A Meta-Process for Software Reuse Process Discovery and Evolution

Pankaj K. Garg, Mehdi Jazayeri, and Michael L. Creech

Hewlett-Packard Labs
1501 Page Mill Road
Palo Alto, CA 94304
Email: garg@hpl.hp.com

Abstract

Our goal is to develop process models for domain-specific kit-based software production and consumption. For this purpose, we are exploring approaches for: (1) empirically discovering and validating software processes, and (2) monitoring and optimizing (evolving) such processes. In this paper we present an iterative meta-process that represents a first step in this direction. We also report on some early usage of the meta-process for an experimental domain-specific kit production process.

Keywords: domain-specific software kits, process models, meta-processe.

Workshop Goals: to better understand the state of the art in the development of generic software architectures.

Working Groups: domain analysis/engineering, reuse process models.

1 Introduction

One approach to software reuse is to develop application programs from domain-specific software development kits (DSK's). With the use of such kits, generic knowledge about an application domain can be encoded such that the process of creating a new application becomes one of *customizing* the kit rather than *programming*. There are two main impediments to the kit approach: the technology and mechanisms for encoding generic domain knowledge as kits needs to be developed, and software development organizations need to be redesigned to follow a kit-based development and maintenance process. In this paper we address the latter issue.

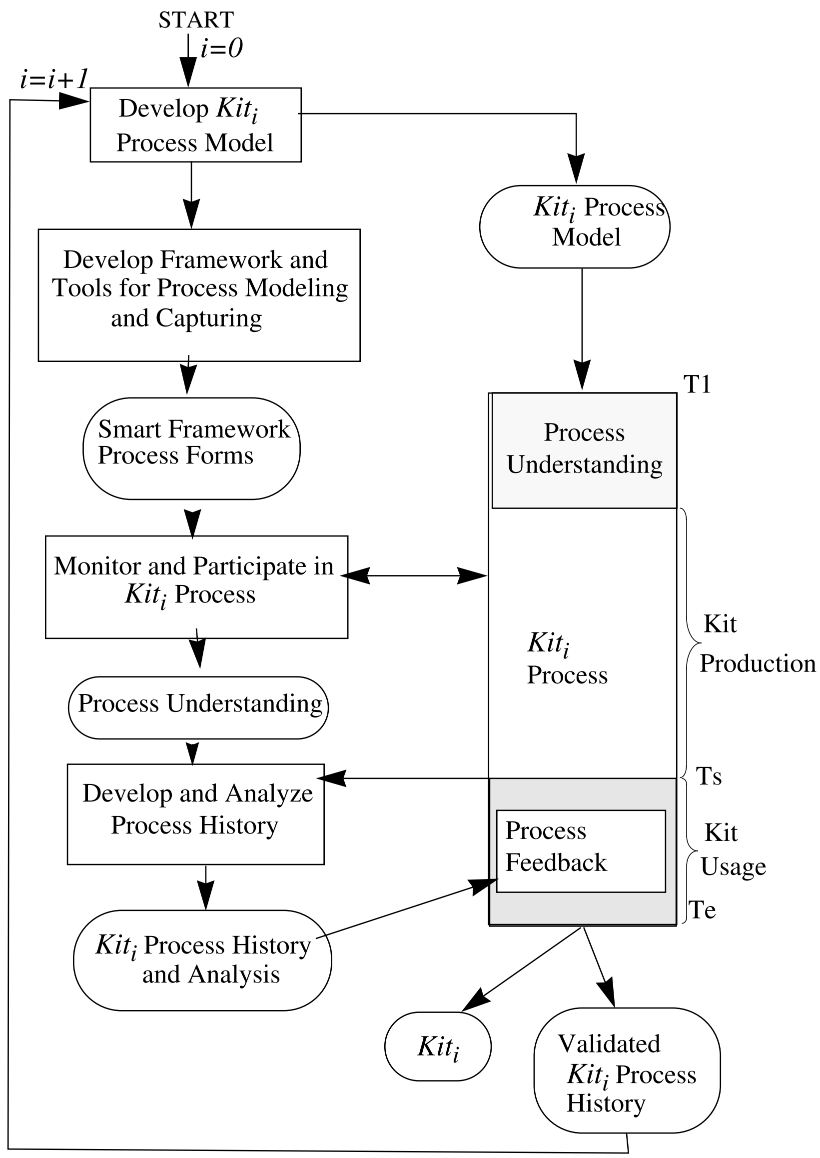

Figure 1: A Meta-Process for DSK process discovery and evolution.

