Fraunhofer Institut
Experimentelles
Software Engineering

Towards Systematic Knowledge Elicitation for Descriptive Software Process Modeling

Authors:
Ulrike Becker-Kornstaedt

Accepted for publication in
Proceedings of PROFES 2001

IESE-Report No. 036.01/E
Version 1.0
September 2001

A publication by Fraunhofer IESE

Fraunhofer IESE is an institute of the Fraunhofer Gesellschaft.
The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competetive market position.

Fraunhofer IESE is directed by
Prof. Dr. Dieter Rombach
Sauerwiesen 6
D-67661 Kaiserslautern

Abstract

Capturing a process as it is being executed in a descriptive process model is a key activity in process improvement. Performing descriptive process modeling in industry environments is hindered by factors such as dispersed process knowledge or inconsistent understanding of the process among different project members. A systematic approach can alleviate some of the problems. This paper sketches fundamental difficulties in gaining process knowledge and describes a systematic approach to process elicitation. The approach employs techniques from other domains like social sciences that have been tailored to the process elicitation context and places them in a decision framework that gives guidance on selecting appropriate techniques in specific modeling situations. Initial experience with the approach is reported.

Table of Contents

1 Introduction

Descriptive software process modeling attempts to determine the actual processes used in an organization [1]. Descriptive process models are key assets in the context of software process improvement programs. They describe current development practices within an organization at a rather high level of detail, and consequently can help detect weaknesses in the process [2].

A major quality criterion for a descriptive software process model is its *accuracy* (i.e., the degree to which the model reflects the actual process), so that any process-related activities can be based on the process as it is carried out. A Process Engineer who uses an inaccurate process model, for instance as a basis for a measurement program, may try to measure activities that are only depicted in the model but do not take place in reality.

Process elicitation in industry is complicated by several factors, such as a high number of different roles in the process, their limited availability, or dispersed process knowledge. However, little research has been done in developing methods for capturing descriptive process information.[1] A first step towards developing such an approach is to adapt techniques developed for similar areas to the software process elicitation context. To guide a Process Engineer in the usage of these techniques, a decision framework is required. This paper presents the development of an approach to systematic process elicitation and first validation results.

The paper is structured as follows: Section 2 gives an introduction to descriptive process modeling and describes the difficulties a Process Engineer encounters in capturing real-world processes. Section 3 proposes the structure of a systematic approach to process knowledge elicitation and introduces its components. Section 4 presents a brief validation of the approach. Section 5 discusses related work. Finally, Section 6 concludes with a summary and outlook.

2 Descriptive Process Modeling

This section sketches the context in which descriptive software process modeling is usually performed in industrial environments.

2.1 Context of Process Modeling

Usually, process modeling is part of a larger process improvement program. The model may be used as a foundation for measurement, to detect weaknesses in the process, or to design a new process based on current practices. Typically, process modeling is performed by a Process Engineer external to the target organization.

Descriptions of the *actual process* [3], i.e., descriptions of how the development process actually takes place in reality, are important assets in software engineering in general and in software process improvement in particular. One key concept for process improvement is to incrementally incorporate experience gained and lessons learned from carrying out the process. Capturing the process as is in an explicit description, in a descriptive software process model, is therefore a first step. Changes to this as-is process can then be incorporated into the model, and the updated and improved process can be disseminated to process participants [2]. The model can be used for a detailed process analysis to detect weaknesses, to define measurement points in the context of measurement programs, to help define metrics and facilitate data collection, or to capture experience. An explicit description of the process is also an important basis for audits, assessments, or certifications. Figure 1 explains the role of descriptive software process models in process improvement and the major steps in descriptive software process modeling, elicitation of process knowledge, creation of the model, and review of the model [3].

The schema presented in [4] names the following information to be described in a process:

- activities, i.e., what is being done,
- artifacts, i.e., what is being used and produced, and
- agents, i.e., descriptions of the responsibilities.

In addition to these, relationships between them, such as product flow, role assignment, or refinement have to be described.

