Light-weight Approach for Software Process Modeling – A Case Study

Tuomas Mäkilä[1], Antero Järvi[1] and Luka Milovanov[2]
[1]University of Turku, Department of Information Technology, Turku, Finland
[2]Plenware Oy Ltd., Embedded Systems, Turku, Finland
tuomas.makila@utu.fi, antero.jarvi@utu.fi, luka.milovanov@plenware.fi

Abstract

Formal software process modeling languages like Software Process Engineering Metamodel (SPEM) together with proper tool support, provide an efficient way to maintain and modify process definition content. In order to adopt a process modeling technology, an organization has to model its current processes. This is not a straightforward task since existing process definitions may be nonexistent, incomplete or incompatible with concepts of a modeling language, and the actual process may vary considerably from the documented one. In this paper, we describe an approach for conducting the as-is process modeling work. As a proof of concept we present a case study on process modeling carried out in the Gaudí Software Factory.

Keywords

Software Process Modeling, SPEM, EPF, Agile Methods

INTRODUCTION

Software process modeling technology is advancing rapidly and gaining wider acceptance in the industry [1,2]. The Software Process Engineering Metamodel (SPEM) standard of the OMG has defined a unified way to model software development processes. Current SPEM standard is at version 1.1 [3], version 2.0 [2] is not released at the time of writing this paper, but has reached final adopted specification phase and will be released in the very near future. The success and penetration of a standard is crucially dependent on the availability of mature tools. In this research we have used an open-source process authoring tool provided by the Eclipse Process Framework (EPF) project [4]. The EPF Composer tool supports all essential SPEM 2.0 modeling mechanisms, although it is not fully SPEM compliant. Also commercial SPEM based process modeling tools are available.

Formal process modeling with suitable tools enhances existing process practices by reducing process management costs and increasing process flexibility, and creates new innovative ways of utilizing processes for supporting development work and project management [5]. A common requirement for realizing these benefits is bringing processes closer to the developers' world and thus enabling to narrow the gab between the defined process and the actual process followed by the developers.

Depending on how the process models will be used in the organization, the required accuracy, level of detail and frequency of modeling work varies. However, the more often we refine or supplement the models, the better match we can sustain between the models and the actual process. Thus modeling has to be seen as an ongoing activity, not as a single project which creates an everlasting process definition. There is a need for a modeling approach that is light-weight and can be executed in parallel with the development work.

In this article we propose an approach for modeling the software development processes using the SPEM language. Prerequisite for the modeling is the existence of explicit or implicit process documentation which can be incomplete, outdated or incompatible with the SPEM language.

MODELING APPROACH

Although there is active research on process modeling, the focus is on the models. Research on the methods of creating the models is rare. Schwegmann and Laske present a procedure for as-is business modeling in [6]. The procedure includes following high-level activities: preparation of as-is

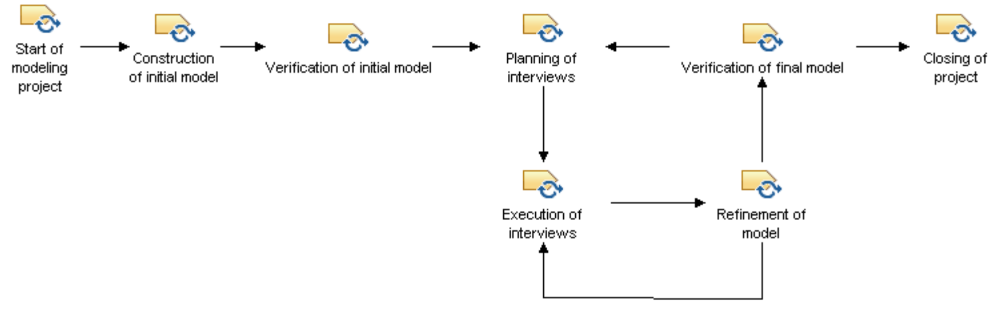
Start of modeling project
Construction of initial model
Verification of initial model
Planning of interviews
Verification of final model
Closing of project
Execution of interviews
Refinement of model

Figure 1 Illustration of the modeling approach

modeling, identification and prioritizing of problem areas, collection and documentation of as-is models and consolidation of as-is models. Kellner and Hansen [7] describe a process modeling project for SPI purpose. Although their focus is on the contents of produced models, some useful information on how the actual modeling should be done can also be found.

