

FORMAL METHODS LIGHT: EXPERIENCE APPLYING FORMAL SPECIFICATION IN THE AEROSPACE AND NUCLEAR SECTORS

J. S. Fitzgerald¹

This presentation describes two industrial projects in which formal methods were applied in the early stages of system development. Both projects were based on the selective application of formal methods, with appropriate techniques targeted on those areas of development where they would be expected to yield a benefit. We term this approach “formal methods light” to contrast with the rigorous application of formal modelling and analysis at all stages in the development of a computing system. The projects discussed are:

The BASE Trusted Gateway British Aerospace (Systems and Equipment) Ltd. developed a secure message-handling system, using formal modelling to describe the system’s security policy and its functionality. Here formal techniques were used to help understand requirements and represent designs. The study has been widely reported [1].

The BNFL Tracking Manager BNFL Engineering and Manchester University used formal modelling to analyse the safety properties of a proposed new architecture for systems to track the movement of nuclear materials through industrial processing. Here formal techniques were to assist in the analysis of a proposed architecture rather than as the first stage in software development.

The BASE Trusted Gateway

In this study, software for a system called a *trusted gateway* was developed using formal modelling techniques. This was compared against a parallel development of the system which did not use formal techniques. Secure message handling often involves developing systems to externally-assessed levels of assurance. Higher levels of assurance [2] demand the use of a formal model to describe security policy, and this provided a motivation for the study.

During the development process, the project technical documents (specifications, designs and code) were held in a relational database supporting traceability of design decisions. Within the database, designs were represented by data flow diagrams on the Ward/Mellor model [3] annotated by pseudocode and natural language. The system data, functionality and security policy were additionally represented formally using the ISO VDM-SL [4] formal specification language. Formal models were developed using tools for syntax-checking, type-checking and testing VDM-SL.

The ability to exploit testing was considered particularly valuable in validating the model. Test sets could be generated from the earliest development stages and employed in implementation testing at a later stage. The ability to execute specifications at a workstation was also cited as an aid to training.

The development using VDM-SL was shadowed by another development which stopped short of using a formal model: the data flow diagrams were annotated only in natural language and pseudocode. Progress and costs in the two developments were compared. The cost figures (Fig 1) suggested that the “formal” path consumed more of the allocated budget in early design stages, but less later (excluding training and tool costs). It was found that corrections which had been applied to the conventionally

¹John Fitzgerald is with the centre for Software Reliability at the University of Newcastle upon Tyne.

Phase	Allocated Time	Conventional Path		Formal Path	
		% Alloc	% Project	% Alloc	% Project
System Analysis	40 %	30 %	33 %	35 %	43 %
Software Design	40 %	40 %	47 %	33 %	41 %
Implementation	20 %	17 %	20 %	13 %	16 %
Totals	100 %	87 %	100 %	81 %	100 %

Figure 1: BASE Trusted Gateway: cost distributions compared

developed code following testing had been responsible for it having rather worse size, complexity and performance than the code developed from a formal model.

Conclusions from the BASE study

Viewed over the entire development, the use of formal modelling was not felt to have imposed a significant cost overhead. Formal modelling improved understanding of the system being developed and was felt to help prevent errors from being propagated through the development process.

The project illustrated how formal techniques can be integrated in small steps into an existing development process. However, the benefits from formal modelling were mostly obtained through its application in early development phases.

One week's training was sufficient for a competent engineer to begin to use formal modelling techniques, given a level of tool support and the availability of expert advice. However, it was noted that formal modelling should be supported by "industrial-strength" tools which allow test cases to be examined.

The BNFL Tracking Manager

The second study considered here concerns the modelling of a tracking manager architecture for nuclear plant. The UK Health & Safety Executive funded a study with BNFL Engineering and Manchester University which used a formal model to reason about a system for tracking the movement of nuclear waste through a processing plant. In common with the BASE Trusted Gateway, this study used a formal model at a very early stage of development (before requirements analysis in this case). However, the development and validation of the model, rather than the development of an implementation, was the main purpose of the exercise.

During industrial processing of nuclear materials it is important to be able to trace containers of material to ensure that safety constraints are not violated and that no material goes unaccounted for. A hierarchical structure of tracking managers had been proposed to trace the movement of containers within the processes and manage the hand-over of containers between processes. Safety is a central issue here and so the aim of the project was to show how formal modelling could contribute to the safety case for the tracking manager architecture.

Again, VDM-SL was used as a modelling language and tools were used for syntax- and type-checking the specifications. Here, however, the validation was not done by testing, but by rigorous mathematical proof [5]. Proof was expensive to accomplish, but the project did uncover deficiencies in the formal model which may not have been detected through testing alone, because of the ability to reason over a class of values, instead of testing a number of possible inputs. In its conclusions, the project made some useful recommendations about how proof could be used in real formal modelling projects.

The formal model described the structure of the plant and the whereabouts of containers. Important

safety constraints on the controlled plant were described in the formal model as *invariants*. In addition to modelling the position of containers in the plant, the functionality of the system was modelled by recording the changes which can happen when requests are made and granted or rejected to allow containers to move between processing units. It was then possible to check whether the permitted state changes do not allow the invariant to be violated. This invariant checking forms an important part of validation.

The BASE study conducted validation by applying test cases to the formal model while the BNFL study utilised formal proof. With a formal modelling language it is possible to show that the invariant is respected to such a level of detail that the proof can be checked by a machine. However, the development of such a fully formal proof is time-consuming and not currently well supported by tools. For some critical applications, the production of some proofs may nevertheless be mandatory. Less formal proof (which leaves some steps to human intuition) can be applied in review and inspection activities.

Conclusions from the BNFL study

Recording invariants was felt to be a particularly valuable part of formal modelling. The choice of what to record in an invariant can come from other development activities such as safety analysis. Formal proof was expensive but helped identify deficiencies in the model which testing might not have detected so early. Proof need not be performed in detail, but can be incorporated into review and inspection procedures.

Concluding Remarks

Both applications are relatively modestly sized, but there are examples of larger applications of formal techniques. Using a formal modelling language to record data and invariants is a good way of starting formal specification and appears to represent an affordable “delta” over existing practice. Formal methods need not be seen as prescriptions for the development of software; perhaps they are of greatest use as tools to be applied judiciously where they are most needed, especially in the early stages of software development.

References

- [1] P. G. Larsen, J. S. Fitzgerald, and T. M. Brookes. Applying formal specification in industry. *IEEE Software*, 13(3), May 1996.
- [2] Office for Official Publications of the European Community. *Information Technology Security Evaluation Criteria*, June 1991.
- [3] P.T. Ward and S.J. Mellor. *Structured Development for Real-Time Systems*, volume 1-3. Yourdon Press, New York, 1985-1986.
- [4] P. G. Larsen, B. S. Hansen, H. Brunn N. Plat, et al. Information technology — Programming languages, their environments and system software interfaces — Vienna Development Method — Specification Language — Part 1: Base language, December 1996.
- [5] Juan Bicarregui, John Fitzgerald, Peter Lindsay, Richard Moore, and Brian Ritchie. *Proof in VDM: A Practitioner's Guide*. FACIT. Springer-Verlag, 1994. ISBN 3-540-19813-X.