complete, consistent and unambiguous manner. Estelle extends a finite state model by adding control structures and data types that provide ability to fully specify network communication protocols and services.

The problem with describing a protocol with clarity and precision using a natural language (e.g., English) is a well-known one. Apart from the vagueness introduced when a natural language is used, ambiguities and contradictions are difficult to detect when related protocol functionalities are defined in different regions of a specification separated by several pages of unrelated text. These problems are eliminated in a formal Estelle specification. All actions for a particular context are defined in one place within an Estelle specification. A specification makes the conditions for state transitions explicit through Estelle constructs. Since Estelle makes it possible to create a complete and unambiguous protocol model, the test cases generated from an Estelle specification can potentially achieve higher fault coverage. MIL-STD 188-220, originally developed in 1993, is now mandated for CNR (Combat Net Radio) communications. It is being implemented in US Army, Navy and Marine Corps systems, and has been demonstrated initially during the Army's Advanced Warfighting Experiment in 1997. The Protocol Engineering Lab researchers at the University of Delaware used Estelle to specify several parts of 188-220 protocol suite.

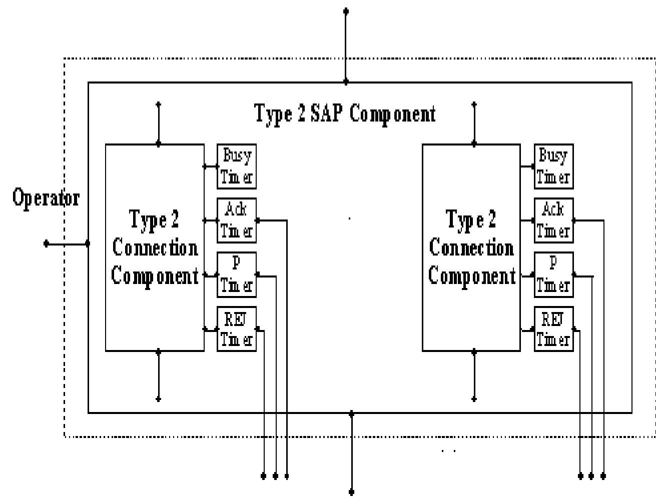
188-220B is a military standard for interoperability of command, control, communications, computers, and intelligence over Combat Net Radios. 188-220B focuses on three layers: Physical, Datalink, and Network. As mentioned previously the test cases generated concern themselves with the testing of the Datalink layer.

Overview of the paper: We present a view of the architecture as defined by the formal specification in (Section 2). The approach to the test generation and the method used are described next (Section 3). We then mention the difficulties faced in our task (Section 4).

2. Architecture of the Datalink Layer Type 2 Service Component

The Datalink Layer provides the control functions to
to ensure the transfer of information over established physical paths, to provide framing requirements for data

and to provide for error control. MIL-STD 188-220's Type 2 Datalink Service provides the Connection mode Operation: a data-link connection shall be established prior to any exchange of information bearing PDUs.



Type 2 Architecture

Fig. 1

Our Estelle specification defines an architecture for Type 2 Service as shown in Fig 1. The architecture consists of several components - They are: Type 2 SAP Component; Receive Ready/Receive Not Ready Busy Timers; P-bit Busy Timer; Acknowledgement Timer; and a Reject Timer. A peer representing an operator or management component is shown in the figure. The interface with an operator module that can interact with the implementation is shown in Fig.1. This operator module abstracts the link layer's interactions with both a human operator and the system management. The normal communications cycle between Type 2 Systems consists of transferring PDUs from the source to the destination, and acknowledging receipt of these PDUs in the reverse direction.

3. Test Case Generation for MIL-STD 188-220B Datalink Layer

The Army Communications-Electronics Command (CECOM) Software Engineering Directorate is developing a Conformance Tester that evaluates the 188-220 implementations identifying where they differ from the standard. This information is used to guide future implementations and standard evolution. In support of this endeavor, the Protocol Engineering Lab at the University of Delaware has been developing test cases to be used by the Conformance Tester at CECOM. The test cases specify a logical sequence of test steps that must be performed to test the Datalink Layer of the protocol stack. Since it is theoretically not possible to perform exhaustive testing of an implementation, tests are made on the events that affect state / transition and exception conditions. Here the behavior of the implementation is observed externally in a means known as black box testing. In conformance testing, the implementation under test (IUT) is viewed as a black box whose behavior is specified by a set of observable actions, called outputs. Applying a set of externally controllable inputs generates outputs.

