Automatically Generated Test Frames from an S Specification of Separation Minima for the North Atlantic Region

Michael R. Donat
donat@cs.ubc.ca

April 30, 1998

Abstract

A partially automated process for generating tests has been experimentally applied to a formal specification of a real world specification for air traffic separation minima. This report discusses the problems addressed by this process along with how and why this automation was achieved.

Technical Report TR-98-04
Department of Computer Science
University of British Columbia
Vancouver, B.C.
CANADA

Contents

1 Introduction

This document reports on the semi-automatic generation of a set of 169 test frames from a formal specification of aircraft separation minima for the North Atlantic. Appendices C and D contain 169 test frames which were automatically generated by a software tool from a parseable representation of the separation minima. Figure 1 provides a sample of one of the automatically generated test frames. The combined set of 169 test frames provides complete coverage of all conditions contained in the separation minima specification. This completeness is defined by a precise coverage criterion. 125 of the 169 test frames contained in Appendix C are instances of the "separation exists" condition. The remaining 44 test frames are instances of the "separation does not exist" condition.

Stimuli	Response
1. AngularDifferenceGreaterThan90Degrees (RouteSegment A , RouteSegment B) 2. ¬ (IsSupersonic B) 3. IsTurbojet A 4. IsTurbojet B 5. ¬ (IsWestOf60W B) 6. ¬ (InWATRSAirspace B) 7. ReportedOverCommonPoint (A , B) 8. ept (A , B) + 10 < "separation check time"	1. "are separated" (A , B)

Figure 1: A test frame from Appendix C.

Each test frame specifies a specific combination of conditions corresponding to a single step in a test procedure.[1] The contents of the "Stimuli" field of each test frame are used to determine the contents of the "Stimuli" field of a test step. A test engineer would refine a test frame into a test step by entering appropriate data values into the "Stimuli" and "Responses" fields of the test step such that the "Stimuli" of the test frame are satisfied.

The test frames in this report are provided as a demonstration of the capability of this test generation approach to produce test frames for a logically complex specification. It is expected that these 169 test frames could be used

[1] A test procedure is a sequence of test steps. Each test step contributes to the demonstration that a specified requirement has indeed been implemented.

directly by test engineers in the development of test procedures for systems that monitor air traffic over the North Atlantic.

The generation of these 169 test frames was performed by means of an algorithm based on a specific, precisely defined coverage criterion. The separation minima were originally written in a formal table notation [1] and was not authored with the intention of generating test frames. The formal specification of this separation minima is based on a description provided in a source document entitled "Application of Separation Minima for the NAT Region" (3rd edition, effective December 1992) published by Transport Canada on behalf of the ICAO North Atlantic Systems Planning Group. The table-based specification was algorithmically translated into an S [6] specification. This report addresses the algorithmic derivation of test frames from this S specification. The coverage criterion used to generate test frames is similar to the intuitive notion that a test exists for each cell in the table-based specification. Each step in this derivation is a logical inference. These inferences can be grouped into meta-steps which parallel the steps that would be taken by a test engineer in a manual process.

Section 2 of this report outlines a process for the refinement of test frames from Appendix C or D into test steps within a test procedure. An overview of the process used to generate the test frames contained in Appendices C and D is briefly described in Section 3. The coverage criterion determines the number of test frames generated as well as serving as the basis of any claim about the completeness of a test procedure. Section 4 provides a description of the coverage criterion used to generated the test frames. Appendix B provides a mathematical definition of this coverage criterion. For each of the test frames, all of the conditions specified in the "Stimuli" field of the test frame are both necessary and sufficient. Section 5 of this report describes an alternate approach which supplements the necessary and sufficient conditions with additional conditions that fully differentiate the test frame from other test frames as a means of helping the test engineer ensure that the expected response has a unique cause. The time required to generate these test frames is described in Section 6. A brief summary of this report is provided in Section 7. The S specification of the separation minima is given in Appendix A.

2 Test Steps from Test Frames

A softcopy of the test frames can be developed into test steps by following the steps below:

1. Sequence the test frames into outlines of test procedures.

2. For each test frame in an outline, select appropriate values that satisfy the stimuli specified by the test frame in a manner compatible with the response in the previous test step.

If it is not possible to select values in step 2, either the outline is infeasible or previously selected values must be adjusted to construct a feasible test procedure.

