

# Making Software Process Simulation Modeling Agile and Pattern-based

Position Paper

Ninie Angkasaputra

*Fraunhofer Institute for Experimental Software  
Engineering (IESE)  
Sauerwiesen 6,  
D-67661 Kaiserslautern, Germany  
angkasa@iese.fraunhofer.de*

Dietmar Pfahl

*Fraunhofer Institute for Experimental Software  
Engineering (IESE)  
Sauerwiesen 6,  
D-67661 Kaiserslautern, Germany  
pfahl@iese.fraunhofer.de*

## 1. Introduction

Though being the origin or predecessor of all process simulation modeling (PSM) approaches used in software engineering today, until recently, there did not exist a detailed and to software engineering needs adjusted simulation modeling methodology around the System Dynamics (SD) simulation modeling approach. The fact that existing process guidance for SD model development was only on informal and coarse grain level stimulated the development of a framework for SDM development, IMMoS (Integrated Measurement, Modeling, and Simulation) [11][12].

The IMMoS methodology supports managers to cope with the dynamic complexity of software development by providing guidance on constructing simulation models and using them as a source for learning and improvement. This was achieved by enhancing existing guidance for SD modeling with an additional component that enforces goal-orientation and with a refined process model that details description of activities, entry/exit criteria, input/output products, and roles involved. To smoothly close the gap to methods already established in empirical software engineering, IMMoS describes how to combine SD modeling with goal-oriented measurement and descriptive process modeling. IMMoS methodology consists of four phases and seventeen main activities distributed among the phases to guide the users in constructing an SD model.

Unfortunately, software PSM is still considered a time-consuming and expensive task not yet widely accepted in software engineering. This is partly due to the specific nature of software development as compared to other engineering discipline where PSM is much more popular. Software development does not only combine methods, techniques and tools but it is a predominantly human activity. Software processes are sequences of (interrelated and partly iterative) actions that people take in order to achieve a result, i.e., a software product.

Requirements on software development are low costs, fast development, accepted failure rate, and flexibility. Changes on requirements, process, and personnel are frequent. In summary, software development processes can be characterized as human-centered, change-prone, very flexible, and thus rather instable. Consequently, it is much more difficult to develop valid software process simulation models in time and in budget than in other engineering disciplines.

## 2. Proposed Enhancement to PSM

We believe the problematic situation of software PSM can be altered by making use of emerging practices in the software development that seem to be promising with regards to increased effectiveness and efficiency. The key idea is to make software PSM more agile by taking benefits from *agile methods* and (*design*) *patterns*. Both share the same objectives, i.e., to reduce product delivery time and budget. Different agile methods aim at making software development processes more flexible and participative (incl. the end-user). Patterns are means to reuse knowledge, problem solutions, or strategies that have proven to be successful in previous projects.

In the context of software PSM, simulation models are the products to be delivered. Our intent is to investigate how agile methods and the concept of patterns can be adapted to enhance IMMoS and thus make software PSM faster, cheaper, and eventually more frequently used in software industry. Our goal is to demonstrate that with the enhanced IMMoS, which will be denoted as Agile/P-IMMoS, SD models in the application domain of software engineering can be developed with less effort and cost without sacrificing the quality.

### 2.1. Why Agile Methods?

A good development method is light in weight and focuses on what people actually do. A method needs to

be “agile”, i.e., incorporating the following values [5]:

- individuals and interactions are more important than processes and tools,
- working products are more important than comprehensive documentation,
- collaboration with customers (end users) is more important than contract negotiation,
- responding to change is more important than following a plan.

Agile methods, such as Agile Modeling [2] and Extreme Programming (XP) [3] transform development processes in such a way that they can be characterized as follows [1]:

- Incremental: develop in small releases to deliver running models and get immediate feedback to improve until becoming the final one.
- Cooperative: make use the communication between the modelers with the domain experts (i.e., the clients).
- Straightforward: the method itself is easy to learn and to modify.
- Adaptive: able to react to last moment changes.

