GoPoMoSA: A Goal-Oriented Process Modeling and Simulation Advisor

Xu Bai and LiGuo Huang
Dept. of Computer Science and Engineering
Southern Methodist University
Dallas, TX 75205, USA
{xbai,lghuang}@smu.edu

He Zhang
National ICT Australia
University of New South Wales
Australia
He.Zhang@nicta.com.au

Alexander Egyed
Institute for Software Engineering and Automation
Johannes Kepler University
Austria
alexander.egyed@jku.at

ABSTRACT

This paper presents *GoPoMoSA*, a Goal-oriented Process Modeling and Simulation Advisor that semi-automatically discovers suitable Software Process Modeling and Simulation (SPMS) techniques for (inexperienced) process modelers to achieve their process modeling goals. *GoPoMoSA* takes the goal-oriented modeling approach that captures the associations among Process Modeling Stakeholder goals and existing SPMS techniques via *Relevant Process Elements* modeled in the knowledge graphs. We evaluated the accuracy and feasibility of *GoPoMoSA* with data collected from 212 published SPMS literatures and a real-world process modeling and simulation case on requirements traceability. Our results show that *GoPoMoSA* (1) was able to find suitable SPMS techniques based on stakeholder goals with an average of 85.38% accuracy; (2) helped novice process modelers effectively and efficiently achieve their goals.

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management—*Software process models*

General Terms

Management, Verification

Keywords

Software Process, Software Process Modeling and Simulation, Goal-Oriented, Process Modeling Stakeholder

1. INTRODUCTION

Software Process Modeling and Simulation (SPMS) leverages planning, managing, controlling, improving software processes, and provides software process researchers and practitioners (e.g., professionals and managers) powerful tools and recognizable benefits. Since the pioneering work in 1980s, with the rapid expanding of research and development in this field, varieties of SPMS techniques and associated tools have been developed. Especially for novice researchers and practitioners, it is therefore a big challenge on how to select a suitable set of techniques and tools to help achieve the Process Modeling Stakeholder[1] goals for systems and software process management and improvement. The authors have recently conducted a systematic literature review on SPMS studies [1] and found that: (1) Stakeholders' process modeling and simulation goals are divergent [7]; (2) There is no one-size-fits-all approach to modeling and simulating software processes. Quite a few SPMS techniques/tools have been developed but they have been evaluated and/or deployed in rather exploratory ways only [1]; (3) Industrial experiences of how to use these techniques/tools to achieve the goals in process management and improvement are rarely reported in the literature [1]; (4) With the increasing complexity of process modeling and simulation scenarios, more complicated hybrid (combinations of different modeling schemes) SPMS techniques are needed to meet the modeling goals [10]. Thus, there is an emergent need to quickly bridge the gap between SPMS techniques/tools and the stakeholder modeling/simulation goals particularly for novice users in order to support the effective process management and improvement.

This paper presents *GoPoMoSA* (Goal-oriented Process Modeling and Simulation Advisor), a semi-automated approach that (1) generates and maintains the associated knowledge graphs of both stakeholder goals and SPMS techniques, generated from published literatures; (2) supports user interactions initiated by the users' inputs of her modeling goals through iterative user feedback; and (3) reasons about the most suitable set of candidate SPMS technique(s) based on knowledge graphs, users' modeling goals and proficiencies.

We have performed an unbiased statistical evaluation of *GoPoMoSA* using 212 collected SPMS literatures between 1981 and 2009 to cross-validate its accuracy. Our test results show that *GoPoMoSA* can suggest suitable candidate techniques/tools which include the actually deployed techniques/tools in over 85% of literatures in the test set. We have also validated the feasibility of *GoPoMoSA* on a real-world process simulation case study on identifying suitable traceability strategies for understanding the mapping between requirements and code. This case study was to understand three different traceability recovery processes in context of the open source system GanttProject. The user indicated that *GoPoMoSA* could quickly match the process

[1]The Process Modeling Stakeholder is the one who is involved in or affected by process modeling activities.