3 Test Frame Generation: Process Overview

This overview provides a brief introduction to the test frame generation process. Details of this process are not essential to the use of the test frames in Appendices C and D. The process used to generate test frames uses an S specification of system requirements and a test frame generation tool, TCG. The purpose of this process is to enhance the current manual process through automation while leaving enough flexibility for engineering judgement to be applied. Figure 2 illustrates this process.

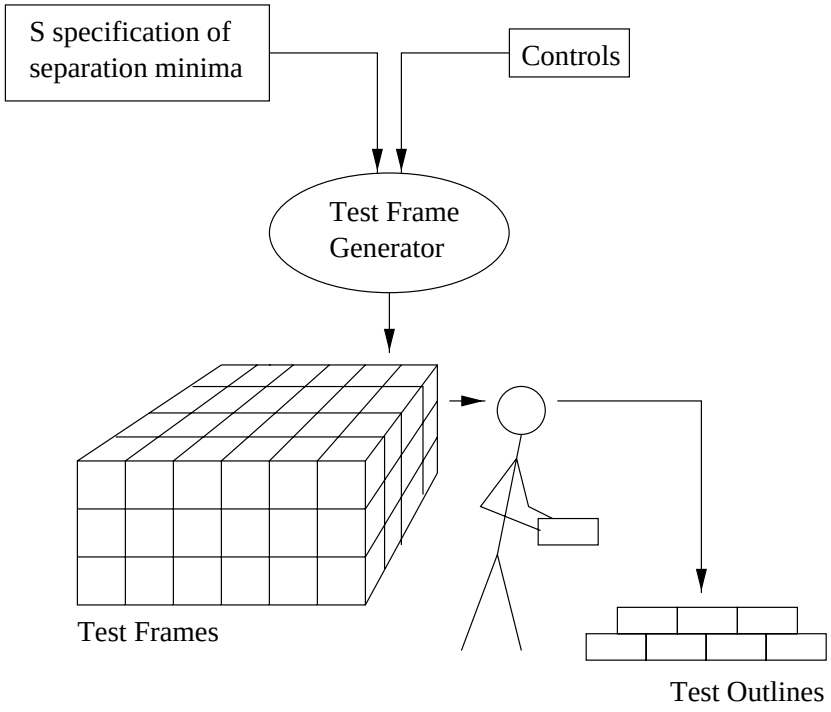

Figure 2: Automatic Generation of Test Frames

Once an S specification has been obtained, the process of generating test frames involves the following steps:

1. Ensure that the specification is composed of stimulus/response requirements of a system. For example, the original S specification simply stated the conditions for separation and did not specify requirements for a system. This was easily translated into the stimulus/response system requirements specification

   ```
   forall A B.AreSeparated (A,B) ⇔ "are separated" (A,B).
   ```

   This specification requires that the system indicate that two aircraft are separated precisely when they are separated according to the requirements specified by `AreSeparated(A,B)`.

2. Add domain knowledge to document dependencies between conditions. This information is used to eliminate infeasible tests. The separation minima specification contained dependency information that was converted into the form expected by TCG.

3. Use the TCG tool to generate test frames from the S specification.

The test frames produced by this process can be used to derive test steps as described in Section 2.

4 Coverage Criteria

The completeness of a test set is determined by a coverage criterion. The test frames in Appendices C and D were generated using a condition coverage criterion. In common terms, this criterion ensures that there is at least one test frame for each condition in the S specification of the requirements. This coverage criterion is based on a mathematical foundation [2]. The precise mathematical definition of this coverage criterion is given in Appendix B. This coverage criterion is intended to be a precise interpretation of the guidance provided in paragraph 6.4.4.1(a)[2] of DO178B [7]: "test cases exist for each software requirement."

This coverage criterion is illustrated by the following example:

> The condition R exists if all of the following conditions are satisfied:
>
> 1. condition A is true or condition B is true, and
> 2. condition C is true or condition D is true.

In this example, the letters A, B, C, D, and R are used to symbolically represent a set of conditions. For instance, the letter A may actually be a phrase such as "the target is using standard pressure setting." Given that each of the four

[2] 6.4.4.1(b) refers to data selection.

conditions A, B, C, and D can be true or false, there are sixteen possible logical combinations of these values. But, of course, it is not practical to generate test steps for each of the possible logical combinations since, in general, the number of test cases would grow exponentially with the number of conditions. The coverage criterion defined mathematically in Appendix B, requires each requirement to be verified once in the sense that every condition must appear in at least one test procedure step. The coverage criterion also requires the conditions to be both necessary and sufficient. For the above example, these constraints can be satisfied by just two test procedure steps. A step in which condition A and condition C are both true together with a step in which condition B and condition D are true would satisfy this coverage criterion. An equally valid combination is a step in which condition A and condition D are both true together with a step in which condition B and condition C are true.