Recent organization design theories suggest that, with the competitive demands of the future, organizations need to be organized around their *core processes* rather than their functional units [6]. Software engineering studies indicate that, to control the runaway costs of software efforts, and to ensure the quality of a software product, the processes by which the software is produced and used need to be explicated [1, 4]. And, finally, explicit process models seem essential for establishing a software reuse program [3]. For these reasons, our work is focused on developing DSK process models.

In order to understand the nature of such processes, our group is experimentally constructing and using example kits over several iterations. In each iteration of a DSK process, we develop prototype process models, along with the development of the next iteration of a kit, such that eventually we can produce empirically validated DSK process models and guidelines that are applicable for organization settings similar to ours. These process models include processes that can be packaged with the kits, to be followed by kit users, as well as process models to be followed by other kit producers.

Figure 1 shows the meta-process that we have adopted for developing the DSK process model. As shown in the figure, we started by participating in the planning of the Kit_0 activity, that led to the development of the initial process model for Kit_0. This model was subsequently used by the kit building team to develop the kit.

While the kit was being developed, we determined a framework for capturing the process of Kit_0. This framework is an adaptation of the knowledge-based Articulator framework [5]. The main feature of the Articulator that was attractive for our application is its *flexibility*. Using the Articulator meta-model, a process modeler can specify the analysis and modeling concepts that will be used to develop the process models, rather than being forced to use a particular set of concepts and method, e.g., the one advocated by tools supporting the IDEF method.

At various stages of the process, we monitored the process to get a feel of what was going on. Ultimately, when we realized that the kit production process was almost complete, at time T_s in Figure 1, we started conducting interviews of the participating people to capture the process history. We analyzed the process history to determine the strengths and weaknesses of the process.

The next stage of the meta-process was to present the process history and its analysis for feedback from the DSK team. As shown in Figure 1, this was necessary to produce a validated process history that can be used to produce a process model for kit_1. One reason for validation is to remove any misinformation introduced while we were developing the history. Another reason for the history feedback was to collect information about the time interval from T_e to T_s that we missed during the interviews. This part of the process was primarily on kit usage. However, most of our time was spent in validating the history for the interval from T_1 to T_s. Therefore, for the next iteration of the meta-process we will be adopting a more continuous monitoring and feedback approach.

2 Process History Elements

The process history is captured using the four elements described below. The words in italics within this description are attributes of the four elements of our process history.

1. Agents - the characterization of the different roles that are executed by the people involved in the process. An agent is characterized by the *Person* represented, the *Role* that he or she plays in the process, the *Skill Set* they bring to the process, the person's *Goal* with regards to that role, and the *Relative Importance* the person attaches to that role.
2. Primary Tasks (P-Tasks) - the planned activities or tasks that were carried out in the process to accomplish the goals of either producing a DSK or using one. Each P-Task has a *Goal* associated with it, lasts for a certain *Duration*, has some people who are *Responsible* for accomplishing the task, some people who are *Participants* in the task, and some people who were *Monitoring* the task. Participants actively contribute to the task execution while the monitors simply observe the activities of the task. Each P-Task produces some *Outputs* from *Inputs*.

3. Articulation Tasks (A-Tasks) - the tasks that are carried out to fix breakdowns in P-Task executions. A-Tasks result from some *Problem*, last for a certain *Duration*, have some people who are *Involved* in fixing the problem, or carrying out the A-Task, and have some P-Tasks that are *Affected* by the A-Task. A-Tasks use some *Resolution Strategy*, e.g., relax the input constraints on the affected P-Task, or change the representation language for a task's output.
4. Products - the results of P-Tasks or A-Tasks. In this iteration of the meta-process (see Figure 1), we have a very high level view of the products, but this will be refined for subsequent iterations.

3 Process History Analysis

The goal of the history analysis is to understand what happened in the process so that in the next iteration we can remove the deficiencies of the history. We use three primary forms of analysis: (1) Goal Analysis [2], (2) Task Analysis, and (3) Participant Analysis.

In the goal analysis, we evaluate the goals of agents working on a task with respect to the goals of the task in the process. The hypothesis is that if the goals of the agents are not aligned with the goals of the task then the task will not be successful. Also, we checked to see if the tasks had explicit goals that contributed to the process goals.

In the task analysis, we analyze task attributes and relationships. For example, whether or not a task had a single agent responsible for the task, or whether or not communication between concurrent and inter-dependent tasks was carried out successfully. In the participant analysis, we take opinions of participants in the process history and highlight them for discussion in the process feedback stage. Based on the discussion during the feedback, we can either adopt, modify, or discard the analysis. Participant analysis is more subjective than goal and task analysis.

Note that none of these analyses techniques are specific to reuse as such. However, since they are being used to discover and evolve a reuse-oriented process, the issues become relevant for this workshop.