ICSSP'11, May 21–22, 2011, Waikiki, Honolulu, HI, USA

modeling/simulation goals with the suitable SPMS technique(s) and shorten the learning curve on SPMS techniques especially for the novice researcher and practitioner.

The reminder of this paper is outlined as follows. Section 2 discusses related work. Section 3 elaborates the *GoPoMoSA* approach. Section 4 presents the results of our initial evaluation and proof-of concept validation. Section 5 concludes and envisages the future work.

2. RELATED WORK

Software Process Modeling and Simulation (SPMS)

Varieties of SPMS techniques have been proposed during the last two decades. These techniques can be classified into *Discrete*, *Continuous*, *Hybrid* techniques based on their modeling perspectives [2]. Among these, Discrete-Event Simulation (DES) [9], Little-JIL [4], System Dynamics (SD) [6] are the most frequently used techniques [1].

Though the benefits of leveraging SPMS techniques in process management and improvement have been recognized, our recent systematic literature review results on SPMS research [1] still reveal that few experiences have been reported for the real-world applications of these techniques especially in industry. This may primarily be due to the fact that both the expertise and experience on software processes and SPMS are necessary to select the appropriate techniques to meet the modeling goals. In particular, inexperienced process modelers lack the experience in determining what techniques to choose and when, where and how to use them in order to achieve their modeling goals. Our approach aims to semi-automate the process of SPMS technique selection for achieving users' modeling goals based on the knowledge graphs generated from historical SPMS research and application literatures.

Process Modeling Stakeholder Classes and Goals

Process Modeling Stakeholder classes were initially proposed and discussed in the Workshop of Modeling Systems and Software Engineering Processes (MSSP) [7], based on the behavior analysis of people involved in software process modeling activities. A set of stakeholder goals of process modeling and simulation were also identified and further discussed in [2]. However, it is problematic that these top-level stakeholder goals are too general to be tangible and addressable in real-world modeling and simulation scenarios. Our approach instantiates the stakeholder goals by modeling them into the knowledge graphs. And this paper proposes a semi-automated approach to help novice process modelers in selecting appropriate SPMS techniques to achieve their modeling goals.

3. APPROACH

3.1 Overview

Figure 1 depicts an overview of *GoPoMoSA* which is composed of two components, the Goal/Technique *Modeler* and the *Reasoner*. The *Modeler* generates and maintains the independent knowledge graphs [3] of stakeholder goals and SPMS techniques based on case studies in SPMS relevant literatures [1] (a calibration that become more accurate the more case studies are known.). The *Reasoner* takes the user's modeling goals, proficiencies and preference as inputs and reasons about the best suitable set of SPMS techniques meeting the user's goals and proficiencies based on the graphs generated by the *Modeler*.

The *Modeler* thus requires knowledge on existing SPMS literatures as inputs. The stakeholder goals, deployed techniques, and their associations via *Relevant Process Elements* are modeled as knowledge graphs [3]. The *Modeler* is expected to be used by experts only - a calibration step that has to be done occasionally only and whose output (the knowledge graphs) are required by the *Reasoner*. The (novice) users, who want to match their goals with suitable SPMS techniques, are expected to only use the *Reasoner*. The *Reasoner* interactively queries the user on goals, modeling proficiencies and preferences, and tries to find candidate techniques that may achieve these goals.

