LiProMo—Literate Process Modeling

Jakob Pinggera, Thomas Porcham, Stefan Zugal, and Barbara Weber

University of Innsbruck, Austria
{jakob.pinggera, thomas.porcham, stefan.zugal, barbara.weber}@uibk.ac.at

Abstract. Recently, research on quality issues of business process models has begun to investigate the process of process modeling, i.e., the process of creating process models. In particular, it has been recognized that during this process, well-functioning communication between domain experts and system analysts is essential for understandable process models. This paper proposes the LiProMo approach to foster communication among system analysts and domain experts by flexibly interlinking textual descriptions and formal process models. The feasibility of LiProMo is shown by a prototypical implementation as well as a visionary scenario that illustrates the usage and benefits of LiProMo. The adoption of Cheetah Experimental Platform as basis for the prototye will support empirical evaluation of LiProMo, as planned for future work.

Key words: business process modeling, process of process modeling, literate process modeling

1 Introduction

Business process models play an important role for managing business processes [1]. Business process models, or process models for short, are for example used to support the analysis and design of process-aware information systems, service-oriented architectures, and web services. In addition, they help to obtain a common understanding of core processes of a business [2] and enable us to identify problems and to discover opportunities for improvement [3].

The process of creating process models, denoted as *process of process modeling* [4], can be characterized as an iterative and collaborative process which typically involves several stakeholders like domain experts and system analysts [5]. It has been recognized that this process of process modeling influences the quality of the resulting process model [4–6]. During this process, information about the domain to be modeled is transferred from the domain experts, who have the knowledge about the domain, but usually lack formal modeling skills, to the system analysts, who select appropriate modeling constructs and formalize this information. This communication, however, is often hampered by the fact that different vocabularies are used, leading to misunderstandings and faulty process models. Similarly, without information from a domain expert, a system analyst may find it difficult to infer the business rules behind modeling constructs [7], bearing a potential source of error. Given the fact that a considerable percentage of lifecycle costs are related to maintenance [8], there is strong demand for better understandable process models reducing the time needed for conducting changes and decreasing the risk of introducing errors.

In this paper, we introduce a technique called Literate Process Modeling (LiProMo), which aims to improve communication during the process of process modeling as well as the maintainability of resulting process models. To this end, LiProMo interweaves the textual descriptions of business processes and their formal business process models. In this way, arbitrary formal process models can be annotated with text fragments, presumably providing a discussion basis for domain experts and system analysts. To put the concepts of LiProMo into practice, we developed a prototypical implementation of an editor for LiProMo. Future validation of the LiProMo approach will be based on this editor.

The paper is structured as follows. Section 2 introduces LiProMo. Section 3 presents the LiProMo prototype. Section 4 describes how we envision the usage of LiProMo. The paper is concluded with related work in Section 5 and a summary and future work in Section 6.

2 Literate Process Modeling

LiProMo is based on the idea of combining graphical process models and informal textual descriptions (cf. Fig. 1A). This combination is motivated by dual channel theory [9], which states that pictorial and textual representations are processed differently by the human mind. Pictorial information is processed in parallel by the visual system whereas textual representations are processed serially by the auditory system [10]. This fact is exploited by dual coding theory, suggesting that conveying information is more efficient when images and text are combined [11]. [12] goes even further by claiming that *"textual encoding is most effective when used in a supportive role: to supplement rather than to substitute for graphics"*. This is underpinned by the findings presented in [13] stating that the understanding of a business problem is significantly increased when reading a BPMN model and the corresponding written use case description.

State of the art process modeling environments like Signavio[1] or IBM Websphere[2] provide means for adding textual descriptions to activities and including comments either directly in the process model or on separate pages. Similarly, wiki-based process modeling systems allow users to link parts of the process model to wiki pages (for an overview see [14]). The major disadvantage of this approach is that by incorporating comments directly into the process model *"visual clutter"* is added, which might *"confound their interpretation by making it more likely they will be interpreted as constructs"* [12]. On the contrary, attaching comments to activities or adding them on separate pages makes them more difficult to access and therefore prone to split-attention effect [15], hampering the understanding of process models [16].

