Jason J. Simas
Computer Science 198
Summer 2001

A SENIOR PROJECT ON FORMAL METHODS

My personal goal was to develop knowledge about Formal Methods that would help me decide whether to study Formal Methods in graduate school, at the PhD level. I figured the type of knowledge I needed was a general understanding of Formal Methods. I initially figured that I could gain this general understanding by studying say three different Formal Methods. However due to the number of Formal Methods I found when I began my research, I figured this approach would not lead to a general understanding but one that was more particular. I then began to look for introductory texts on Formal Methods. Though there were some introductory texts, they appeared to be specific either to a particular Formal Method, to Formal Methods for hardware and not software engineering, or specific to some application area such as avionics. I did find what the Oxford University's Formal Methods website described as introductory articles, and these appeared to be more general; so I read them. They were:

Bowen, J. P., M. G. Hinchey. 1999. "Formal Methods." In High Integrity System Specification and Design, Pp. 127-133. London: Springer.

Bowen, J. P., M. G. Hinchey. 1994. "Seven More Myths of Formal Methods." IEEE Software, 12.4: 34-31. In Bowen & Hinchey 1999, Pp. 135-152.

Bowen, J. P., M. G. Hinchey. 1995. "Ten Commandments of Formal Methods." IEEE Computer, 28.4: 56-63. In Bowen & Hinchey 1999, Pp. 217-230.

Clarke, E. M., J. M. Wing. 1996. "Formal Methods: State of the Art and Future Directions." ACM Computing Surveys, 28.4: 626-643.

Hall, Anthony. 1990. "Seven Myths of Formal Methods." IEEE Software, 7.5: 11-19. In Bowen & Hinchey 1999, Pp. 153-167.

Hoare, C. A. 1987. "An Overview of Some Formal Methods for Program Design." IEEE Computer, 10.9: 85-91. In Bowen & Hinchey 1999, Pp. 210-216.

Wing, J. M. 1990. "A Specifier's Introduction to Formal Methods." IEEE Computer, 23.9: 8-24. In Bowen & Hinchey 1999. Pp. 167-199.

I also read a chapter from an appendix article on Formal Methods from a software engineering textbook. It was:

Pressman, R. S. 2001 "Formal Methods." Software Engineering, Pp. 673-698. New York: McGraw.

Though some of the content from these articles was helpful, much of it wasn't. A lot of the content was arguments for Formal Methods or advice in how to apply Formal Methods in projects. As I said, I was more interested in understanding the nature of Formal Methods and none of the articles described this in the detail that I required for determining whether I wanted to pursue Formal Methods in graduate school. I wanted content that had the generality and detail that I was familiar with in textbooks such as standard software engineering textbooks. A software engineering textbook describes the processes and procedures of engineering software, the different kinds of procedures, and in what *technical* contexts these different procedures are applied. There was no such Formal Methods textbook that I could find.

In light of this, I decided to read a paper that I had earlier decided to overlook because it appeared to me to have content not of the most general kind – a paper by John Rushby that was written for the FAA and NASA called "Formal Methods and the Certification of Critical Systems".

Rushby, John. 1993. "Formal Methods and the Certification of Critical Systems." Unpublished paper. Computer Science Laboratory, SRI International. Pp. 1-312.

In this paper I found content that, like the other introductory articles, mixed a lot of the "what" and "how" Formal Methods with the non-technical "why" and non-technical "when" Formal Methods. However, I did find that his paper did provide much more detail in the "what" and "how". In fact, it had an introduction to some complex mathematical logics and issues (e.g. higher order type theory, Russell's paradox, and decision procedures) that I found very useful in developing an understanding of Formal Methods and its complexity.

Although I had found some good information, I still wasn't satisfied. This paper was written in 1993. I figured a lot could have changed since then. To satisfy myself that I wasn't reading some grossly outdated material, I reread the Clarke-Wing paper which was written in 1996 and also read a paper that was written by Rushby in 1999.

Rushby, John. 1999. "Integrated Formal Verification: Using Model Checking With Automated Abstraction, Invariant Generation, and Theorem Proving." Theoretical and Practical Aspects of SPIN Model Checking: 5th and 6th International SPIN Workshops, ed. D. Dams, et al. Pp. 1-11. London: Springer, 1999.

I decided that the new approaches that were being discussed in these two papers were mostly just more complex applications of the older procedures. For instance, in Rushby's 1999 paper, he appears to be talking about the same stuff in his 1993 paper, but applies these procedures repeatedly feeding the output of one into the output of another and so on. I figured that since I was familiar with a lot of the terms he was using, I really would be gaining a fairly good understanding of the general procedures or mechanics of Formal Methods by studying Rushby's 1993 paper more heavily.

