INTRODUCING FORMAL METHODS TO EXISTING PROCESSES

Andrew Butterfield[1]

INTRODUCTION

This paper examines some of the issues surrounding the introduction of Formal Methods into an existing Software Development Process, and to explain the circumstances for which it is best suited. We place considerable emphasis on the need to be very careful when introducing a new software development or quality assurance technology, such as formal methods, in order to ensure that it fits in well with existing processes and practice in the organisation in which it is being introduced. The discussion is backed up by reference to the industrial experience of the authors in applying and introducing formal methods to large software and systems developers, particularly in the telecommunications field.

Experiences with Formal Methods

Through K&M Technologies Limited the authors worked on various projects in industry, applying formal methods to various parts of the development process life-cycle. We discuss these experiences here, with particular emphasis on the benefits that arose from the use of formal methods.

In order to avoid the technical details usually associated with specific problem domains which tend to cloud the key issues, we have chosen to recast the problems in terms more readily accessible to all. In addition to understanding some of the key results, it is important to note that in each case the client/customer organisation was comprised of large traditional engineering groups/divisions in addition to the more recent software (engineering) group/division.

Experience 1

Scenario: Consider a large public library, organised in the traditional manner around the basic concepts of book, catalogue, reader, librarian, etc. It is required to "computerise" the library in the sense that basic concepts shall be open to fresh interpretations/implementations. The concepts of catalogues, borrower, etc., must all be revisited.

Practice: Two different models (traditional/computerised) of a portion of the same system were constructed. This included invariants and relationships between the models. When one attempted to verify the property, established for the traditional model, with respect to the computerised model, a serious inconsistency was determined.

Lesson: Do not rely on one model alone. Products of sufficient complexity give rise to different views. Checking for cross-consistency will usually identify errors in understanding of the requirements.

[1] K&M Technologies

Experience 2

Scenario: A large subscriber base exists for a company's products and is automated. Much of the technical intricacies of the in-house computing system are distributed between one engineer's memory and some incomplete documentation. A tender for a new hardware/software database platform is to be issued, appropriate to its telecommunications needs and rigorous fault-tolerance requirements.

Practice: The tender document had to be in precise English. The engineer responsible was debriefed over an intense two day period and a formal model constructed within one week. A draft text was constructed and analysed. The final text delivered to the suppliers was backed up by the formal model.

Lesson: A formal model clarifies the real requirements and may be used as a guide in constructing the requirements document. All concerned took the matter with a great deal of earnestness and had greater confidence in the written text, knowing that there was a model behind it. Although there was no requirement that the delivered product conformed to the formal model, there was an earnestness on the part of the suppliers to be seen to do as well as those who had prepared the requirements text.

Experience 3

Scenario: A particular product had to be developed by three distinct groups, each building a distinct component on a system in which they all communicated with each other. Each group had its own set of requirements.

Practice: The construction of a formal model was used in an attempt to understand the (functional) requirements. It was determined quite quickly that there was a serious problem at the interfaces between the groups. The result would have been disastrous for the product as a whole.

Lesson: There must be a single master requirements document and, consequently, overview model. Often, in the course of modelling, information has to be sought from various groups. Such an activity may lead to an identification of management problems or problems in the overall process.

Experience 4

Scenario: A corporation needed to get a product out into the marketplace in time in order to remain competitive. Management believed in the usefulness of semi-formal design methods with the appropriate tool support. Training was obtained, the project was launched, and a state-of-the-art tool which supported the method in question was acquired at considerable cost.

Practice: First, in spite of the hard work, the recommendations of the method were not followed. Only a single iteration of the design had been attempted and the design had been processed by the tool with approximately 500-plus pages of output. All the relevant information was recorded in a database maintained by the tool and printed out as appendices. Upon interviewing the team members, it emerged that any change made to the design would have rendered the database inconsistent. In other words, the tool was seriously flawed.

Lesson: The project was ultimately abandoned at a considerable cost running into tens of millions of ECU's. There were other reasons apart from the design failure. A key one is that tools for any method need to be evaluated in-house before a major project is committed to using them.

Summary of Experiences

In the case of the perceived level of difficulty of the mathematics underlying the Formal Method used, the best remark is that which we quote here from a (software) engineer in one of the customer organisations who exclaimed most enthusiastically: "It's not rocket science math!"

The mathematics is certainly well below that expected of any (ordinary) engineer in an organisation. We never attempted to inject formal methods into the mainstream process within any organisation. It was evident that the use of formal methods had to be complementary to whatever process was in place. We do recognise that there may be certain situations where it would be essential to have inline formal methods. But, in general, the formal methods activity will be alongside and apart.

Finally, it is worth noting that computer technology has an enhancing effect on process. If the process is bad then the computer technology will make things worse, if it is good, then computerisation will improve it. Formal methods have a real role to play in identifying and exhibiting faults in the process.

Adopting Formal Methods

Given that an effective software development process is in place, we now turn to the issue of how best to introduce formal techniques. We consider that a successful introduction of formal methods requires the following things:

1. Champions
2. Critical Mass
3. Gradual Integration
4. Education
5. Time

Champions

As with any new technology, there will often be considerable natural resistance to the introduction of formal methods. To overcome this inertia, it is important that the adoption of such techniques be overseen by a Champion—a highly motivated individual who understands the advantages of the new technology, and is willing to maintain forward momentum in the introduction of the new concepts and ideas that come with this new technology.

Critical Mass

It is important, especially in the early stages of adopting formal methods, that a sufficient number of people in the organisation are involved in its introduction to ensure that a form of critical mass is reached. As a rule of thumb, in the psychological process of technology transfer, we estimate the size of the critical mass to be five to seven people.

Gradual Integration

The adoption of formal methods requires a certain change in the kinds of concepts used, and the culture surrounding, the capture and analysis of requirements, and the production of specifications. Like all conceptual and cultural changes, a gradual approach to their introduction is much more likely to succeed, than a sudden switch. In particular, as far as process development is concerned, it is very important to ensure that the adoption of a formal method does not lead to a severe or abrupt change in working practices at a critical point in the life cycle.

Education

As with any new technology, the intended users will have to be educated in its principles, and trained to apply it and its tools effectively. As formal methods is now an integral part of many third-level curricula, the education step is at least easier today than it was when formal methods first started to appear in industrial settings. Extensive training in the application of various methods is also available. But it should be kept in mind that such training will invariably have to be tailored to meet in-house process requirements.

Time

Formal methods is not a plug-and play technology, and requires time to adopt properly. The key to its successful adoption is its gradual adoption over time—bit by bit. Typical lead times from the date of initial introduction to that of significant pay-back (as determined by profits/quality improvement of the end-product in the marketplace) may be of the order of a few years for a large organisation.

Conclusions

In conclusion, we can offer the following recommendations regarding the introduction of formal methods into a software development process:

1. Have a proper process in place
2. Find a champion willing to run with it and ensure they have enough resources to reach a critical mass
3. Remember—formal methods are developed to help your process work more effectively, so integrate then into your process, rather than disrupt your process to suit them.

This work was supported by K&M Technologies Limited, of which the authors are consultants.