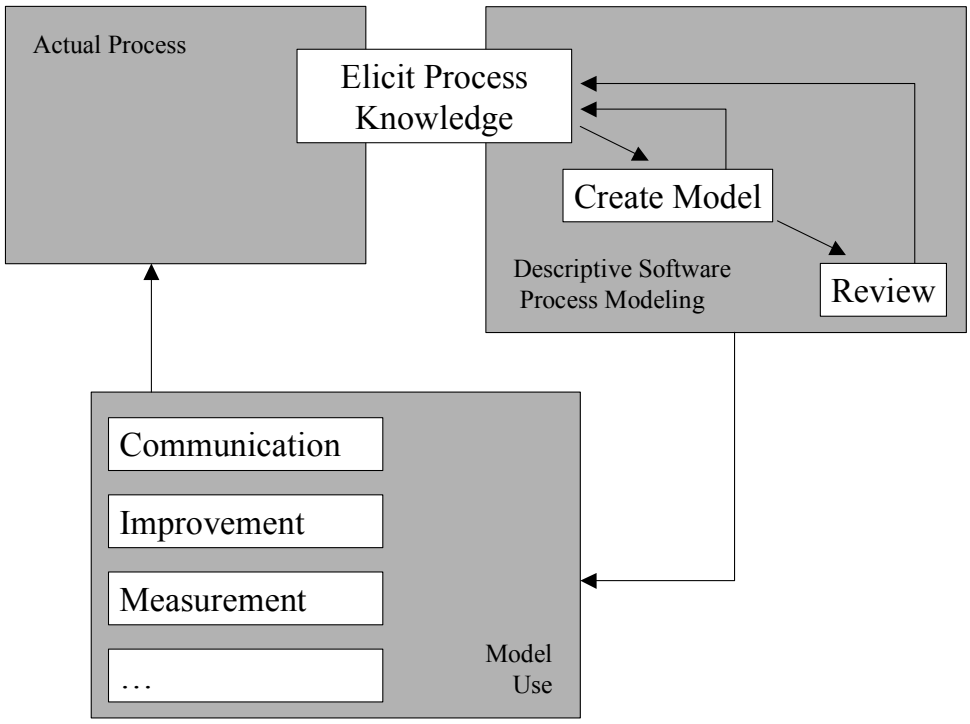

Figure 1 Steps in descriptive software process modeling

However, what information needs to be mapped onto the model, or the level of detail to be covered depends on the specific purpose of the model, and its intended users.

The major tasks in developing a descriptive model are elicitation of process knowledge, formalization of the process knowledge to create a model, and review of the model [5]. Usually, there are rework cycles going from model creation back to elicitation, for instance, when important information is missing, and from review to the elicitation step, if it shows that the model is incomplete or incorrect. The elicitation step is the first one in process modeling on which the other steps build. Thus, the result of the elicitation step is crucial.

Typical information sources for software process knowledge reported in the literature are analysis of process artifacts, observation of process performers, and interviews with Process Performers (see, for instance, [3]).

Analysis of Process Artifacts. Analysis of artifacts refers to examination of artifacts produced by a group under study. In the case of process elicitation, two categories of artifacts can be distinguished: Process artifacts and process documentation.

Process artifacts are the artifacts developed as intermediate or final products in the process. Process artifacts provide evidence for the actual process. Examples of process artifacts are requirements or code documents. Process artifacts can give information about themselves, their contents, and their structure. Relationships between the artifacts and the activities producing these artifacts are sometimes included.

Process documentation refers to documents that describe how the process is to be performed and what needs to be done to obtain the expected results. Examples for process documentation are process handbooks or project plans. Process documentation has the advantage that it usually describes the relationships between different information entities, such as which documents an activity should produce, or which role is supposed to perform an activity. However, process documentation gives information on the official process [3] (i.e., the process as it should be performed). This information does not necessarily have to match the actual process.

The major strengths of using existing documents are that they can be reviewed repeatedly, and that they are unobtrusive [6].

Observation of Process Performers. Observation refers to the selection and recording of a set of behaviors in their natural 'environment'. One major problem of observation is that only those events that occur during observation can be captured. In addition, observation is often perceived as obtrusive.

