Embedding Formal Development in Software Engineering

Ken Robinson

1 School of Computer Science & Engineering
University of New South Wales, NSW 2052 Australia,
ken@cse.unsw.edu.au
2 National ICT Australia

Abstract. Formal methods have had considerable success in developing the theoretical fundamentals of many areas of computing, but have had considerably less success at penetrating the areas where those formal techniques could be applied. This can be regarded as a failure of formal methods. This paper is concerned, in particular, with the lack of impact of formal techniques on software development, and especially on software development curricula. The paper presents some diagnoses of the cause of the problem and discusses attempts at integration of formal development into a Software Engineering program.

1 Background

As a general observation the use of formal methods in system development appears to have had very little penetration into practice. One reason we submit is because Formal Methods has developed as though it is a separate discipline sitting in glorious isolation from the rest of Computer Science, Software Engineering and Computer Engineering. While members of the Formal Methods community have a common concern, namely to use rigorous and formal techniques in computing, they are motivated by a variety of problems and there are no foci. For fundamental research in formal methods this is not a disadvantage, but it is a serious obstacle to the transfer of formal techniques to application areas.

1.1 Curricula

The effect of the isolation and separation of formal methods from “the rest” is frequently mirrored in curricula and tends to lead to curricula that are ineffective in the integration of formal techniques into computing applications. A curriculum may contain one or more —frequently only one— course called Formal Methods, but with no connections to other courses. Consequently and conversely, it is frequently the case that the other courses make no reference to, or use of, the formal techniques studied in the Formal Methods course. Most damaging is the
result that students, having seen no application of the formal methods, tend to
dismiss them as irrelevant.

The most recent draft of the Software Engineering Curriculum from IEEE
Computer Society and ACM [6] has a core course called Formal Methods in
Software Engineering. This course is at least heading in the right direction as far
as its name is concerned, but its content is a mixture of many formal methods
and it is clear that no significant application of formal techniques will be able
to take place in the time available.

1.2 From Computer Science to Software Engineering

Much of what passes as formal methods would be classified by this author as
computer science, where the intended connotation is that “computer science” is
a science concerned with the study of computing phenomena. To apply those
methods and techniques in software engineering, attention must be given to the
form of the development process and the need for system decomposition and
composition. These requirements are non-trivial and formal techniques will only
succeed if they can work within and support those requirements.

In a software engineering program it is imperative that formal methods be in-
tegrated at some level in the software development process. The techniques must
be exposed and tested at some reasonable level to the requirements mentioned
above. Only by demonstrating some capability in dealing with those require-
ments can there be any expectation of acceptance of the methods.

At the author’s university we have attempted to address some aspects of those
requirements and to use a formal method in non-trivial practical situations. The
rest of this paper reports on the aims, content and experience of that exercise.

2 Experience at UNSW

At The University of New South Wales (UNSW), we first introduced a formal
method into a project based course in 1991. At that time we used Z and we
devised a process for approximating Z specifications from Data Flow Diagrams
(DFD) [13]. In 1997 we introduced a Software Engineering program [20], in which
we run a series of workshops over the first three years. In each workshop we cover
some phase of the software development process and the students work in teams
on a project. The workshops are practice-based not lecture-based.

In the second year of the program we run a course on Software System Spec-
cification [4], in which we initially taught Z and now teach the B Method (B).
In parallel with that course we run a workshop, Software Engineering Workshop
2A [16], in which B (initially Z) is used to model the requirements for some
system. In the next semester we run another workshop, Software Engineering
Workshop 2B [17], in which the specification is taken though design to a proto-
type implementation.