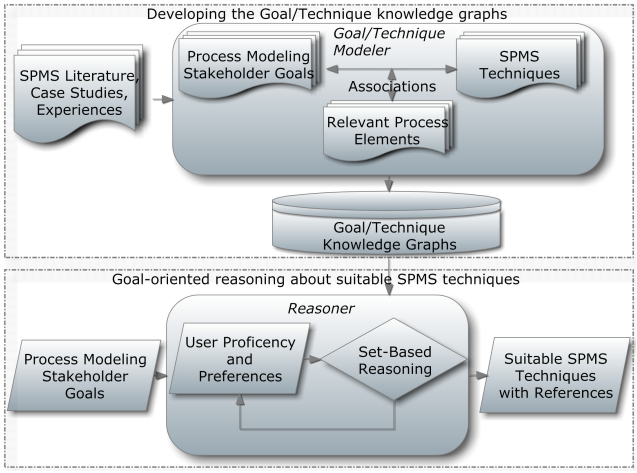

Figure 1: Overview: *GoPoMoSA* framework

3.2 Modeler: Modeling Goals

Stakeholder goals are achieved through software process modeling and simulation. We can model the stakeholder goal as a 4-tuple with the following four attributes.

Goal Identifier (GID). *GID* uniquely identifies a stakeholder goal. For instance, a brief description of a goal extracted from the literature (e.g.,"to achieve higher CMM level") can be the *GID*.

Scope. It indicates at what level of detail the software process modeling and simulation goal gets applied. This can be measured by the scope or granularity of the software process to be modeled. For instance, there are different ways on how to improve the process in an organization (e.g., project level or activity/task level improvements), which require different types of SPMS techniques and process parameters at different levels of detail.

Type. We learned from systematic literature review on SPMS [1] that, stakeholder goals can be categorized into three types: *Understanding Processes*, *Developing SPMS Techniques*, *Managing and Improving Processes*. Goals for understanding processes are mainly concerned with capturing quantitative or qualitative relationships among process elements, e.g., discovering how software quality improvement will affect the schedule and cost. Goals for developing SPMS techniques are targeted at creating and evaluating the techniques themselves, such as modeling the dependency among process elements, or evaluating the effectiveness and scalability of a technique. Goals for managing and improving processes focus on monitoring, controlling, configuring and changing the processes, e.g., tailoring the process to adapt to the emergent business opportunities or change risk priorities during the process execution.

Relevant Process Elements. Each goal has to associated with a set of process elements in order to become tangible and meaningful. For example, a stakeholder goal described merely as "improving the process" is not meaningful or useful until the *Relevant Process Elements* are clearly identified. "improving the process" can be elaborated as "reducing the cost while maintaining the software quality" or "optimizing the workflow structure". Goals described with process elements are more useful to stakeholders because the measurement of these process elements can be the indicator of whether and how well the goals are fulfilled.

GoPoMoSA represents every goal identified into a knowledge graph composed of its *GID*, *Scope*, *Type* and *Relevant Process Elements*. In this section and section 3.3, we use Raffo's work "*Software process simulation to achieve higher CMM levels*" [8] as an example to illustrate the construction and representation of the knowledge graphs in *GoPoMoSA*. From [8], we identified that the stakeholder goal was to "achieve higher CMM level" (*GID*) in an "organization" (*Scope*) by "managing the software process" (*Type*), concerning with "man power, cost, schedule and quality" (*Relevant Process Elements*). Figure 2 shows the generated goal knowledge graph.

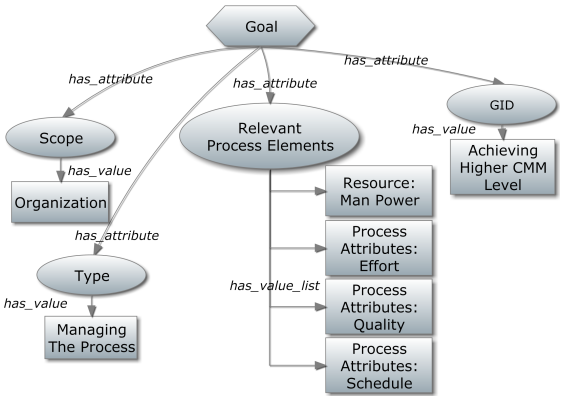

Figure 2: Example goal knowledge graph

3.3 Modeler: Modeling SPMS Techniques

Each SPMS study may report the application of one or more SPMS techniques in achieving stakeholder goals. The SPMS techniques provide means of formalizing the process elements in order to facilitate process simulation, evaluation, verification and validation. We can model each SPMS technique as a 5-tuple with the following five attributes.

