Preface

The development of products in disciplines such as mechanical, electrical, or software engineering is a challenging task. Costs have to be reduced, the time-to-market has to be shortened, and quality has to be improved. Skilled engineers and sophisticated tools for supporting technical work are necessary prerequisites, yet they are not sufficient for meeting these ambitious goals. In addition, the work of developers must be coordinated so that they cooperate smoothly. To this end, the steps of the development process have to be planned, an engineer executing a task must be provided with documents and tools, the results of development activities have to be fed back to management which in turn has to adjust the plan accordingly, the documents produced in different working areas have to kept consistent with each other, etc.

This book reports on models and tools for managing development processes. It provides both a survey of the current state of the art and presents our own contributions. The material covered in this book is based on research in different engineering disciplines (mechanical, software, and chemical engineering). It presents a unified view on the management of development processes in these disciplines.

The current state of the art is characterized by a large variety of tools for process management. Project management systems support classical management functions such as planning, organizing, and controlling by means of project plans (e.g., PERT charts). Engineering/product data management systems or software configuration management systems are concerned with the products of development processes such as designs, manufacturing plans, and NC programs in mechanical engineering; or requirements definitions, software architectures, and modules in software engineering. Workflow management systems manage the flow of work according to a defined procedure that coordinates activities such that defined objectives are achieved by set deadlines. Process-centered software engineering environments drive the software development process according to a process model that defines the steps to be executed and the constraints on their ordering.

Unfortunately, these tools still suffer from several limitations. Project management systems operate at a too coarse-grained level and do not take the products of development processes into account. Conversely, engineering/product data management systems and software configuration management systems focus on products, but neglect the management of activities. Workflow management systems and process-centered software engineering environments are often too inflexible and do not adequately support the dynamics of development processes.

We present an integrated approach which addresses these shortcomings. We have developed models for managing development processes that consider products, activities, and resources in an integrated way. Moreover, the models are designed to cope with the dynamics of development processes. A development process may rarely be defined in advance; rather, it constantly evolves during execution. Some reasons are given below:

– The tasks to be performed depend on the product structure which is determined only during development. For example, the architecture of a software system defines the modules to be implemented and tested.
– Development rarely proceeds smoothly from one phase to the next. Rather, errors and inadequate solutions which are detected in later phases are fed back into earlier phases. The consequences of feedback may be hard to predict, and may range from small local changes to large global ones.
– "Walking on water and writing software from a specification are easy if both are frozen." In reality, however, development must be prepared to cope with continuous changes to the requirements.
– In order to reduce development efforts, organizations strive for reusing previous results. Then, the development process depends on which results can be reused to what extent. This knowledge is often not available beforehand.
– If milestones have to be accomplished earlier than expected, it may be necessary to accelerate development on critical paths, assign more developers to the project, etc.
– Organizations are constantly striving for improving their processes resulting in optimized process definitions. It is desirable to propagate these optimizations into ongoing development processes.
– Current development methods such as concurrent and simultaneous engineering accelerate development by increasing parallelism. To be successful, they require the sophisticated coordination of engineers working on different parts of a product, or in different working areas such as design and manufacturing planning.

We have developed a management system which provides customized environments for its different kinds of users. The management environment supports managers in coordinating technical activities by presenting graphical, global views, and commands for planning, analyzing, controlling, and monitoring. Developers use the work environment which maintains agendas of tasks, manages a workspace of documents for each task, and offers a uniform interface for activating development tools in order to carry out technical activities. Finally, the modeling environment is used to adapt the management system to a specific application domain. So far, we have studied applications in mechanical and software engineering; our current work also addresses chemical engineering.

The data maintained by the management system and the operations performed on these data are fairly complex. This calls for a formal specification at a high level of abstraction. We have selected attributed graphs as the underlying data model because they are ideally suited for representing management data such as version histories, configurations of interdependent documents, and task nets. A programmed graph rewriting system serves to specify operations on these graphs in terms of complex graph transformations. Management tools may be generated from this operational specification, avoiding the need for coding in a conventional, rather low-level programming language.

This book is composed of four parts. Part I introduces basic notions such as development, process, or management. Furthermore, it provides an overview of our approach to the management of development processes and compares it to related work.

Part II surveys the current state of the art. We draw a "grand picture" of models and tools for process management. To organize the discussion, we present taxonomies for classifying and comparing existing approaches. Furthermore, we apply these taxonomies to sample sets of process management systems in order to illustrate the spectrum of approaches developed in this field. Finally, we also attempt to assess the current state of the art.

Part III summarizes our work in SUKITS, an interdisciplinary project that was carried out by computer scientists and mechanical engineers at Aachen University of Technology. Its overall result was a management system which was applied to mechanical engineering within the project, but can be applied to other application domains as well. The management system supports integrated management of products, activities, and resources and takes various aspects of dynamics into account (in particular, product-dependent task nets, feedback, and simultaneous engineering). The management system was fully implemented, and it was successfully applied to non-trivial scenarios.

Part IV presents our ongoing work toward a universal and adaptable management model. This work was carried out in the final stages of SUKITS and subsequently in the IMPROVE project (a Collaborative Research Council dealing with development processes in chemical engineering).

This book is a revised version of my habilitation thesis. Many people have contributed to the work presented here. Prof. Manfred Nagl has been advising me for more than a decade. During this period, we have had many fruitful discussions; I have benefited much from his continuous inspiration. Prof. Carlo Ghezzi (Politecnico di Milano, Italy) and Prof. Theo Härder (University of Kaiserslautern, Germany) both agreed spontaneously to act as co-advisors in spite of their heavy workload.

My thesis was carefully reviewed for publication in the LNCS series. In particular, the review helped me considerably in improving the motivation for my work.

Prof. Andy Schürr has been a friend and colleague for a long time. My work on practical applications of graph rewriting is hardly conceivable without his contributions. In 1995, I spent a sabbatical at NTNU in Trondheim, Norway. This was the beginning of a fairly successful cooperation with Prof. Reidar Conradi

who provided me with many new insights into software configuration management. Several colleagues, students, and programmers have contributed to the work described in this book. In particular, I would like to thank Marita Breuer, Peter Heimann, Gregor Joeris, Dr. Carl-Arndt Krapp, Sven Krüppel, and Ansgar Schleicher. I would also like to thank all members of our group who have not been directly involved in my work. Each of them has assisted me in some respect, and they also created a good working atmosphere.

Finally, I would like to thank my wife Monika for her constant support and understanding. Moreover, I am indebted to my parents and my sister Anni. In particular, this book is dedicated to my father who has always supported and encouraged me.

Aachen, April 1999