5 Test Frame Styles

The TCG tool is capable of listing conditions for test frames in one of two styles. The "base style" lists only those conditions that are necessary and sufficient to cause the response. However, this list may not be sufficient to differentiate this cause of the response from that of an overlapping test frame. For this purpose test frame conditions can be listed using the "differentiated style." The style is selected by the test engineer.

The difference between "base style" and "differentiated style" is illustrated in the following example.

```
Produce response R if any of the following conditions are
true:

 1. the value of field X is less than 5,
 2. the value of field Y is less than 3, or
 3. the value of field Z is less than 7.
```

The test frames for this fragment using a base style are:

–Test Frame 1:

Stimuli	Response
1. X < 5	1. R

–Test Frame 2:

Stimuli	Response
1. Y < 3	1. R

–Test Frame 3:

Stimuli	Response
1. Z < 7	1. R

This style allows for the maximum amount of choice exercised by test engineers in constructing test steps. However, while specifying the test step corresponding to test frame 1, it may be necessary to specify values for Y and Z. The test step corresponding to:

Stimulus	Response
1. X = 4	1. R
2. Y = 2	
3. Z = 8	

does not differentiate between test frames 1 and 2. The differentiated style can assist test engineers by adding constraints to the list of conditions that differentiate the test frames. In this example the set of differentiated test frames is:

–Test Frame 1:

Stimuli	Response
1. Y < 3	1. R
2. ¬ (X < 5)	
3. ¬ (Z < 7)	

–Test Frame 2:

Stimuli	Response
1. Z < 7	1. R
2. ¬ (X < 5)	
3. ¬ (Y < 3)	

–Test Frame 3:

Stimuli	Response
1. X < 5	1. R
2. ¬ (Y < 3)	
3. ¬ (Z < 7)	

Differentiated test frames can be useful in ensuring that test engineers construct test steps that are differentiated. However, in some cases, test frame differentiation takes significant processing time and there may be several alternatives to choose from in order to achieve differentiation. In the TCG prototype, the choice between alternatives is arbitrary and might not always be appropriate according to best engineering judgement.

As a second example, compare a test frame for longitudinal separation (Appendix C.1.3) with its differentiated form (Appendix D.1.3). The base test frame is,

Stimuli	Response
1. AngularDifferenceGreaterThan90Degrees (RouteSegment A , RouteSegment B) 2. ¬ (IsSupersonic B) 3. IsTurbojet A 4. IsTurbojet B 5. ¬ (IsWestOf60W B) 6. ¬ (InWATRSAirspace B) 7. ReportedOverCommonPoint (A , B) 8. ept (A , B) + 10 < "separation check time"	1. "are separated" (A , B)

and the differentiated version of the same test frame is,

Stimuli	Response
1. AngularDifferenceGreaterThan90Degrees (RouteSegment A , RouteSegment B) 2. ¬ (IsSupersonic B) 3. IsTurbojet A 4. IsTurbojet B 5. ¬ (IsWestOf60W B) 6. ¬ (InWATRSAirspace B) 7. ReportedOverCommonPoint (A , B) 8. ept (A , B) + 10 < "separation check time" 9. ¬ (VerticallySeparated (A , B)) 10. ¬ (LaterallySeparated (A , B)) 11. EnterWATRSAirspaceAtSomeTime A 12. EnterWATRSAirspaceAtSomeTime B 13. IsWestOf60W A 14. MachTechniqueUsed A 15. MachTechniqueUsed B 16. OnPublishedRoute A 17. OnPublishedRoute B 18. "SameOr Diverging Tracks" (A , B) 19. ept (A , B) + 10 < EndTime ("WATRSOppDir NoLongSepPeriod" (A , B))	1. "are separated" (A , B)

The advantage of the differentiated test frame is that these conditions ensure there is no overlap with another test frame for vertical separation. The

disadvantage is that there may be several different ways to differentiate the test frame, but the current prototype test frame generator takes this flexibility away from the engineer by making an arbitrary choice. It is important to note that test frame style is independent of coverage criteria.