[7] describes how observation of Process Performers was used to find out about the actual process. [8] propose to use data collected on the events of an executed process to discover the behavioral aspects of the process. However, this assumes that people already collect data on the process and know what data to collect.

Interviews with Process Performers. Interviews with Process Performers are flexible with respect to information content and level of detail. However, Process Performers can only report on their view of the process.

2.2 Problems in Process Modeling

Literature sources (for instance [3]) describe possible sources for process information. But they do not provide details on *how* to extract the process knowledge from these sources, what techniques to use to exploit them, or under which conditions to exploit which type of information source.

One characteristic of software processes is that they are creative, human-based processes. Other than most manufacturing processes, where always the same

product is being produced – apart from minor modifications – , software processes deal with individual solutions.

Some of the problems that often make software process elicitation in industrial practice difficult include:

- **P1**: Knowledge about the process is dispersed within the organization due to the complexity of software development. This complexity implies that a high number of Roles is involved, each of them with very specific tasks in the process, and each of them is only involved in a fraction of the overall development process. [9] It is impractical to interview everybody involved in a process. Thus, sampling strategies for interviews are needed.
- **P2**: In large teams, especially different parts of the software process may be performed in different geographical locations. This makes it more difficult to get a consistent and complete picture of the development process.
- **P3**: Depending on their roles in the process, different Process Performers have individual views of the process [9]. Union of the views may lead to an inconsistent global picture. This is especially the case when the process is very complex, and when the different Process Performers are exclusively involved in their parts of the process.
- **P4**: Some process steps are performed very seldom. Observation of such process steps cannot be planned, and Process Performers tend to forget about these steps or are not able to report them accurately.
- **P5**: Process Performers may leave out aspects of the process [10]. Often, Process Engineers may not be aware of getting incomplete information. Thus, it is very important that Process Engineers have some background information so that they can assess the information provided by Process Performers.
- **P6**: Access to Process Experts is difficult to obtain [11]. Experienced Process Performers are not only the most favored interview partners for process elicitation, but the more experienced Process Performers are, the more they are involved in projects, and the tighter are their schedules, and the more difficult it is to arrange interviews with them.
- **P7**: Process Engineers from outside the target organization do not know the organization well enough to judge who is the right expert to provide the information needed. Badly selected interview partners may give inadequate information (such as process steps other than those asked for, or the Process Performer may not be able to provide the knowledge at the level of granularity requested). On the one hand, this is wasted effort for the people interviewed; on the other hand, this usually leads to additional cycles in modeling and review.
- **P8**: External Process Engineers, being unfamiliar with the company-specific terminology, may misinterpret information and develop an inaccurate process model. The more detailed the process information, the more difficult it is to interpret it in the right context.

The problems mentioned above lead to incorrect or incomplete process models, or models that do not fully suit their intended purpose. Such models cause additional effort from both Process Engineers and Process Performers for re-interviews, modifications, and additional reviews.

Thus, process elicitation needs to be performed in a way that is efficient for Process Performers and for the Process Engineer, meaning it avoids inefficient interviews (for instance, because a Process Performer is interviewed who can not give the information required) and rework (i.e., re-interviews, modification, and additional reviews) because of misunderstanding or incomplete information and additional review cycles.

In practice, process elicitation as currently practiced depends very much on the skills and experience of the Process Engineer performing the modeling task. A systematic approach to process knowledge elicitation is needed to alleviate some of the problems mentioned above, and to make software process elicitation efficient for Process Experts and less dependent on the skills and experience of the Process Engineers. The next sections describe some techniques and how they were developed, as well as a decision model that could make them available in practice.

3 Method Design

3.1 Overall Framework

A large number of the problems reported in Section 2 can be alleviated if a Process Engineer already has some background knowledge on the organization before getting involved in details of the process. Thus, process elicitation should be performed in two stages: process familiarization and detailed elicitation [11], as depicted in Figure 2. Figure 2 details the descriptive process modeling step as described in Figure 1. To each of these stages, appropriate techniques have to be associated.