4 Some Early Results

As discussed earlier, we started the first meta-process iteration with an explicit process for developing the first iteration of a kit. Based on our analysis of the history of the process, we discovered several problems with the history, including problems of goal alignment, task responsibility, and concurrent task synchronization. Based on this analysis, we were able to suggest a process for domain specific kit production and consumption, which was much simpler than the process that we had for Kit_0, and is shown in . Note that this is just the first iteration through our meta-process, and is, therefore, an early result. We expect this to be refined further as we go through more iterations of the meta-process.

One interesting aspect of this process is that, since the team did not have any experience in the domain for which the kit was being produced, it was necessary to prototype one application in the domain. Moreover, the application prototype was useful throughout the process. For example, the prototype could be used as an illustration for kit usage to help teach building new applications in the domain.

Another interesting aspect was the set of issues relating to the synchronization of the inter-dependent tasks of domain analysis and kit design. We had first assumed that these tasks could be done in parallel with the results of domain analysis feeding into the kit design. However, this model does not work, specially in the short cycle times such as ours, because these activities seem to highly inter-dependent and each of them requires substantial time. We are looking at alternative ways of synchronizing these tasks.

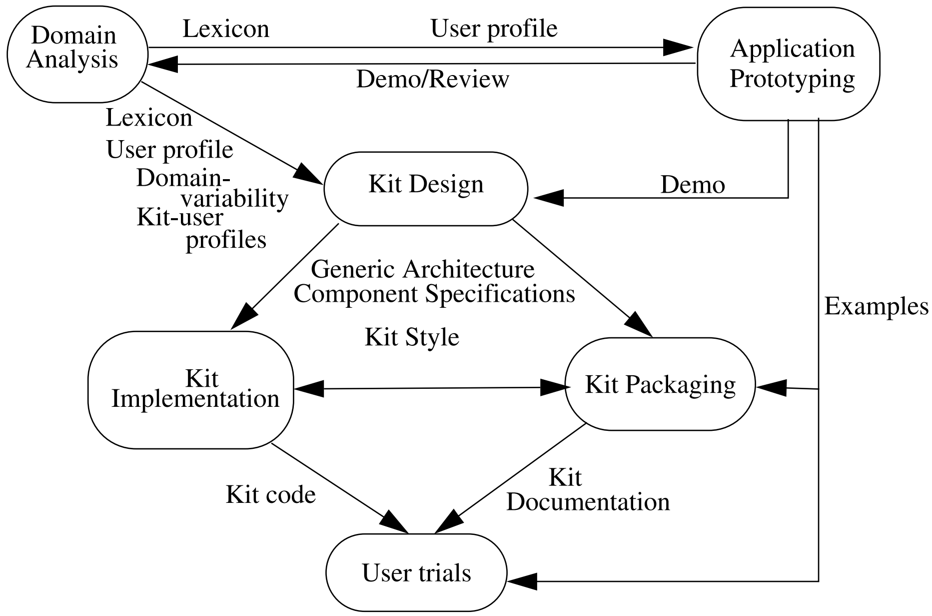

Figure 2: An Empirically derived Domain-specific kit production and consumption process

5 Summary

In this paper we have posited that a meta-process is necessary for discovering and evolving software reuse processes. We have presented one such meta-process along with its early experimental use. We anticipate using this meta-process for a few more iterations to determine an empirically grounded domain-specific kit-based software production and use process.

6 Acknowledgments

We thank the participants of the kit production team from HP Labs for their patience in letting us intrude and observe their process. We thank Lew Creary and Allan Shepherd for their extensive comments on earlier versions of this paper. We thank Dan Berry of the Software Engineering Institute for his feedback on this work. We thank Martin Griss and Kevin Wentzel for their support of this project.

References

[1] B. I. Blum, *Software Engineering: A Holistic View*, Oxford University Press, New York, NY, 1992.

[2] B. W. Boehm, *Software Engineering Economics*, Chapter 3, Prentice-Hall, Englewood Cliffs, NJ, 1981

[3] P. K. Garg, “Process Modeling of Software Reuse”, In *Proceedings of the 5th International Workshop on Institutionalizing Software Reuse*, Palo Alto, October 1992.

[4] W. Humphrey, *Managing the Software Process*, Addison-Wesley, Reading, MA, 1989.

[5] P. Mi and W. Scacchi, “A Knowledge-Based Environment for Modeling and Simulating Software Engineering Processes”, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 2, No. 3, Sept,1990.

[6] G. A. Rummler and A. P. Brache, *Improving Performance: How to Manage White Space on the Organization Chart*, Jossey-Bass Inc., San Francisco, CA 94104, 1990.