Requirements for Approach

To define the concept of light-weight modeling, we set the following requirements for our proposed modeling approach:

1. *Fast modeling*: the development processes shall be modeled effectively so that the focus can be kept on the use of the models rather than on the definition of the models.
2. *Low interference with development*: the modeling shall not affect the normal development activities more than is absolutely necessary. This makes it possible to model the software development process in parallel with the on-going development projects.
3. *No need for extensive consultation*: the modeling shall be possible to execute by following the written modeling approach description and general modeling guidelines. This makes industrial adoption more plausible and takes also small organizations into consideration.
4. *Technology independence*: modeling shall be possible for all kinds of software development processes, regardless of the methods they are based on.

The modeling of processes can be a complex task and consume a large amount of resources if done thoroughly [6,7]. Therefore, we must also accept some process simplifications in order to speed up the modeling and reduce the interference with development projects. On the other hand, based on the findings of Kellner and Hansen, a too low number of interview iterations will reduce the accuracy of the final process model [7][p.23]. Their findings also suggest that processes which have been executed repeatedly are easier to model.

Our modeling approach suites well for the cases where the process is well-established and a consensus of work practices exists within the development organization. In this case, the number of costly interview iterations can be kept minimal while still yielding sufficient accuracy of the model.

Modeling Process

Our modeling process consists of seven steps which are executed sequentially. The high-level overview of the approach is illustrated in Figure 1.

Start of modeling project. Plan the modeling project and communicate the plan to all participants. There should be one lead modeler who is responsible for the modeling work. The lead modeler should have experience in software process modeling. Also the purpose of the final model should be specified, because it defines the level of detail of the model.

Construction of initial model. The initial process model is constructed based on the existing process documentation. The initial model acts as a starting point for modeling the current process. It can also work as a reference when comparing the differences between the documented and actual processes. If no explicit process documentation exists, implicit process documentation, i.e. guidelines and templates of the development organization, should be used.

Verification of initial model (Optional). Authors of the existing documentation, namely *process experts*, should verify that the model is in line with the existing process documentation and that the modelers have understood the documentation right. The model should be refined during the verification meeting, so that no additional meetings are needed. Time-consuming, detailed changes can be done after the verification.

Planning of interviews. The interviews serve several purposes. Firstly, to clarify unclear parts of the process model. Secondly, to find contradictions between the process documentation and the actual process. Thirdly, to find differences on how different process stakeholders use the process. The interviewees should be chosen so that they represent different process stakeholders in as broad way as possible.

Execution of interviews. Minor modifications to the model should be made interactively with the interviewees to illustrate and verify the model changes immediately. If possible, notes should be made from the comments of the interviewee.

Refinement of model. Model should be refined right after or even during the interviews based on the interviewee comments. A proper modeling tool is essential since it allows fast, interactive and flexible modeling.

Verification of final model. The purpose of the step is to verify that the modelers have understood the interviewees correctly and validate that the model is accurate enough for indented use. If there are major contradictions between participants, the interviews should be re-planned and repeated.

Closing of modeling project. The lead modeler should create a baseline of the model and make sure that it is stored properly and distributed to all stakeholders.

CASE STUDY AND RESULTS

Modeling Environment

The Gaudí Software Factory [8] is a software construction unit at the department of Information Technologies at Åbo Akademi University. The goal of the Gaudí factory is to produce software for the needs of various university research projects. Software process in the Gaudí factory is based on agile methods, particularly on Extreme Programming [9]. It is characterized by short (one or two weeks) iteration cycles followed by small releases, and close involvement of a customer [10] or customer proxy [11] in software projects. The Gaudí process is defined as a collection of so-called software best practices focusing on product quality and project agility [12]. Some of the practices were directly borrowed from Extreme Programming, while the most of them were adapted to suit the university environment [13].

Modeling Steps

Besides the description of the software best practices which comprise the Gaudí process, there is no other description of the process in a conventional way. However, a need for a clearer description of the Gaudí process has been arising, mainly because of two reasons. Firstly, due to the high staff turn-around [8] in Gaudí, there is a need for clear process description to teach the process to new programmers, customers and coaches. Secondly, the Gaudí process is supposed to be very flexible and allow tailoring – an accurate process model would be a great help in this task. Therefore, the Gaudí factory makes a good pilot study for testing our approach for software process modeling.