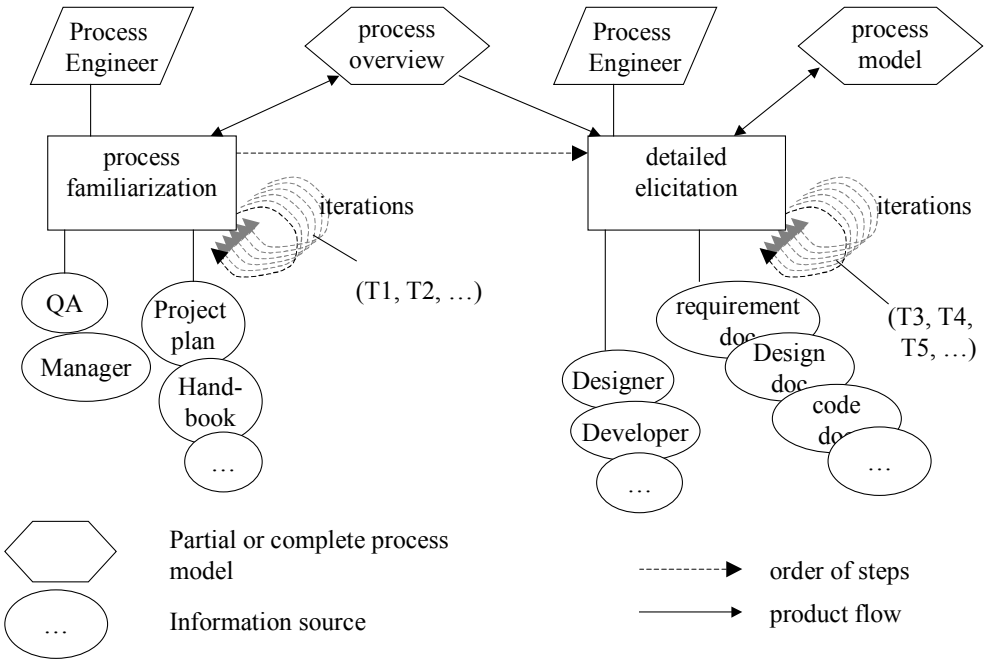

Figure 2 Process elicitation

The aim of the *process familiarization phase* is to obtain an overview of the process and its general structure. Typical human information sources in this situation are Managers and Quality Assurance people. Documents helpful at this stage include process handbooks or project plans that give an overview of the process. In this stage, techniques that give an overview of the process are mainly used for knowledge elicitation (T1, T2, …). The techniques have to be

mainly used for knowledge elicitation (T1, T2, …). The techniques have to be characterized with respect to their scope (i.e., what type of knowledge can they elicit) and under which circumstances they are to be used. Having an overview of the process can facilitate elicitation in the later stages and relieve some of the problems described above. For instance, an overview of the process can help determine which roles need to be asked (P1), or may help the Process Engineer understand where inconsistencies in process understanding result from (P2). Process familiarization can alleviate problems related to not knowing the target organization (P7), (P8). The result of this stage is an overview of the process structure, such as the major activities and artifacts, and possibly the product flow among them.

In the second phase, the *detailed elicitation phase*, the Process Engineer tries to obtain details on the process, such as detailed descriptions of activities or artifacts. During this stage, example artifacts from the process can provide valuable information. Typical human information sources are Designers or Developers, i.e., roles that have a very detailed view of their part in the process. Typical documents to look at are example documents, such as requirements documents, design documents, or code documents. Techniques employed in this stage (T3, T4, T5, …) are mostly techniques that allow to obtain in-depth information.

3.2 Method Design

To further fill the framework described in the previous section, concrete techniques have to be defined.

Process elicitation is done over many iterations. In each iteration step the process model is further refined, completed, or inconsistencies are removed from the model. In each iteration step, a Process Engineer must decide what information is missing in the current version of the process model. Accordingly, he has to select a technique and source that allow him to elicit missing information to further complete the process model.