Technique Identifier (TID): *TID* uniquely identifies a SPMS technique. For example *TID* can be the name of technique in the literature.

Modeling Scheme: This captures how the processes are perceived by the modeling technique, e.g., a collection of tasks and their interactions (Discrete Modeling), a dynamic feedback system (Continuous Modeling), a combination of these two schemes (Hybrid Modeling). Different modeling schemes may have their distinct properties and different application scenarios.

Simulation Support: This indicates whether the technique supports the execution of process model. For example, simulation support is not needed if a SPMS technique is used for the static modeling of a process only (e.g., to develop a task break down structure which divides a higher-level task into lower-level tasks).

Modeling Capabilities: Each technique can model a set of process elements. The *Modeling Capabilities* are thus defined by these process elements. For example, the COCOMO II model is used to model the *cost* and *schedule* of a software project so that *cost* and *schedule* are included in COCOMO II's *Modeling Capabilities*.

Usability Properties: This attribute defines how a technique is designed for the sake of users' modeling proficiency and preference. For instance, does tool support exist for the technique (Tool Support)? Does the technique support visualization of process elements or execution (Visualization)? Does the technique require specific programming skills (Coding)? The set of *Usability Properties* is evolving with the emerging needs of process modeling stakeholders.

GoPoMoSA models every technique identified from literatures into a knowledge graph composed of its *TID*, *Modeling Scheme*, *Simulation Support*, *Modeling Capabilities* and *Usability Properties*. For example, the case in [8] adopted the technique named as "Northrop Grumman Model" (*TID*), taking the "discrete modeling" (*Modeling Scheme*) approach to modeling and simulating the "activities, man power, quality, cost and schedule" (*Modeling Capabilities*). This technique "supports simulation" (*Simulation Support*) with a dedicated "tool" (*Usability Properties*), "visualizes" the running of process, and requires "process parameters" as user inputs. The generated graph is shown in Figure 3.

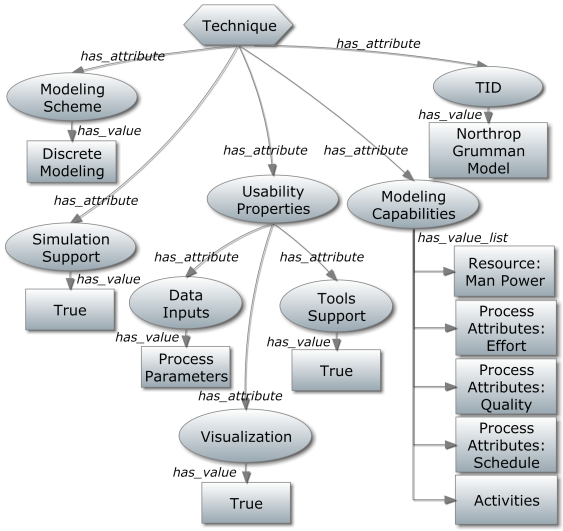

Figure 3: Example technique knowledge graph

A technique can serve different goals, which implies that case studies may report the deployment of the same technique under different modeling and simulation scenarios, deploying different set of its *Modeling Capabilities*. *GoPoMoSA* simply adds these different uses in the knowledge graph. In this case, multiple knowledge graphs for the same technique are generated and then combined by a JOIN operation on its attributes. Assuming a literature set L ($|L| = n$) used the same technique t, a combined knowledge graph of t can be obtained by:

$$t.properties = \bigcup_{i=1}^{n} L_i.t.properties$$

Hybrid SPMS techniques (e.g., a combination of System Dynamics and Discrete-Event Simulation) are modeled based on multiple independent technique knowledge graphs. To model hybrid techniques, each participating technique contributes to a subset of *Relevant Process Elements* of the