3.1 Testing Methodology Approach

Ideally we would have liked to utilize the Estelle specification of the protocol to automatically generate test cases, but the tools available today are not capable of generating tests for a protocol of the size of MIL-STD 188-220B. An alternative is to expand Estelle's extended FSMs into pure FSMs and make use of the existing procedures to generate tests from pure FSMs. However this approach results in state explosion – the expanded pure FSM would have too many states to be of any practical use. We adopted the method of generating tests manually for MIL-STD 188-220B's Type 2 Datalink Layer Service.

Here the protocol specification is given as an automaton and the system under test, also modeled as an automaton, is compared to the specification to determine whether the IUTs behavior conforms to that of the specification. Two representations of the protocol were used: the MIL-STD 188-220B English representation and the University of Delaware's Estelle specification.

3.2 Transition Types and Testing Discipline

For testing purposes and for clarity, the test cases generated for MIL-STD 188-220B Type 2 Datalink Layer Service were divided into seven categories where all categories test both valid transitions and illegal input transitions.

valid transitions: defined for the "normal" behavior of the tested implementation under test.

illegal input transitions: test inputs that are syntactically incorrect and manifest faulty behavior.
E.g., an unsolicited acknowledgment PDU

There are infinitely many sequences of legal inputs; hence the goal was to specify a subset of likely inputs.

The choice is an incomplete one but the complexity of the protocol prevented automated test case generation. The exercise of generating the test cases was split into two phases. At all times it was assumed that the tester station and the implementation were reset so that both the ends started off afresh (referred as the Asynchronous Balanced Mode) for each test case. For each test case the timer intervals have been specified roughly in seconds and the amount of time is within the legal limits as mentioned in the protocol specification. The test cases start off from tests that are basic. They increase in complexity as the number of tests grows in size. A total of eighty test cases were generated from our manual method of test case generation.

3.3 Phase 1 - Test Case Generation from the English Representation of the Protocol

The specification of the protocol in English (referred as the English Spec) was analyzed thoroughly; this phase consisted of reading through the text identifying possible test cases. The specification of the protocol in Estelle (referred as the Estelle Spec) was used when there were seemingly ambiguous statements in the English Spec, i.e., phase one was based on the English Spec with the Estelle Spec as a backup. We classified the test cases generated in this phase into six classes taking the nature of the protocol into account. The seventh category (as mentioned above) comprises of the test sequences generated in Phase 2.

The categorization is as follows:

- protocol parameter tests
- “station busy” condition tests
- “out-of-sequence” behavior tests
- logically disconnected mode tests
- communication link resetting tests
- exception conditions tests

Examples of protocol parameters that were tested are the values of variables such as the maximum number of retransmissions and the maximum number of outstanding PDUs at a given station. The "station busy" tests test the implementation’s responses to "busy" conditions at the tester’s end. "Out of sequence" tests are testing the implementation by sending out-of-sequence PDUs to the implementation and testing for retransmission(s).

Testing the logically disconnected mode consists of logically disconnecting the communication link between the implementation and the tester and testing the response of the implementation by sending various kinds of PDUs. The communication link is reset and the implementation is tested for the implementation’s response in the communication link resetting tests. Testing the exception conditions or Frame Reject Response (FRMR) test an implementation’s response when "invalid" PDU(s) are sent to an implementation. Examples of invalid PDUs include PDUs with information field(s) when an information field is not permitted by that PDU definition, PDU(s) with information field whose size exceeds the maximum length that can be accommodated by the receiving station on that data-link, and PDU(s) with invalid sequence numbers. A sequence number is invalid when it is a value that has already been used or falls out of the "window" limit. These tests cover the basic protocol functionality and, as mentioned before, include the tests for the valid transitions and illegal input transitions. An example test case follows:

Test : A test for sequence number resetting.

