Analyzing SAFER using UML and VDM++*

Sten Agerholm[1] and Wendy Schafer[2]

[1] IFAD, Forskerparken 10, DK-5230 Odense M, Denmark
[2] Wofford College, Spartanburg, South Carolina, USA

Abstract. The use of graphical notations in conjunction with formal notations can significantly aid in the software modeling process, due to their complementary benefits. Graphical notations are superior for visualizing models at a high level. Formal notations support precise and unambiguous modeling and add rigor to the modeling process. This paper presents a case study illustrating the use of the Unified Modeling Language (UML) in conjunction with the formal object-oriented specification language VDM++ in order to analyze a model of the SAFER system presented in the NASA Formal Methods Guidebook [10].

Extended Abstract

With the recent trend in industry toward object-oriented analysis and design methodologies, research in the area of formal methods has expanded with an additional focus on object-oriented techniques. In particular, the use of diagrammatic views for object-oriented modeling is becoming widely recognized also in the formal methods community as a means to decomposing complex problems and presenting abstract graphical perspectives of models.

Notably, the formal methods group at the NASA Jet Propulsion Laboratory has put forward the complementary benefits of graphical and formal modeling in their work on promoting lightweight formal methods [5, 9, 3]. They observe that object-oriented graphical models can be used as intermediate representations facilitating the process of formalizing requirements, as this can enhance the accuracy of the initial formal specifications and reduce the effort to produce them. They also observe that graphical models can provide higher level structural views of the requirements, while the formal models can fill in the processing details, and allow detailed behavioral analysis.

Though the complementary support of graphical and formal notations seems obvious, there has been a fundamental lack of proper tools that support the transition between such notations. This means that often the transition must be conducted manually and sometimes in a rather indirect way. For example, in the NASA work cited above, OMT models are translated to the PVS specification language [11], which does not directly support object-orientation. Hence, the

* The main body of the paper has been excluded in this version. The full version can be downloaded from the address `http://www.ifad.dk/publications.htm` or by contacting the first author at `sten@ifad.dk`.

translation of key object-oriented concepts such as inheritance between classes can be problematic.

In this paper, we report on a case study where such inconveniences of combining graphical and formal modeling are overcome. Firstly, we use the VDM++ specification language, which is an object-oriented extension of ISO Standard VDM-SL [8, 6, 4] and supported by the VDMTools of IFAD [7]. Secondly, we use a tool, the Rose-VDM++ Link, which automates the translation between VDM++ and UML through a number of transformation rules [2]. This means that UML is beneficial not only to provide intermediate representations prior to formalizing a model as suggested in [5], but also throughout the modeling and validation processes in order to provide abstract diagrammatic visualizations of the formal models. The paper illustrates the use of formal techniques as a complement to the leading industrial design notation and tool, namely, the UML and Rational Rose 98.

Our case study uses the SAFER system from the NASA Formal Methods Guidebook [10], but presents an alternative object-oriented development of the system rather than the modular approach used there. Moreover, this paper focuses on a lightweight, testing-based approach to validation as in [1] rather than verification. VDMTools is used for rigorous checking of models, such as type checking and dynamic consistency checking by executing models.

The full version of this paper (see `http://www.ifad.dk/publications.htm`) is a sequential report of the procedure we followed during our work. Starting from a context diagram and an understanding of the SAFER system, we develop a UML object model of the system. Using the Rose-VDM++ Link, we map this model into a VDM++ specification. After adding the necessary functionality to the model, we execute a suite of validation tests. In the development, we make extensive use of the Rose-VDM++ Link to obtain complementary views of our models.

Visualization is a key element during the development process and this is supported well with the Rose-VDM++ Link. In forward engineering, the user models the overall object-oriented aspects of the system in UML, translates the model to VDM++, and proceeds by giving more concise descriptions to aspects of the model in VDM++. Subsequently, in reverse engineering, the VDM++ specification can be translated back to UML either for documentation purposes or to further elaborate the object-oriented aspects of the model. In this way, round-trip engineering exists between the two notations through the use of the Rose-VDM++ Link.

Object-oriented graphical notations, such as UML, are useful for abstraction and visualization in the modeling processes, but they do not have a mathematical basis. They also fail to provide either a formal or automated analysis of the system, and a model can be continuously examined and re-examined due to a lack of a formal perspective of clarity and precision forming an endless analysis phase. On the other hand, formal notations, such as VDM++, enable validation and verification in the modeling process through precise and unambiguous specification, but are often complicated to structure and organize during devel-

opment. Model development transitions between the two models occur easily with a natural progression from abstraction to formalization in (forward) engineering and from rigorous to more general in reverse engineering. In these ways, the graphical and textual notations are complementary.

References

1. S. Agerholm and P.G. Larsen. Modeling and Validating SAFER in VDM-SL. In *Fourth NASA Langley Formal Methods Workshop*. NASA, September 1997. NASA Conference Publication 3356. Available at http://atb-www.larc.nasa.gov/Lfm97/.
2. S. Agerholm and O.S. Pedersen. Enhanced UML Analysis with Formal Modeling and Validation. In *Draft*, Marts 1998.
3. R. Covington D. Hamilton and J. Kelly. Experience in Applying Formal Methods to the Analysis of Software and System Requirements. In M. Larrondo-Petrie R.Ẽrance and S. Gerhart, editors, *Workshop on Industrial-Strength Formal Specification Techniques*, pages 30–43. IEEE Computer Society Press, April 1995.
4. John Dawes. *The VDM-SL Reference Guide*. Pitman, 1991. ISBN 0-273-03151-1.
5. S. Easterbrook, R.R. Lutz, R. Covington, J.C. Kelly, Y. Ampo, and D. Hamilton. Experiences Using Lightweight Formal Methods for Requirements Modeling. *IEEE Transactions on Software Engineering*, 24(1):1–11, January 1998.
6. J.S. Fitzgerald and P.G. Larsen. *Modelling Systems – Practical Tools and Techniques in Software Development*. Cambridge University Press, 1998.
7. IFAD World Wide Web. `http://www.ifad.dk`.
8. P. G. Larsen, B. S. Hansen, et al. Information technology — Programming languages, their environments and system software interfaces — Vienna Development Method — Specification Language — Part 1: Base language. International Standard, ISO/IEC 13817-1, December 1996.
9. R.R. Lutz. Reuse of a Formal Model for Requirements Validation. In *Fourth NASA Langley Formal Methods Workshop*. NASA, September 1997. NASA Conference Publication 3356. Available at http://atb-www.larc.nasa.gov/Lfm97/.
10. NASA. Formal Methods, Specification and Verification Guidebook for Verification of Software and Computer Systems. Vol 2: A Practitioner's Companion. Technical Report NASA-GB-001-97, Washington, DC 20546, USA, May 1997. Available from http://eis.jpl.nasa.gov/quality/Formal_Methods/.
11. PVS World Wide Web page. `http://www.csl.sri.com/pvs/overview.html`.