The initial model was created based on the technical report on the Gaudí Software Factory [12]. After completing the initial model, it was verified by interviewing the process expert who was one of the original authors of the Gaudí process description. Also the key stakeholders of the process were identified and interviews with them and modeling team were scheduled. The key stakeholders, a developer and a coach from different projects, were interviewed about the actual hands-on execution of the process during the development projects. The interviews revealed new details about the process, problems in the initial work flow, and elements that were over-emphasized in the original process description. The interviews also complemented each other. The developer had more information about actual development tasks while the coach knew better about the managerial issues. This is of course quite obvious finding, but shows that making a comprehensive model with low effort is possible only if the interviewees are selected carefully at the beginning of the modeling project. An initial model was modified during the interview session based on the comments of the interviewees. Notes were also made from the interviewees' feedback to be used in subsequent steps.

After the interviews, the draft model was modified based on the interview notes. This model was again verified by the process expert and the relevance of the model was evaluated. Some minor adjustments were made, but in general the model was satisfying. The final model is presented in the next section.

Final Model

Interviews revealed that there are two kinds of projects in the Gaudí factory. Therefore the main iteration was divided into two different models: customer driven and technical coach driven. The former process model is used in projects with clear customer role. In this case separate acceptance testing is done and there is need for release builds. This model can be seen in Figure 2. The latter process model is used when the coach of the project also works as a customer proxy. In this case the role of the coach resembles more of the traditional project manager role. Because the customer proxy i.e. the coach works closely in the project, separate acceptance testing is not needed. Instead, the coach evaluates the functionality of the software constantly by doing exploratory testing. This model is illustrated in Figure 3.

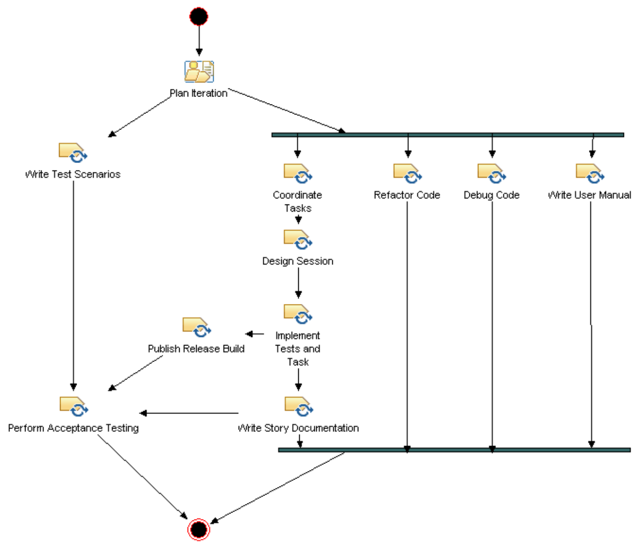

Figure 2 Modeling step 4 a: Final model for the customer driven projects.

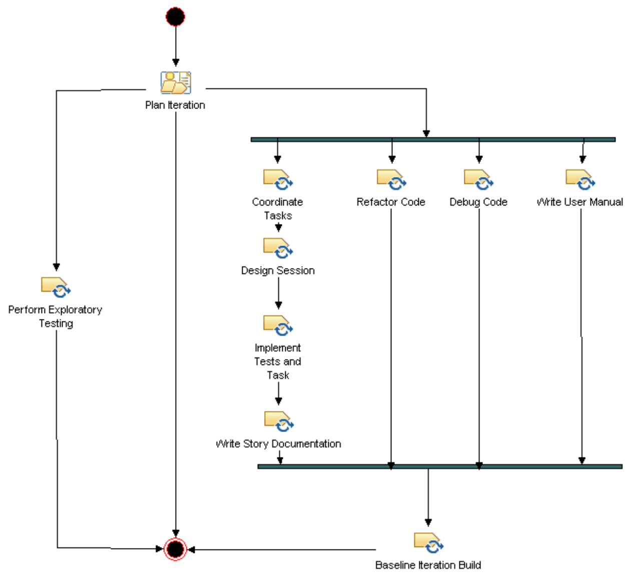

Figure 3 Modeling step 4 b: Final model for the technical coach driven projects.

Table 1 Estimated effort of the modeling participants in hours. Legend: LM = Lead modeler, AM = Assistant modeler, PE = Process Expert, C = Coach, D = Developer.

Step	LM	AM	PE	C	D	Total
Project Start	1	1	1	-	-	**3**
Initial Modeling	15	-	-	-	-	**15**
Verification of Initial Model	1.5	1.5	1.5	-	-	**4.5**
Interviews Planning	0.5	-	0.5	-	-	**1**
Interviews Execution	3	3	-	1.5	1.5	**9**
Model Refinement	3	3	-	-	-	**6**
Verification of Final Model	1.5	1.5	1.5	-	-	**4.5**
Project Closing	1	-	-	-	-	**1**
Total	**26.5**	**10**	**4.5**	**1.5**	**1.5**	**44**

Analysis of the Modeling Experiment

The model of the Gaudí software process presented in the case study turned out to be an accurate enough for the selected purpose. The SPEM diagrams of the model give a clear overview of the Gaudí process to process owners and software engineers. The model captures most of the process activities.