Section : 5.3.7.2.7 (p 5 – 48)

Step 1:
Test Case Input
Send I PDUs with sequence numbers N (S) = 0, 1, 2, 3, 7 to the IUT.
Test Case Output
The IUT sends back one, two, three or four S PDUs acknowledging the four I PDUs that the IUT receives. The IUT sends back a REJ PDU with N(R) =4 or SREJ PDUs N (R) =4 (bit P =1), N(R) = 5, 6 (bit P =0).
Step 2:
Test Case Input
Send a RSET command PDU to the IUT.
Test Case Output
The IUT sends back a UA response PDU.
Step3:
Test Case Input
Send an I PDU with N (S) = 0 to the IUT.
Test Case Output
The IUT sends back a RR response PDU with N(R) = 1.
Comments
The RSET command PDU is used for sequence number resetting. Here the test shows an example of data transfer where a REJ exception condition occurs. The resetting procedure is employed to reinitialize the receive-state variable in the addressed station (the IUT) V (R) and the send-state variable in the local station (the tester). It is worth mentioning that after the resetting procedure a station may choose to resend the contents of the information field of unacknowledged outstanding I PDUs.

This test case tests section 5.3.7.2.7 of MIL-STD 188-220B. The tester should send I PDUs (data) with sequence numbers 0,1,2,3,7. The IUT should observe that there are missing I PDUs for sequence numbers 4,5,6 and return appropriate ACK information. Then the tester resets the IUT, and tests if the IUT is prepared to restart sequence numbers from 0.

3.4 Phase 2 - Test Case Generation from the Estelle Specification of the protocol

Phase two of test generation was a rigorous one in which we used the Estelle Spec going through the Finite State Automaton (FSA) tracing transitions for different inputs and generating test case sequences per se.

The behavior of a hypothetical implementation was traced on paper for the set of our inputs. The tests generated in this phase are meant to test the IUT in a more rigorous manner - each of the test sequences can be called a compound test because each test sequence encompasses two or more tests that have been categorized earlier. In this phase, going through the FSA comprises starting in a "Normal" state and going on a transition tour. Here as mentioned before the test cases are "compounded" which implies they test the implementation in a series of inputs and outputs. In this phase of test generation we use the notion of "before the IUT responds" to imply that the time interval between the corresponding test inputs is of a small magnitude. An example test case follows:

Test : Test the IUTs behavior .

Section : 5.3.7.2.5.4.1(p 5-44), 5.3.6.2.3.3(p 5-33), 5.3.7.2.5.10(p 5-46), 5.3.6.2.2.2(p 5-32), 5.3.6.2.2.4(p 5-32), 5.3.6.2.3.4(p 5-33), 5.3.6.2.3.5(p 5-33)

Step 1:
Test Case Input
Send I PDUs with sequence numbers N (S) = 0, 1, 2 to the IUT.
Test Case Output
The IUT sends back one, two or three RR response PDUs acknowledging the three I PDUs that the IUT receives.

Step 2:
Test Case Input
Send an I PDU N (S) = 10 to the IUT.
Test Case Output
The IUT sends a REJ PDU or SREJ PDUs to the tester.

Step 3:
Test Case Input
Send a RNR command PDU with the bit P to 1 to the IUT.
Test Case Output
The IUT sends back a RR response PDU with bit F = 1.

Step 4:
Test Case Input
Send a RSET command PDU to the IUT.
Test Case Output
The IUT sends back a UA response PDU to the tester.

Step 5:
Test Case Input
Send I PDUs with sequence numbers N (S) = 0, 1 to the IUT.
Test Case Output
The IUT sends back one, or two RR response PDUs acknowledging the three I PDUs that the IUT receives.

Step 6:
Test Case Input
Send a RR command PDU with an information field to the IUT.
Test Case Output
The IUT sends back an FRMR response PDU with C/R = 0, V (S) = 0, V(R) = 3, bit X =1 and bit W = 1.

Step 7:
Test Case Input
Send a FRMR response PDU to the IUT with C/R = 1, V (S) = 3, V(R) = 0, bit W = 1.
Send I PDUs with sequence number N (S) = 4 to the IUT.
Send a DM response PDU to the IUT.
Send an I PDU with N (S) = 0 to the IUT.
Test Case Output
The IUT sends back a DM response PDU.