## 2.2. Why (Design) Patterns?

In software engineering, patterns are an emergent discipline originated from the object-oriented community<sup>1</sup> in 1987. In general, patterns describe best practice software development knowledge. Each pattern adheres to a three-part rule that expresses a relation between a certain context, a problem, and a solution. The development of software systems requires significant experience or best practice know-how even when supported by methods. During the development, many problems have occurred and may occur again. The real challenge is how to solve them immediately. Therefore, developers rarely solve problems completely from scratch, but reuse existing solutions proven to be good. This experience is to be packaged as reuse elements so-called patterns. For example, Gamma et. al. [7], package such expert knowledge in the form of a systematically edited, schematized and structured catalog of patterns.

## 3. Results of a Literature Survey

A literature survey has been done to investigate to what extent agile methods and patterns have been transferred into software PSM. The result was that to date not much work has been addressed this topic.

Regarding transfer of agile methods, only one related piece of research was found [8]. It is not located in the application domain of software PSM but focuses on agile

development of software for simulation. In this work, the authors apply Yare Programming (YP), an agile programming methodology, in developing the software for urban simulation that lets different stakeholders evaluate the results. YP was chosen due to its open process: the development process and its status are visible (i.e., not a black box) to the customers, and can be inspected easily. Even more important is its flexibility: the process allows for quick response to different stakeholder values and concerns.

As for the application of patterns, up to now, only few publications have reported the application of patterns in different application domains. Examples are pattern-based modeling for simulation of physical buildings [9], simulation using reusable elements in the form of building blocks for simulation of airport terminals [6], and simulation using model template for simulation of software test phases[4]. However, descriptions of how the reuse process can be aligned to the PSM process or whether the creation and updating of reusable elements is possible during the development are still missing. Another piece of related work is [10] where the author presents a methodology for software project planning based on process patterns.

## 4. Research Agenda

Our research plan is outlined in Figure 1. We will take IMMoS as the baseline software PSM method and extended it with agile methods and (design) patterns into Agile/P-IMMoS. The IMMoS method supports the System Dynamics simulation modeling approach and was validated in an industrial context [11].

The research activities are outlined in the following steps:

- Investigate how pattern techniques can be used to package and catalogue reusable elements of software processes (②). This will help answer the following questions: Which pattern description is suitable for software process simulation patterns? What is the adequate level of abstraction of the patterns in order to reuse them in different PSM paradigms and languages (e.g., Vensim, Extend)? How should patterns be catalogued to provide useful overview and fast search? The packaging and cataloguing of patterns will be referred to as pattern registration.
- Investigate how best practices of agile methods that have proven to be successful can be adapted to make IMMoS more agile (③).
- Identify validated and published software process simulation models that are good candidates of inputs to the pattern catalogue (④).
- Explore the added business value expected from the proposed software PSM method (i.e., Agile/P-

---

<sup>1</sup> Coplien, <http://hillside.net/patterns/>

IMMoS) (⑤, ⑥, ⑦, and ⑧). Main benefits are expected in the areas of software development and organizational learning.

- Describe how Agile/P-IMMoS will be validated.

## 5. Summary

In this position paper we propose to investigate into the potentials of existing best practices in software engineering, namely agile methods and patterns, in order to provide a light-weight, flexible software PSM method that is expected to deliver software process model faster and cheaper without compromising quality.