modeling goal and a subset of its *Modeling Capabilities* is selected. The modeling of the usage scenarios hybrid techniques, which tells the user how to integrate and where to deploy these hybrid techniques, is currently not described in this paper but in our future work. Under different usage scenarios, the same combination of techniques may be integrated in different ways (e.g., vertically in the same development phase or horizontally across different phases). With the current *Modeler*, *GoPoMoSA* suggests the combination of candidate techniques to the user and let her decide how to integrate them in her application scenarios.

3.4 Modeler: Associating Goals with Techniques

Assuming a SPMS literature documents a successful application of SPMS techniques to achieve certain modeling goals, the *Relevant Process Elements* (*RPE*) set of stakeholder goals has to be a subset of *Modeling Capabilities* (*MC*) set of techniques used in the same literature, to ensure the achievements of stakeholder goals. That is, the *Relevant Process Elements* in Figure 2 should be covered by the *Modeling Capabilities* in Figure 3. In other words, for literature l, its goals set G ($|G| = n$) and deployed techniques set T ($|T| = m$) should satisfy

$\bigcup_{i=1}^{n} l.G_i.RPE \subseteq \bigcup_{j=1}^{m} l.T_j.MC$

GoPoMoSA automatically generates the associations between the goal graph and techniques graph. An association exists if the above condition is satisfied.

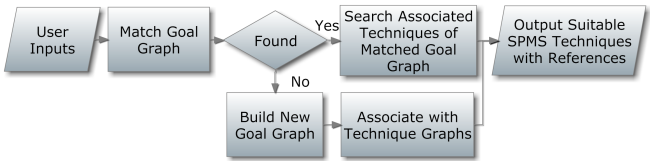

Figure 4: The workflow of GoPoMoSA Reasoner

3.5 Reasoner: Set-Based Reasoning Engine

Figure 4 shows the workflow of the *Reasoner* (which is the only part used by the end user). The *Reasoner* of *GoPoMoSA* takes the user input of her modeling/simulation goal and tries to search within the existing goal knowledge graph set G built by the *Modeler*. If the user's goal can be covered by an existing graph $g \in G$, all the associated techniques with g will then be provided to the user as a potentially suitable set of candidate techniques. If there is no perfect match, the user inputs will be used to generate a new goal knowledge graph g', and the *Reasoner* will report on individual technique knowledge graph t (which is in existing technique graph sets T) which can satisfy the user preferences, and the cartesian product of g' and t indicates all the *Relevant Process Elements* of g' have valid association with t. In case that no individual technique can serve the user's goal, combinations of individual techniques (based on existing approaches) are tested and provided to the user.

3.6 Reasoner: Integrating User Modeling Proficiency and Preference

During its interaction with the user, *GoPoMoSA* will ask the user about modeling proficiency and preference. Example questions can be "*what are your preferred programming languages?*" and the user's answers will become another set of input for the *Reasoner* to be checked against the *Usability Properties* of the candidate techniques output from the previous reasoning step. Assuming the set of a user's *Proficiency and Preference* is PP, and the set of a candidate technique's *Usability Properties* is UP, if we have $PP \subseteq T.UP$, then the technique can be finally recommended to the user without much further learning effort. Otherwise, the closest candidate techniques with a learning curve will be recommended. The literatures related to suggested techniques are also provided to user for reference. Note that the scalability will not be an issue because the questions are only relevant to the *Usability Properties* of the candidate techniques output from the previous reasoning step.

4. EVALUATION

In this section, we discuss the evaluation on the accuracy of *GoPoMoSA* on 212 SPMS literatures. In addition, a proof-of-concept case study based on a real-world process simulation scenario is described.