As process modeling projects and their contexts differ a lot from one another regarding issues such as their goals, the requirements for the model, or the environment in which process modeling takes place, it can not be expected that a single procedure or a deterministic algorithm describing the order of techniques to apply will be adequate to suit every process modeling project.

Instead, a systematic approach to process elicitation will rather consist of a set of techniques specifically designed or tailored to software process knowledge, and a set of heuristics or guidelines (collectively called a *decision model*) describing when to apply which technique and what the expected outcome is. For instance, a Process Engineer who has no further information on the overall

process will most likely not be able to use information on the details of a coding activity, as the overall context of the activity is unclear to him.

The major factors determining whether a technique can be applied depends on what knowledge is already available, what knowledge is still missing, and the information sources available.

Thus, to systematically describe a software process elicitation method, the following components are used:

- A schema describing what the building blocks of the method, i.e., the techniques used in the method have to look like. This schema allows to characterize the interfaces between different building blocks, and it allows to extend the method by developing new building blocks.
- A schema that allows to classify process information to describe process models in terms of completeness as well as knowledge that still needs to be elicited.
- Concrete building blocks to instantiate the framework for the method. These building blocks may be domain-specific.

3.3 Techniques

Knowledge elicitation from artifacts and interviews with, or observation of, Process Performers has been done in other disciplines. Qualitative research methods refer to research procedures that produce descriptive data: people's own written or spoken words and observable behavior [12].

Qualitative research techniques have been used for a long time, especially in social sciences. In addition, these techniques have been tailored to other, more technical, contexts, for instance, Software Engineering [13], or in Expert System design. Thus, experience does not only exist with respect to the usage of these techniques, but also regarding the tailoring and adaptation of the techniques to other contexts.

To develop a systematic approach to software process elicitation, techniques from other disciplines have to be (and in part were) explored for their applicability to software process elicitation.

A field that has tailored qualitative research techniques to solve its own problems in a more technical context is the Knowledge Acquisition (KA), i.e., 'the activity of gathering knowledge for especially knowledge-intensive programs'[14].

Techniques that have already been employed in the KA context are therefore good candidates for tailoring to a method for systematic knowledge elicitation in the context of software process modeling.

Process elicitation techniques are not only needed to elicit knowledge from different sources, but also for different stages of process elicitation, requiring different types of knowledge, and different degrees of detail. These adapted techniques take into account the specifics of software processes and software process knowledge.

3.4 Schema for Building Blocks of the Method:

To select the most promising techniques applicable in their current context, taking into account characteristics of the software process and the organization, Process Engineers need heuristics.

The applicability of a technique depends on:

- what stage the technique is intended to cover,
- the type and detail of knowledge that should be elicited in the next step,
- the information sources currently available, and possible techniques to exploit these sources,
- premises for using a technique (for instance, a technique may require a Process Engineer to have background knowledge on the process, as pointed out by P5, or needs to know the organizational structure),
- which role or artifact can provide the knowledge needed (e.g., a Manager or a Tester will not be able to provide detailed information on activities and artifacts related to requirements specification). This may require sampling strategies (see 1) (*Source*),
- possible risks of applying a technique. For instance, when using techniques that involve human experts, they report their own view of the process – which may be different from the actual process (see P3) (*Risks*).

Process knowledge is difficult to describe in terms of completeness. Completeness of process knowledge is difficult to judge, as there is no ‘reference body of knowledge’.

However, what can be described more precisely is the type of information elicited (i.e., can the technique elicit information regarding activities or artifacts) with respect to its usage in the process model. One way to characterize software process knowledge is to map process knowledge onto existing software process modeling schemas, such as the one described and implemented in the Spearmint tool [15]. This schema describes the major information entities of a process model, i.e., activities, artifacts, roles, as well as the relationships prod-

uct flow, role assignment, and refinement of these entities. To further describe a process, it is possible to attach user-defined attributes to the process entities.

Within this framework, process knowledge can be further classified according to semantic properties. For instance, process knowledge could be classified as 'a detailed description of an activity'. Employing such a schema, it is possible to describe in terms of schema elements what information is already captured in the model and what information is missing. In addition, using defined techniques allows tracing the different steps followed in elicitation, and the intermediate results.