6 Processing Times

Computing the base test frames required a total of three hours[3] on an Ultra-Sparc 60. Computing the differentiated test frames required five and a half hours on the same machine. Constructing an initial set of scripts for generating test frames took approximately one hour.

Since the S specification (Appendix A) is large and complex, the TCG tool does not have the capacity to process it in full detail. An iterative approach was used to overcome this problem. In the first iteration, the specification was expanded to a level of detail that could be processed by the TCG tool. The resulting test frames contain non-primitives which were expanded in subsequent iterations.

The condition dependencies listed at the end of the S specification where added when infeasible test frames were found in the TCG output or when the TCG tool found no feasible test frames in a particular iteration. (Finding no feasible test frames implies that the input specification for that iteration was also infeasible.) This added a few days to the construction of the scripts for generating feasible test frames. This was due to condition dependencies which exist between different levels of abstraction within the specification. This suggests that although this iterative approach is capable of processing large, complex formal specifications, more work is required to allow this particular type of condition dependencies to be determined with less effort.

7 Summary

This document has reported the production of 169 test frames using an automated process. Test frames can be used during test development to construct test steps within test procedures. The automatic production of test frames from an S specification of system requirements has the potential to reduce the labour required to produce test steps for logically complex conditions such as rules of aircraft separation. In addition, the test frames are produced according to a precise definition of coverage which ensures the coverage provided by the test frames is consistent and homogenous. Conditions for test frames can be listed in one of two styles: 1) necessary and sufficient, or 2) necessary and sufficient along with additional conditions to ensure no test step can satisfy more than one test frame.

[3] The times given are the elapsed time reported by the unix time utility.

The same approach can be applied to other specification languages with a similar semantics [3]. Other issues including requirements tracking (traceability) and specification readability issues, can also be found in [3]. Further details of this research can be found in [5, 4, 8].

Acknowledgments

Nancy Day generated the original version of the S specification of the separation minima for the purpose of this case study from its tabular form in [1]. Conversations with Jim Ronback regarding notions of coverage have been invaluable. Comments by Jeff Joyce have contributed to the presentation of this report.

This work is supported by *formal***WARE**, a university-industry collaborative research project sponsored jointly by the B.C. Advanced Systems Institute, Raytheon Systems Canada Ltd., MacDonald Dettwiler, The University of British Columbia and The University of Victoria.

http://www.cs.ubc.ca/formalWARE

References

[1] Nancy A. Day, Jeffrey J. Joyce, and Gerry Pelletier. Formalization and analysis of the separation minima for aircraft in the north atlantic: Complete specification and analysis results. Technical Report 97-12, Department of Computer Science, University of British Columbia, October 1997.

[2] Michael R. Donat. Automating formal specification-based testing. In Michel Bidoit and Max Dauchet, editors, *TAPSOFT '97:Theory and Practice of Software Development, 7th International Joint Conference CAAP/FASE*, volume 1214 of *Lecture Notes in Computer Science*, pages 833–847. Springer-Verlag, April 1997.

[3] Michael R. Donat. Automatically generated test frames from a Q specification of ICAO flight plan form instructions. Technical Report TR-98-05, Department of Computer Science, University of British Columbia, Vancouver, B.C., Canada, April 1998.

[4] Michael R. Donat. *A Discipline of Specification-Based Test Generation.* PhD thesis, Department of Computer Science, University of British Columbia, Vancouver, B.C., Canada, 1998. In preparation.

[5] Michael R. Donat and Jeffrey J. Joyce. Applying an automated test description tool to testing based on system level requirements. In *8th Annual Symposium of the International Council on Systems Engineering*, Vancouver, July 1998. International Council on Systems Engineering. http://www.incose.org.

[6] Jeffrey J. Joyce, Nancy Day, and Michael R. Donat. S: A machine readable specification notation based on higher order logic. In Thomas F. Melham and Juanito Camilleri, editors, *Higher Order Logic Theorem Proving and Its Applications, 7th International Workshop*, volume 859 of *Lecture Notes in Computer Science*, pages 285–299. Springer-Verlag, 1994.

[7] RTCA, Inc. and EUROCAE. *DO-178B, Software Considerations in Airbourne Systems and Equipment Certification*, 12B edition, December 1992.

[8] Kalman Toth, Michael R. Donat, and Jeffrey J. Joyce. Generating test cases from formal specifications. In *6th Annual Symposium of the International Council on Systems Engineering*, Boston, July 1996. International Council on Systems Engineering. http://www.incose.org.