Finally, it is significantly faster to capture the basics of the Gaudí process with the presented diagrams, rather than reading the existing 27-page Gaudí document [12].

Since the proposed modeling approach is straightforward there were neither major delays nor problems during the modeling. We were able to follow the proposed modeling approach quite faithfully. The total work effort was approximately one man-week distributed into one month's period. The details on the modeling effort are presented in Table 1.

CONCLUSIONS

In this paper we have presented an approach for conducting as-is process modeling. The presented approach is a fast and light-weight way to model the existing software process of an organization, when some process documentation exist.

As a proof of concept we have also presented a case study on process modeling carried out in the Gaudí Software Factory. As a result of this case study, we have seen that the proposed approach works in practice. Furthermore, the approach turned out to satisfy our preset requirements as it is fast, has low interference with development, does not require extensive consultation and is methodology independent. In addition, the resulting process model showed directions for the process improvement in the Gaudí factory.

The modeling approach proposed in this paper resulted in two final models for Gaudi factory: the *customer-driven* and *coach-driven* models (see Section 3.3). These two types of the process have been established in the Gaudí factory since its start, but were never implicitly documented anywhere. This discovery speaks for the accuracy and objectivity of the proposed modeling approach in the presented case study and shows the ability of bottom-up modeling to reveal existing, undocumented high-level development strategies.

However, the final model is not complete. For example, the model does not include full work-product and role descriptions. In fact, we do not see the need to model comprehensive work-product flow because of the agile nature of the Gaudí process. Nevertheless, the model is accurate and we believe that complemented with essential work-product templates and role descriptions, the model would serve as good process documentation in the Gaudí factory.

While Gaudí factory offers a fine, controlled environment to experiment with the process modeling techniques, we will also test the proposed modeling approach in purely industrial settings. The problems with process modeling also apply to the purely commercial software development, namely how process documentation can be supported by modeling techniques and how models can be flexibly tailored to fit different development needs. The forthcoming industrial cases will represent different development methods, so that technological independence of the modeling approach and its applicability in different situations and organizations can be verified.

REFERENCES

[1] Peter Haumer. Second revised spem 2.0 submission. OMG Meeting, 2006.

[2] Object Management Group. Software Process Engineering Metamodel Specification, v2.0, March 2007. ptc/07-03-03.

[3] Object Management Group. Software Process Engineering Metamodel Specification – Version 1.1, January 5 2005. formal/05-01-06.

[4] Eclipse process framework project homepage. http://www.eclipse.org/epf/. Accessed on March 16 2007.

[5] Antero Järvi, Tuomas Mäkilä, and Harri Hakonen. Changing role of spi - opportunities and challenges of process modeling. In Proceedings of the 13th European Conference on Software Process Improvement (EuroSPI 2006), volume 4257 of Lecture Notes in Computer Science, pages 135 – 146. Springer Berlin / Heidelberg, October 2006.

[6] Ansgar Schwegmann and Michael Laske. Process Management - A Guide for the Design of Business Processes, chapter As-is Modeling and Process Analysis, pages 107 – 133. Springer, 2003.

[7] Marc I. Kellner and Gregory A. Hansen. Software process modeling. Technical report, Software Engineering Institute, May 1988.

[8] Ralph-Johan Back, Luka Milovanov, and Ivan Porres. Software development and experimentation in an academic environment: The gaudí factory. Journal of Systems and Software, 2007. To appear.

[9] Kent Beck. Extreme Programming Explained: Embrace Change. Addison-Wesley, 1999.

[10] Ralph-Johan Back, Piia Hirkman, and Luka Milovanov. Evaluating the XP Customer Model and Design by Contract. In Proceedings of the 30th EUROMICRO Conference. IEEE Computer Society, 2004.

[11] Piia Hirkman and Luka Milovanov. Introducing a Customer Representative to High Requirement Uncertainties. A Case Study. In Proceedings of the International Conference on Agile Manufacturing, 2005.

[12] Ralph-Johan Back, Luka Milovanov, and Ivan Porres. Software development and experimentation in an academic environment: The gaudi experience. Technical Report 641, TUCS, Nov 2004.

[13] Luka Milovanov. Agile Software Development in an Academic Environment. PhD thesis, TUCS, Dec 2006.