Modeling Software Processes and Artifacts

Introduction to ECOOP'97 Workshop 10

Jyväskylä, Finland, June 9, 1997

Klaas van den Berg

Twente Research & Education on Software Engineering
Faculty of Computer Science, University of Twente
P.O. Box 217, 7500 AE Enschede, the Netherlands
e-mail: vdberg@cs.utwente.nl

1 Introduction

The workshop on *Modeling Software Processes and Artifacts* has been attended by 16 researchers from 10 different countries. It explored the application of object technology in process modeling. The workshop was organized by *Mehmet Aksit*, *Klaas van den Berg* and *Pim van den Broek* (University of Twente), *Leon Osterweil* (University of Massachusetts), *Karl Lieberherr* (Northeastern University) and *Francesco Marcello* (University of Pisa), and was chaired by Mehmet Aksit. After the introduction by Klaas van den Berg and the invited lecture by *Reidar Conradi* (Norwegian University of Science and Technology), a number of participants presented their position papers and there was ample time for discussion. In this introduction, we first give an overview on some background work, and describe the aims, aspects and approaches in software process modeling, and then we shortly comment on the position papers in this reader.

2 History

Software process modeling has relatively a short tradition. Nowadays, there are several workshops and conferences being held yearly. The first international workshop ISPW-1 was held at Runnymede, UK, in 1984 [17]. At ISPW-6 in 1990 [11], an example was presented of a software process, which is being used as baseline for the comparison of software process models and environments. The first international conference on software process modeling, ICSP-1, was held at Redondo Beach, CA, USA, in 1991 [16]. European workshops on software process technology started at Milan, Italy, with EWSPT-1, also in 1991 [15]. Conferences on software process improvement SPI are closely related to the area of process modeling. At the International Conference on Software Engineering ICSE, there are also regularly contributions on process modeling topics. Some special issues of magazines were devoted to this area (1991, *Software Engineering Journal* 6(5); 1993, *IEEE Transactions on Software Engineering* 19(12)). Although there is no long research tradition in this area, there is already a vast amount of literature available and many research groups are active [18]. In Europe, the PROMOTOR Working Group co-ordinated several projects [8]. We briefly now describe some modeling issues, serving as background to the current workshop.

3 Aims

Curtis et al. (1992) [6] give the following overview with objectives and goals of software process modeling:

- Facilitate human understanding and communication: requires that a group is able to share a common representational format
- Support process improvement: requires a basis for defining and analysing processes
- Support process management: requires a defined process against which actual process behaviours can be compared
- Automatic guidance in performing process: requires automated tools for manipulating process descriptions
- Automatic execution support: requires a computational basis for controlling behaviour within an automated environment.

Obviously, automated process enactment requires a more detailed and formalised model than a model just aiming at human understanding the process.

4 Aspects

The goal of process improvement has been incorporated in the SEI Software Maturity Model and several key process areas have been identified [14]. These areas can be classified in addressing the following categories or aspects:

- Managerial, such as project planning, subcontract management.
- Organisational, i.e. process definition, change management.
- Engineering aspects: requirement analysis, design, coding, testing, etc.

It is clear that these aspects are present in software process modeling in general, as in the reference model for process technology presented by Christie et al. (1996) [5]. They distinguish four main elements: the enterprise operations, the process development, the enactment technology and the process assets. The enterprise operations deal with the process effectiveness of organisations, and the technology that exists to support that effectiveness. The process development deals with the construction of process technology assets. These assets support the organisational process activities. The enactment technology deals with the technology components that need to be in place for the construction of effective process enactment systems. The process assets are the parts in the process, which have to be designed for reuse and placed in an asset library.

5 Approaches

Basic concepts in software process modeling are the artifacts (i.e. the (sub-) products) and activities or process steps, which produce externally visible state changes to the artifacts [4][7]. Another important concept is the meta-process: the part of the process in charge of maintaining and evolving the whole process, i.e. the production process, its meta-process, and the process support.

As in traditional modeling techniques, one can focus on the data in the process, the artifacts, or on the transformation functions, the activities. Products and processes are dual entities. Some approaches in software modeling are process-centered and other

approaches are product-centered. The relative merits of both approaches may become apparent in applying the approach to the Software Process Example [11].

6 Issues

Two of the key process areas in the capability maturity model at level four are the Quantitative Process Management, aiming at controlling the process performance quantitatively, and Software Quality Management, aiming at a quantitative understanding of the quality of the software products [14]. Attributes of process and product entities have to be identified and measures have to be defined and validated. However, many attributes are inherently uncertain [10]. Moreover, there are many ambiguities in the process steps. The quantitative support has to cope with uncertainty and ambiguity. Software products are evolving rapidly due to changing requirements, requiring a high adaptability of the products and composability of solutions. The productivity in software development heavily relies on the reusability of products and subproducts on all levels of the development: not only reuse of code, but also of design (for instance by design patterns), frameworks and software architectures. The products must be very adaptable to facilitate customization to new requirements.