4.1 Accuracy of GoPoMoSA

This section answers the question how accurate is the set of techniques that *GoPoMoSA* selects and suggests to the user in terms of goal fulfillment. In general, each SPMS literature discussed an application scenario of at least one SPMS techniques to achieve certain modeling/simulation goals. As mentioned in Section 3.4, we assume the literatures always reported successful goal fulfillment (i.e., incorrect calibration results in incorrect feedback) by certain SPMS techniques based on our review results.

We used k-fold cross-validation to evaluate the prediction accuracy of *GoPoMoSA* based on case studies collected from 212 SPMS literatures. Assuming $k = 3$, we randomly selected 2/3 of the case studies from literatures as the training data set to generate Goal and Technique Knowledge Graphs and their associations while the remaining 1/3 as the testing set. That is, for testing, the modeling goals defined were used as the inputs to *GoPoMoSA*, which output the appropriate technique set. Then we compared the techniques found by *GoPoMoSA* with the techniques reported to be actually deployed in the literature. If the latter was a subset of the former, then we claimed that the prediction was accurate (true positive). Otherwise, the prediction was inaccurate. Assuming the cardinality of literatures in test set is $|TEST|$, and the number of literatures whose actually applied techniques were found in the *GoPoMoSA* output (true positive) is N_{tp}, we have $Accuracy = \frac{N_{tp}}{|TEST|}$.

Note that *GoPoMoSA* might suggest more technique candidates (referred as unreported techniques) other than what were actually reported in the literature. However, we found that the unreported techniques still could have been alternative techniques to also serve these goals. Even if some of the unreported techniques couldn't serve the goal well, our tool largely reduced the user's effort by only investigating the suggested techniques. Totally 3 iterations of such cross-validations were performed with randomly selected different training and testing data sets.

Figure 5 shows the measured accuracy in three independent tests and the average. In average, *GoPoMoSA* achieves 85.38% accuracy in finding the appropriate set of techniques to achieve the modeling/simulation goals proposed in the literatures of testing set. The actual case is even better because (1) approximate 20% of the literatures documented SPMS techniques occurred only once in the collected literatures. If such literatures happened to fall into the testing

set, their goals and techniques would not be modeled based on the training data set so as not to be found by the *Reasoner* later; (2) the accuracy did not count in the outputs which were not identical to but might still be suitable alternatives of the techniques described in the literatures, and (3) *GoPoMoSA* provided candidate techniques for all test set when user goals were modeled correctly.

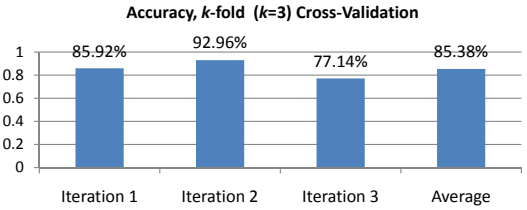

Figure 5: The accuracy of GoPoMoSA

4.2 Case Study: Modeling Processes of Requirement Trace Recovery

To demonstrate the feasibility of our approach, we here present the result of our initial proof-of-concept case study using *GoPoMoSA* to help the novice user select the suitable SPMS technique in modeling three processes of requirement trace recovery. Egyed et al. have been investigating on the cost-effectiveness of recovering the traces between requirements and codes by novices (students) and experts (perfect developers) in [5]. The recovering process factors are under attention. Three trace recovering processes to be investigated include: (1) a batch process, where no traces are recovered occasionally but not updated in between; (2) an incremental process, where traces are captured and updated with every change (every commit of code into the version control system); and (3) a batch-increment process, where traces are initially not captured (perhaps until some major release) and then captured and maintained incrementally thereafter (a hybrid of 1 and 2). Their modeling goals are summarized as follows

- Primary *Type* of goals is understanding the process;
- The *Scope* of process covers 2-3 sequential activities;
- The *Relevant Process Elements* include the *effort* of trace recovering in seconds, *quality* of traces in terms of percentage of suspicious traces (unchecked traces after commits), and *time* points these process attributes being checked;
- Their *User Proficiency and Preference* for simulation techniques include no visualization need, no learning of specific tools, and quantitative data relationships. These special aspects of *GoPoMoSA* are thus not relevant here.