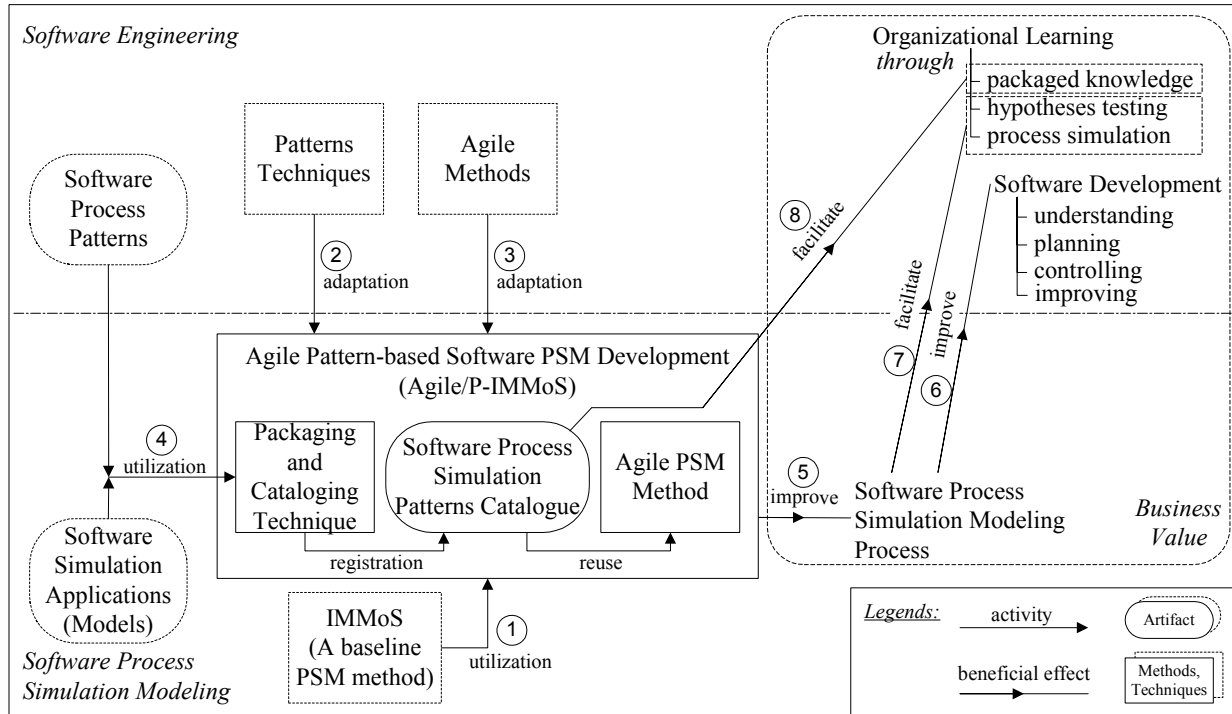


Figure 1: Research Plan on Agile/P-IMMoS

## REFERENCES

- [1] Abrahamsson P, Warsta J, Siponen MT, Ronkainen J, "New Directions on Agile Methods: A Comparative Analysis", in Proceedings of the 25th international conference on Software Engineering, Portland, Oregon, 2003.
- [2] Ambler SW, "Agile Modeling", John Wiley & Sons, New York, 2002.
- [3] Beck K, "Extreme Programming Explained. Embrace Change", Addison-Wesley, 1999.
- [4] Berling T, Andersson C, Höst M, Nyberg C, "Adaptation of a Simulation Model Template for Testing to an Industrial Project", in Proceedings of the 4th Process Simulation Modelling Workshop (ProSim-2003), Portland, USA, 3-4 May, 2003.
- [5] Cockburn A, "Agile Software Development", Addison Wesley, Boston, 2002.
- [6] Edwin Valentin, Alexander Verbraeck. Simulation Using Building Blocks. Proceedings of AI, Simulation and Planning in Highly Automated Systems, 2002.
- [7] Gamma E, Helm R, Johnson R, and Vlissides. J, "Design Patterns – Elements of Reusable Object-Oriented Software", Addison-Wesley, 1995.
- [8] Freeman-Benson B, Borning A, "YP and Urban Simulation: Applying an Agile Programming Methodology in a Politically Tempestuous Domain", in Proceedings of the 2003 Agile Development Conference, Salt Lake City, 2003.
- [9] Martin Schuetze, Jan Peter Riegel, Gerhard Zimmermann, "A Pattern-Based Application Generator for Building Simulation", in Proceedings of the 6th European Conference held jointly with the 5th ACM SIGSOFT international symposium on Foundations of Software Engineering, Zurich, Switzerland, p. 468-482, 1997.
- [10] Münch J, "Pattern-based Creation of Software Development Plans", Stuttgart: Fraunhofer IRB Verlag, PhD Theses in Experimental Software Engineering, Vol. 10, 2001.
- [11] Pfahl D, "An Integrated Approach to Simulation-Based Learning in Support of Strategic and Project Management in Software Organisations", Stuttgart: Fraunhofer IRB Verlag, PhD Theses in Experimental Software Engineering, Vol. 8, 2001.
- [12] Pfahl D, Ruhe G, "IMMoS: A Methodology for Integrated Measurement, Modelling, and Simulation", in International Journal of Software Process: Improvement and Practice, published online: 30-Oct-2003, pp189-210.