Comments
The tester tests the IUT by sending an out of sequence PDU. The tester pretends that the tester is busy and tests the IUTs response for a busy condition at the tester. The tester then checks the IUTs response to a RSET command PDU and tests the IUT for normal data transfer. Then the tester sends an invalid PDU to the IUT and the IUT responds with an FRMR response PDU. The tester sends a sequence of PDUs and the IUT sends back a DM response stating that the IUT is now in the ADM state. [MIL-STD 188-220B Estelle Specification] - (transitions ISO_75, [ISO_70_OPTION_1 | MOD_11], ISO_124, ISO_63A, ISO_75, ISO_ 61, ISO_51, ISO_18, ISO_15, ISO_4).

The test case tests MIL-STD 188-220B for the sections indicated in test case. The tester should send I PDUs with sequence numbers 0,1,2 and the IUT should acknowledge the I PDUs sent to the IUT. The tester then should send an out-of-sequence PDU (sequence number = 10) and the IUT should send back a REJ (Reject) PDU or SREJ (Selective Reject) PDUs. The tester then should send a RNR (Receive Not Ready) PDU to the IUT indicating a "station-busy" condition at the tester's side and the IUT starts the Busy Timer at the IUT's end - the IUT should send back an RR (Receive Ready) response PDU after the Busy Timer times out. The communication link should be then reset with the tester sending the IUT a RSET command PDU and the IUT acknowledging that RSET PDU. An invalid PDU should sent to the IUT and the IUT should send back an FRMR (Frame Reject Response) PDU in response. The tester then should send a FRMR response PDU to the IUT followed by an out-of-sequence IPDU, a DM (Disconnect Mode) PDU and then an IPDU. The response from the IUT should be DM response. (IUT is now logically disconnected from the tester).

3.4 Restrictions and Hurdles

Generating test sequences manually for a formal specification the size of MIL-STD 188-220B was non trivial as the finite state automaton had to be simulated manually for test inputs. The exercise demands a lot of attention to each transition in the specification as the events are traced on paper.

While each generated test sequence is reliable, it takes a bit of work to be done manually before forming a test sequence. Since the events are traced out manually, the behavior of the timers has not been specified concretely but that aspect would not affect the integrity of the generated test sequences.

4 Conclusions

We have described the exercise of generating test cases for conformance testing manually from the English version and the formal specification of the protocol. Although the desired (and ideal) manner of generating test sequences would be to utilize the formal specification specification of the protocol and use protocol conformance test-case generation software in generating conformance tests, the complexity of MIL-STD 188-220B Type 2 Service could not be handled by existing protocol conformance test-case generation software. The method used involves tracing the transitions for various inputs and transforming the results into a test sequence. The whole exercise of test-case generation was divided into two phases with the second phase accounting for the more rigorous test cases (the Estelle Spec was used more in the second phase as compared in the first phase when the English Spec was used more). All test cases are presented in informal English with a test case having inputs to the IUT and the output(s) from the IUT, comments explaining the test sequence. A total of eighty test sequences were generated and have been delivered to CECOM to aid the development of the 188-220 conformance test facility.

5. References

[1] S. Budkowski, P. Dembinski. An Introduction to Estelle: A Specification Language for Distributed Systems. Computer Networks and ISDN Systems, 14(1), 1987.

[2] ISO International Standard 9074: Estelle – A Formal Description Technique Based on an Extended State Transition Model. Information Processing Systems – Open System Interconnection, 1989.

[3] P.Amer, M.Fecko, A.Sethi, U.Uyar, A. Duale. Formal specification and conformance testing of army communications protocols. Proc 5th ARL/ATIRP Conf, College Park, Mar01.

[4] Military Standard – Interoperability Standard of Digital Message Device Subsystems (MIL-STD 188-200B), 20Jan98.

[5] M. Fecko, U. Uyar, P. Amer, A. Sethi, T. Dzik, R. Menell, M. McMahon. A success story of formal description techniques: Estelle specification and test generation for MIL-STD 188-220B, Computer Communications, 23(12), 6/00, 1196-1213.