Based on these inputs, *GoPoMoSA* outputs two candidates: a System Dynamics (SD) based simulation or a Discrete modeling approach with capabilities of modeling quantitative process attributes. Taking the user's preference into account, SD and its related references (e.g., [6] etc.) were finally recommended. This finding was accurate as the simulation was actually manually decided using SD before this case study, and this case study independently confirmed this decision. This case study is thus another independent confirmation that demonstrates that *GoPoMoSA* is able to recommend techniques for satisfying process goals, i.e., for understanding the effects of three different trace recovering processes on the cost-effectiveness for novices and experts. The user's feedback on the successful application of *GoPoMoSA* not only demonstrates the feasibility of the approach, but also shows that the tool can help users with little process simulation experience quickly.

5. CONCLUSIONS AND FUTURE WORK

Selecting appropriate process modeling and simulation techniques is critical for effective process management and improvement. With varieties of SPMS techniques and modeling goals developed and documented in a large number of literatures, it imposes a big challenge especially for users who have little modeling and simulation background to discover the suitable technique set to achieve their goals. We have used Raffo's "*Software process simulation to achieve higher CMM levels*" as an example to illustrate how *GoPoMoSA* models goals, techniques and their association and how it reasons about the techniques based on users' goals, modeling proficiencies and preference. Our evaluation results indicate that *GoPoMoSA* can find appropriate SPMS techniques to achieve stakeholder goals with an average of 85.38% accuracy and an initial proof-of-concept case study shows its feasibility to help researchers and practitioners efficiently select and understand SPMS techniques for a real-world process modeling and simulation example.

Future work will investigate how to integrate different usage scenarios of SPMS techniques into the *Modeler* so that more detailed deployment advice can be provided.

6. REFERENCES

[1] X. Bai, L. Huang, and H. Zhang. A systematic mapping study of software process modeling and simulation. *Technical Report, SMU, TR03CSE11*, 2011.

[2] X. Bai, L. Huang, H. Zhang, and S. Koolmanojwong. Hybrid modeling and simulation for trustworthy software process management: a stakeholder-oriented approach. *Journal of Software Maintenance and Evolution: Research and Practice*, 2010.

[3] R. Bakker. Knowledge Graphs: representation and structuring of scientific knowledge. *PhD Disseratation, University Twente*, 1987.

[4] A. G. Cass, B. S. Lerner, J. Stanley M. Sutton, E. K. McCall, A. Wise, and L. J. Osterweil. Little-jil/juliette: a process definition language and interpreter. In *ICSE '00*, pages 754–757. ACM, 2000.

[5] A. Egyed, F. Graf, and P. Grunbacher. Effort and quality of recovering requirements-to-code traces: Two exploratory experiments. *Requirements Engineering, IEEE International Conference on*, 0:221–230, 2010.

[6] R. Madachy and B. Boehm. *Software Process Modeling With System Dynamics*. John Wiley & Sons, 2004.

[7] R. Madachy, S. Koolmanojwong, L. Osterweil, L. Huang, and M. Phongpaibul. In *Presentations on the Workshop of Modeling Systems and Software Engineering Processes*, 2008.

[8] D. Raffo. Software process simulation to achieve higher CMM levels. *Journal of Systems and Software*, 46(2-3):163–172, Apr. 1999.

[9] D. M. Raffo. Capturing software process and product characteristics in process models using task element decomposition. *CASCON94*, 1994.

[10] H. Zhang, B. Kitchenham, and D. Pfahl. Software process simulation modeling: An extended systematic review. In *International Conference on Software Process (ICSP'10)*, pages 309–320. Springer, 2010.