Since the technical information that I was interested in was scattered about in Rushby's paper and since it was 300 pages, I figured that some good scholarly work would be to extract and reorganize this information forming a 12 or so page exposition[1] of it – an exposition that could serve as an introduction just to the technical issues or *mechanics* of Formal Methods. I also figured that I would gain a huge amount of general technical knowledge about Formal Methods this way; much more than I would have learned had I studied, say, the Z specification language and applied it in specifying some sorting algorithm.

My approach to writing the paper was as follows. I read Rushby's paper once – a paper version. I read chapters 1, 2, and the appendix a second time while "electronic"-highlighting[2] passages I found relevant. I extracted the highlights into a text document keeping references to the page the passage came from. I grouped the passages into 20 categories such as Formal Methods and its sub-categories. The resulting text document was 40 pages long. I then read the document twice each time eliminating duplicate or less relevant information until the document was 15 pages. In the process of writing directly from this document, I condensed it further forming my senior project paper which ended up being 12 or so single spaced pages. I am fairly satisfied with my paper, but it is a little terse. The process described in this paragraph took three weeks and about 150 hours. Overall my project took about 200-250 hours which I think is a hefty amount for a three unit course.

For my senior project, I also produced two mini-reports. One was an informational report on the best universities to earn a PhD doing research in Formal Methods. The second was a description of some possible relations between Philosophy and Formal Methods.

Overall the project was a good experience, and I think I proceeded quite intelligently in accomplishing my personal goal of determining whether I should research Formal Methods in graduate school. Although I have not yet made this determination, I do see Formal Methods differently. I was hoping that Formal Methods would provide more *provable* assurance for the quality of an engineered computer system; although this provable assurance is technically possible, it doesn't appear to be practical. Now, I think of Formal Methods as more of a debugging technique for early specifications and designs of a system or of critical parts of a system. If Formal Methods does generally improve assurance to a significant degree, the establishment of that fact will employ more empirical considerations

[1] Expositions are quite common in Philosophy, but less so in Computer Science I think.
[2] I converted the postscript paper to a PDF file, and I used Adobe Acrobat to do the highlighting.

than I before thought. The second lesson is really more of a question. I'm wondering just how much overlap between Formal Methods for hardware versus software engineering there is. I have recently found a collection of surveys of (hardware) Formal Verification methods, and I am wondering how relevant they are to my interests. There also is one textbook on hardware Formal Verification that may prove useful. This list[3] is:

Gupta, Aarti. 1992. "Formal Hardware Verification Methods: A Survey," Formal Methods in System Design, Vol. 1, pp. 151-238.

Kern, C. and M. Greenstreet. 1999. "Formal Verification in Hardware Design: A Survey," ACM Transactions on Design Automation of E. Systems, Vol. 4, April 1999, pp. 123-193.

Kropf, Thomas. 1997. "Formal Hardware Verification : Methods and Systems in Comparison," In Lecture Notes in Computer Science, No. 1287, November 1997, Pp. 1-384.

Kropf, Thomas. 2000. Introduction to Formal Hardware Verification, London: Springer. Pp 1-299.

McFarland, M.C. 1993. "Formal Verification of Sequential Hardware: A Tutorial", IEEE Trans. Computer-Aided Design Integrated Circuits Systems, Vol 12, No 5, May 1993, Pp. 633-654

Seger, C. 1992. "An Introduction to Formal Verification, " Unpublished paper. Technical Report 92-13, UBC, Department of Computer Science, Vancouver, B.C., Canada.

Shankar, A. U. 1993. "An Introduction to Assertional Reasoning for Concurrent Systems", ACM Computing Surveys, Vol 25, No. 3, Sept. pp. 225-262.

Various Contributors. 1996. "Survey of Formal Verification", IEEE Spectrum, June 1996, pp. 61-67.

Yoeli, M. 1991. "Formal Verification of Hardware Design", IEEE Computer Society Press, 1991.

Maybe I should add two final notes. I don't really like the Oxford University's Formal Methods website. It appears to me that there is just too much information and it is poorly organized. I would much rather have access to a website that has less, more consistently high quality information and have it be better organized. My recommendation to other students starting research in Formal Methods is to not rely on that website to provide you with an adequate starting place for your research. The second note is a pair of quotes from John Rushby's paper that perhaps describes the problem I had with finding good information that was both technical and general.

> "Moving beyond this, to the second level of rigor, we find self-contained specification languages, and more structured treatments of proof obligations, and (sometimes) of proofs themselves. Unfortunately, I have found no books that can be recommended unreservedly as good general introductions to this level of formal methods." (105)

> "Formal methods supported by automated tools, and especially those that provide proof checking or theorem proving for the third level of rigor, are generally described in books that are specific to individual systems." (105)

[3] This list was compiled by Jayanta Bhadra of the University of Texas in Austin and can be found at: http://www.cerc.utexas.edu/~jay/fv_surveys/