Preface

Software engineering research has different profiles in Europe and North America. While in North America there is a lot of know how in the practical, technical, and organizational aspects of software engineering, in Europe the work concentrates more on foundations and formal modeling of software engineering issues. Both approaches have their individual strengths and weaknesses. Solely practice driven research in software engineering runs in the danger of developing into a shallow field failing to find a solid scientific basis or to contribute substantially to the progress in software engineering. Work concentrating on formal aspects alone is in the danger of becoming too theoretical and isolated from practice so that any transfer into practical application will fail.

Substantial progress in software engineering can be achieved, however, by bringing together pragmatic and foundational work in software engineering research. This can provide a step towards a common scientific basis for software engineering that allows us to integrate the various research results, leading to fruitful synergetic effects. It will also help to identify critical research paths and to develop an adequate paradigm for the scientific discipline of software engineering.

In software and systems engineering it is necessary to distinguish the enormous difference between the dynamics in development we refer to and the limited scope assumed by many of today's software managers who still use outdated techniques. Many of the unsolved problems associated with the old techniques are symptoms of a lack of formalization and a lack of automation support.

It was the goal of this workshop to bring together experts from science and practice in software and systems engineering from North America and Europe. The workshop focussed on unified sets of formal models and associated methods suitable for automation for many aspects of software development, in particular those that address change and those that apply on a large scale. Some of the aspects of software evolution are

- modifiable software architectures,
- resource changes,
- context changes,
- requirements changes,
- changes to decomposition structures, and
- changes in plans.

These issues are closely related to formal representations of the version history, and formal representations of the activities that produced existing versions or have been proposed to produce future versions. The essence of the software engineering product model is to establish and maintain consistency among various kinds of software artifacts throughout the development and evolution process, including consistency between requirements, architectures, and programs.

Automation support is needed to determine dependencies and to use this dependency information to provide decision aid for software synthesis, analysis, and evolution. Many versions of each artifact are produced as the software evolves, and changes in the dependency structure must be recognized and reacted to. The challenge is to formalize the problems in this area better, and to develop some of the badly needed technical solutions.

If we as a community can succeed in doing this, the results will provide convincing evidence that formal methods can have strong practical value, and help reverse the trend of weakening support for the subject from both industry and governments. It seems that previous work on formal methods can be applied to problems related to these topics, but it may require non-traditional approaches. This challenge helped to trigger new ideas at the workshop, and perhaps opened new opportunities for progress.

It is well recognized in the meanwhile that software and systems engineering as an important issue in technical systems still lack a proper scientific basis. The many efforts in academia, especially under the heading formal methods, towards such a scientific basis have produced many valuable and interesting scientific results; but still a lot of work lies ahead of us to actually integrate this with the practice of software engineering. Nevertheless, we can observe that a beginning has been made to bring together practical and scientific approaches. A good example for this is the Unified Modeling Language, which was designed only recently and will evolve further. The fact that a proper semantic basis is needed for a proper methodological support is much more recognized than before. Nevertheless, more efforts are necessary to give the scientific research the right focus looking at the questions that are important for the practice and to stimulate a transfer of ideas between academia and application. It was the goal of the workshop to contribute to this process.

The workshop took place in early October 1997 in Bernried, Germany. It was a highly successful event and an encouraging step towards the unification of the various aspects and techniques of software and systems engineering. It is our pleasure to thank Luqi for the excellent cooperation in preparing and implementing the workshop and Sascha Molterer for his distinguished help in organizing the workshop. We also thank the Army Research Office and in particular Dave Hislop for the financial support.

August 1998 Manfred Broy, Bernhard Rumpe

Table of Contents

Foundations of Software Engineering

Methodology

Evaluation and Case Studies

Tool Support and Prototyping