7 Object Technology

Object technology has been used in process modeling at various levels. In an assessment of project within the PROMOTOR Working Group [8], it appears that object technology is used in the modeling phase of the basic components, in process modeling languages, in supporting tools and enactment engines, and in meta-processes. Object technology has some obvious advantages [3]. Objects provide structural and behavioural views of the system architecture; they provide reusability and encapsulation in design methods, and concurrency in complex systems. However, there are several obstacles in using traditional object oriented techniques [2]. Composition filters provide composable solutions [1], notably on concurrency and synchronisation.

8 Themes and Position Papers

In this workshop we consider software process modeling in the context of object technology. We can distinguish the role of object technology in the actual software development (analysis, design, implementation, etc.) as well as in the modeling and support of this development process. Both can be object-oriented or not. We focus on the OO-modeling and support of the software development process, on which OO-techniques are being used for what purpose, their strengths and weaknesses.

In the invited lecture, *Reidar Conradi* and *Chunian Liu* give an overview of process-centered software engineering environments in the context of software process improvement. *Francesco Marcelloni* and *Mehmet Aksit* address the problem of uncertainty in object-oriented methods. They propose the use of fuzzy logic to handle this problem. *Wiebke Reimer* and *Wilhelm Schäfer* describe - in the position paper presented by *Thomas Schmal* - an extension to the object-oriented process modeling language ESCAPE, which copes with process and configuration management. *Pavel*

Hruby discusses the advantages of a product based development process. In the paper on active software artifacts, *Mehmet Aksit et al.* propose a framework for software production in which decisions made by software developers are incorporated into the software itself - the active artifacts - making the software more reusable and adaptable. *Jean Bézivin* uses object composability techniques in the definition of process models. *Jun Han* discusses the integration of process and artifact modeling using object technology. Finally, *Ilia Bider* and *M. Khomyakov* include business processes in the object-oriented process model to handle dynamic and distributed planning.

9 References

- [1] Aksit, M, Wakita, K., Bosch, J., Bergmans, L., & Yonezawa, A. (1993). *Abstracting Object Interactions using Composition Filters*. In ECOOP '93, Workshop on Object-Based Distributed Programming, LNCS 791, pp. 152-184, Springer, Berlin.
- [2] Aksit, M. & Bergmans, L. (1992). *Obstacles in Object-oriented Software Development*. OOPSLA '92, ACM SIGPLAN Notices Vol. 27, No. 10, pp. 341-358.
- [3] Aliee, F.S. & Warboys, B.C. (1995). *Applying Object-Oriented Modelling to Support Process Technology*. Proceedings IDPT-Vol. 1 A. Ertag et al., Eds., Austin Texas.
- [4] Conradi, R., Fernström, C. & Fuggetta, A. (1994). *Concepts for Evolving Software Processes*. In: Finkelstein et al. (1994), pp 9-31.
- [5] Christie, A.M., Earl, A.N., Kellner, M.I. & Riddle, W.E. (1996). *A Reference Model for Process Technology*. In: Montangero (1996), pp. 3-17.
- [6] Curtis, B., Kellner, M.I. & Over, J. (1992). *Process Modeling*. Comm. ACM, Vol. 35 No 9, pp. 75-90.
- [7] Feiler, P.H. & Humphrey, W.S. (1992). *Software Process Development and Enactment*, (CMU/SEI-92-TR-04, ADA258465). Pittsburgh, PA: SEI, Carnegie Mellon University.
- [8] Finkelstein, A., Kramer, J. & Nuseibeh, B. (1994). *Software Process Modelling and Technology*. Wiley, New York.
- [9] Fuggetta, A. & Wolf, A. (1996). *Software Process*. Wiley, Chichester.
- [10] Huff, K.E. (1996). *Software Process Modeling*. In: Fuggetta & Wolf (1996), pp. 1-24.
- [11] Kellner, M.I., Feiler, P.H., Finkelstein, A., Katayama, T., Osterweil, L.J., Penedo, M.H. & Rombach, H.D. (1990). *Software Process Example*, Proceedings ISPW-6, T. Katayama Ed., IEEE Computer Society Press.
- [12] Montangero, C. (1996). *Software Process Technology*. Proceedings 5th European Workshop, EWSPT'96, LNCS 1149, Springer, Berlin.
- [13] Lonchamp, J. (1994). *An Assessment Exercise*. In: Finkelstein et al. (1994), pp. 335-356.
- [14] Paulk, M. C. et al. (1993). *Capability Maturity Model for Software*, (CMU/SEI-93-TR-24), *Key Practices of the Capability Maturity Model*, (CMU/SEI-93-TR-25), Version 1.1, Pittsburgh, PA: SEI, Carnegie Mellon University.
- [15] *Proceedings of the First European Workshop on Software Process Modeling* (1991), Milan, Italy.
- [16] *Proceedings of the First International Conference on the Software Process* (1991), IEEE Computer Society, Washington, DC.
- [17] *Proceedings of the First International Software Process Workshop* (1984), ISPW-1, Runnymede, UK, IEEE Computer Society.
- [18] See for instance: <http://hamlet.cogsci.umassd.edu/SWPI/1docs/SPResearch.html>