An additional benefit of tying the process knowledge to be elicited to a process modeling schema that is implemented in a tool is that the technique allows tracing process knowledge down to the process model. Another extension of the elicitation method is that it can be directly coupled with the tool.

A technique used in the approach could look like the following:

Technique: Focused Discussion based upon Process Model Diagram

Source: Human Process Expert

Stage: Familiarization

Description: Structured interview that focuses on structural properties of the process (e.g., product flow, control flow). During the interview, the Process Engineer develops a graphical representation of the process according to the information provided by the interviewee. The diagram is discussed with the interviewee, and possibly refined further. This technique is applicable to elicit structural properties of the process.

Derived from: Focused Discussion. A focused discussion is a structured interview where the topic of the interview centers on a specific type of information such as situations, artifacts, or diagrams [5],[16]. A focused discussion related to process artifacts can help elicit details regarding the artifact, who was involved in producing the artifact, what are the steps, who uses the artifact as input, etc.

Knowledge required to start: none; this technique is suitable for early stages of process elicitation, to obtain an overview of the process structure.

Type of knowledge elicited: activities, artifacts, product flow, control flow between activities

Granularity: overview of process

Additional material: none

Remark: This technique is especially suited to give an overview of the process structure in the early stages of elicitation. The structure of the process model is a very good starting point to elicit more details on the different process entities.

Risks:
- Process Engineer uses a notation that is not familiar to Process Performer.
- Since the Process Engineer has no background knowledge the Process Performer can dominate the interview.
- Process Performers confound the official process (as may be described in a handbook) with the actual one (the one taking place in reality).
- Interviewees may report the official process, because they think this is what the interviewer wants to hear .

4 Validation of Methodology

When validating a process modeling approach, one has to be aware that the quality of the resulting model and the efficiency of the modeling process may be influenced by factors other than the method itself. One of these factors is for instance the experience of the Process Engineer: An experienced Process Engineer without an explicit method may achieve a model of higher quality than an inexperienced Process Engineer with a sophisticated method.

Thus, the best way to evaluate knowledge elicitation technique is to use case studies. Although case studies cannot prove the applicability of a technique under all possible circumstances, they are a good means to show the capability and deficiencies of a technique in one situation.

4.1 Experience with Approach

This section presents a first case study where process elicitation was done following the approach described above. The process model was to describe the major activities, artifacts, and roles in the development process, as well as product flow and role assignment.

In this case, little background material on the process was available in the beginning. For process familiarization it was therefore essential to first obtain an overview of the structure of the process, i.e., the relevant activities, artifacts, and the product flow between them. Thus, it was decided to use a focused discussion, based on a diagram to obtain background knowledge of the process.

We used this technique in a group interview involving five developers. In the interview, we first asked for the relevant activities, then for each of the activities, the artifacts related to them. For activities and artifacts we had used a very simple notation: paper notes of different shapes. These were put up on a whiteboard. Product flow could be easily drawn between those process entities. Using a whiteboard facilitated discussion and modification of the process model. One problem was that the picture soon became too large and complex, resulting in an unclear, cluttered picture. The technique proved to be very well suited to obtain a good overview of the process structure. In addition, the focused discussion showed to be very useful as a group technique. The Process Engineer was rather the moderator of the group than an enquiring interviewer, as the group discussion had synergy effects. One major advantage of the group discussion was that inconsistencies in the Process Performers' under-

standing of the process (e.g., developers' opinions differed with respect to whether a certain artifact was needed in an activity) could be discussed and resolved directly. The information gained from the group discussion reflects more than one person's view. The group interview also increased the Process Performers' understanding of the process.

[12] describe that a major risk of all group interview techniques is that people cannot be expected to say the same things in a group that they might say in private. When using this technique for Process Elicitation, Process Performers may not report the actual process for fear of consequences when their superiors are present. In our case, this did not happen. Individual interviews that were conducted afterwards did not reveal differences to the process model elicited in the group interview.

One problem we encountered was related to the difference between product flow and control flow that was often confused. Whereas product flow only denotes which artifacts are used and produced in which activities, this does not necessarily determine the exact order, the control flow, of activities. This experience will be incorporated into the next version of the decision model as a risk.

The process knowledge gained using this technique showed to be a good foundation for further detailed interviews. Furthermore, due to the good overview, sampling of interviewees could be done fairly easy – based on their roles in the process (see P1). These roles had been identified in the first stage of process elicitation.

During the detailed elicitation stage, the process model was further refined. It showed especially, that there were very few inconsistencies that had to be resolved, as the overall structure of the model was clear.

5 Related Work

This section describes frameworks that have been developed for the elicitation of software process knowledge. [17] describes the *Elicit* approach, a general strategy to develop descriptive software process models. The approach gives a good general overview of the core steps of elicitation - comprising activities such as *understand the general organization*, *plan the elicitation strategy*, and *elicit the process knowledge* needed. The elicitation step is based on document analysis: reading and identifying the structure of selected process documents. However, Elicit does not take into account that the majority of process information is typically provided by Process Performers. In addition, more detailed techniques are missing in the Elicit approach.

[9] introduces an approach called *Multi-View-Modeling*, consisting of three steps. In the first step, different role-specific process models, so-called *views*, are developed. In the second step, the different views are checked for similarity and consistency. After resolving potential inconsistencies, the views are integrated to form a comprehensive software process model in a third step. However, the approach does not give hints on how to systematically obtain the knowledge needed to create the different views of the process model from the various process experts.

[18] describes a strategy to develop (business) process models in his *process modeling cookbook*. The method suggests that a process modeler should gather as much initial information from conversations with process participants and should leave aside any official documents and verbal support. The process modeling cookbook claims that official process documentation may be misleading. For process elicitation, the author suggests using interviews. However, the techniques described in the cookbook focus on the creation of the model, and not on process elicitation.

[8] suggest capturing data from events that take place during the software process, whereas an event is anything that can be observed in the course of the process. However, the approach does not describe how to systematically evaluate those events to come up with a description of the process.

6 Summary and Outlook

This paper describes the situation in descriptive software process modeling and the need for a more systematic approach to software process elicitation. The design of a more systematic approach is sketched. This approach consists of a set of techniques and a set of selection criteria for each of the techniques. The techniques are adapted from other fields that have already used knowledge elicitation in similar contexts. The selection criteria describe the context under which a technique is likely to be successful, what type of knowledge it is able to elicit, what type of background knowledge is needed to apply the technique, and what are possible risks when applying the technique. The selection criteria take the specific problems of software process elicitation into account.

This approach is a first framework towards more systematic process elicitation. Although software process knowledge cannot be characterized with respect to its completeness, since a reference body of knowledge is not available, knowledge can be characterized with respect to its type, and – to some degree – its level of detail. One means for this characterization is to use an existing process modeling schema. Other factors that have an impact on which technique to apply are formulated as entry criteria and risks.

As software process elicitation deals in large parts with humans, process elicitation will always need a high degree of flexibility. Using this approach allows analysis of the approach and learning. The structure presented here is a first step towards more systematic process elicitation. However, an important aspect will be to incrementally extend the approach. Criteria may have to be revised or refined, as more experience in the application of the techniques is available.

Acknowledgements

The author would like to thank Leif Kornstaedt and Jürgen Münch for the inspiring and fruitful discussions on descriptive software process modeling and systematic process elicitation. Additionally, she would like to thank Sonnhild Namingha for her *systematic* proofreading of the paper.

7 References

[1] Bill Curtis, Marc I. Kellner, and Jim Over. Process Modeling. Communications of the ACM, 35 (9): 75-90, September 1992.K. E. Huff. Software process modelling. In A. Fuggetta and A. Wolf, editors, Software Process, Trends in Software, chapter 1, pages 1–24. John Wiley & Sons, 1996.

[2] K. E. Huff. Software process modelling. In A. Fuggetta and A. Wolf, editors, *Software Process*, Trends in Software, chapter 1, pages 1–24. John Wiley & Sons, 1996.

[3] S. Bandinelli, A. Fuggetta, L. Lavazza, M. Loi, and G. P. Picco. Modeling and improving an industrial software process. *IEEE Transactions on Software Engneering*, 21(5):440–454, May 1995.

[4] J. W. Armitage and M. I. Kellner. A conceptual schema for process definitions and models. In D. E. Perry, editor, *Proceedings of the Third International Conference on the Software Process*, pages 153–165. IEEE Computer Society Press, October 1994.

[5] E. S. Cordingly. *Knowledge Elicitation Principles, Techniques, and Applications*, chapter Knowledge Elicitation Techniques for Knowledge-based Systems, pages 89–172. Ellis Horwood Limited, Chichester, Great Britain, 1989.

[6] R. K. Yin. Case Study Research: Design and Methods. Sage, 2nd edition, 1994.

[7] D. B. Walz, J. J. Elam, and B. Curtis. Inside a software design team: knowledge acquisition, sharing, and integration. *Communications of the ACM*, 36(10):63–77, October 1993.

[8] J. Cook and A. Wolf. Automating process discovery through event-data analysis. In *Proceedings of the Seventeenth International Conference on Software Engineering*, pages 73 – 82. Association of Computing Machinery, April 1995.

[9] M. Verlage. An approach for capturing large software development processes by integration of views modeled independently. In *Proceedings of the Tenth Conference on Software Engineering and Knowledge Engineering*, pages 227–235, San Francisco Bay, CA, USA, June 1998. Knowledge Systems Institute, Skokie, Illinois, USA.

[10] M. I. Kellner and G. A. Hansen. Software process modeling: A case study. In *Proceedings of* the 22nd Annual Hawaii International Conference on System Sciences, volume II, pages 175–188, 1989.

[11] U. Becker-Kornstaedt and W. Belau. Descriptive Process Modeling in an Industrial Environment: Experience and Guidelines. In R. Conradi, editor, *Proceedings of the 7th European Workshop on Software Process Technology* (*EWSPT 7)*, Kaprun, Austria, pages 177–189, Lecture Notes in Computer Sciences, Springer-Verlag. 2000.

[12] S. J. Taylor and R. Bogdan. Introduction to Qualitative Research Methods: A Guidebook and Resource, Third Edition. John Wiley and Sons, 3 edition, 1998.

[13] C. B. Seaman. Qualitative Methods in Empirical Studies of Software Engineering. *IEEE Transactions on Software Engineering*, 1999.

[14] H. Eriksson. A survey of knowledge acquisition techniques and tools and their relationship to software engineering. *Journal of Systems and Software*, (19):97–107, 1992.

[15] U. Becker-Kornstaedt, D. Hamann, R. Kempkens, P. Rösch, M. Verlage, R. Webby, and J. Zettel. Support for the Process Engineer: The Spearmint Approach to Software Process Definition and Process Guidance. In *Proceedings of the Eleventh Conference on Advanced Information Systems Engineering (CAISE '99)*, Heidelberg, Germany, June 1999. Lecture Notes in Computer Science, Springer–Verlag.

[16] N. J. Cooke. Varieties of Knowledge Elicitation Techniques. *International Journal of Human-Computer Studies*, 41:801–849, 1994.

[17] N. H. Madhavji, D. Höltje, W. Hong, and T. Bruckhaus. Elicit: A Method for Eliciting Process Models. In D. E. Perry, editor, *Proceedings of the Third International Conference on the Software Process*, pages 111–122. IEEE Computer Society Press, October 1994.

[18] P. Kawalek. The Process Modelling Cookbook: Orientation, Description, Experience. In *Proceedings of the 2nd European Workshop on Software Process Technology* (*EWSPT 2*, pages 227, 1992.

Document Information

Title:	Towards Systematic Knowledge Elicitation for Descriptive Software Process Modeling
Date:	September 2001
Report:	IESE-036.01/E
Status:	Final
Distribution:	Public