LiProMo follows the idea of Literate Programming [17], which was designed to foster program comprehension. It was later introduced in UML modeling, combining textual descriptions and UML models to create more comprehensive documentations [7]. We adopt this idea for business process modeling to support domain experts and system analysts when collaboratively creating process

[1] www.signavio.com
[2] www.ibm.com/software/websphere

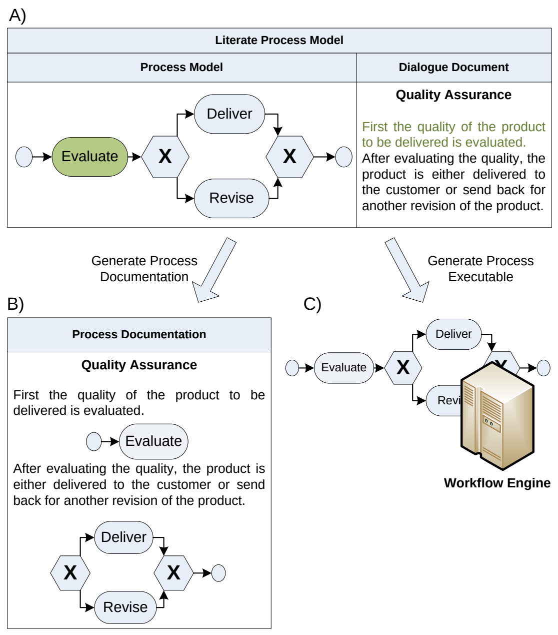
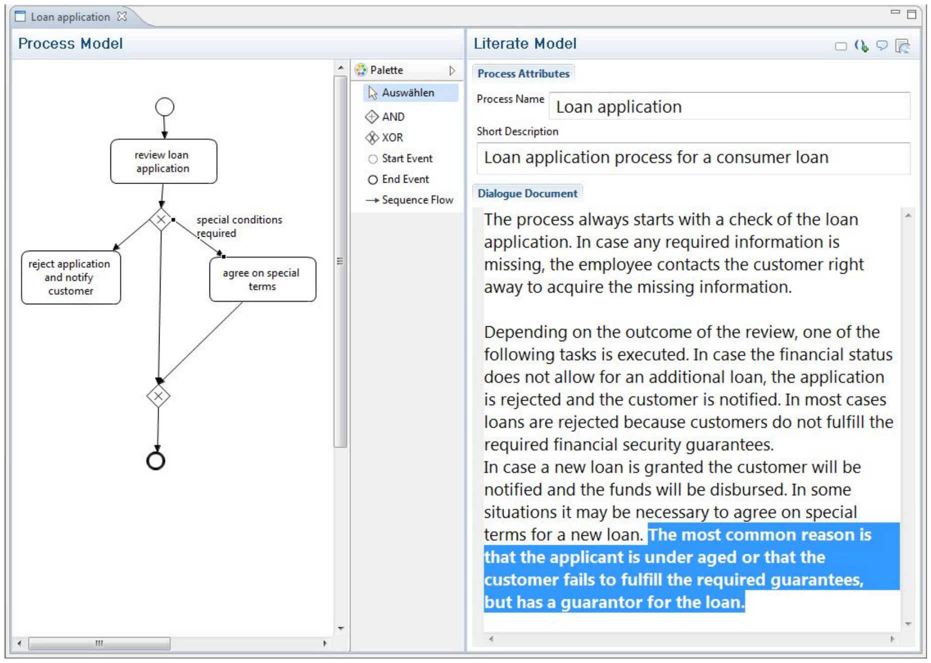

A)
Literate Process Model
Process Model
Dialogue Document
Evaluate
Deliver
Revise
X
Quality Assurance
First the quality of the product to be delivered is evaluated.
After evaluating the quality, the product is either delivered to the customer or send back for another revision of the product.
Generate Process Documentation
Generate Process Executable
B)
Process Documentation
Quality Assurance
First the quality of the product to be delivered is evaluated.
After evaluating the quality, the product is either delivered to the customer or send back for another revision of the product.
C)
Workflow Engine

Fig. 1. Literate Process Modeling

models and to improve process model maintainability. Fig. 1 sketches the basic idea of LiProMo. The process modeling editor depicted in Fig. 1A consists of two separate areas holding the process model and the corresponding textual description. To avoid split attention effect, the LiProMo approach juxtaposes the complete textual description and the graphical process model and links model elements with the corresponding task descriptions. The explicit links between process model and textual description can also be exploited for generating process documentations by interweaving the process model and the corresponding parts of the informal specification in a single document (cf. Fig. 1B). When revisiting the process model for conducting changes, the process documentation provides valuable knowledge about modeling choices made in the past, since not only the information contained in elements of the formal process model, but also additional information is readily available and linked to the corresponding model elements, e.g., explanatory examples. The additional information provided by the documentation reduces the risk of introducing errors. Efforts needed for creating additional documentation are expected to be regained as less time is spent on fixing errors [17]. LiProMo models can also serve as executable specifications, i.e., the process model can be fed into a workflow engine providing execution support for process models (cf. Fig. 1C).

3 Literate Process Modeling Editor

We developed a prototypical editor for LiProMo to assess its feasibility. As illustrated in Fig. 2, in a LiProMo model, the textual description of the whole process model and the actual process model are juxtaposed. During the creation of the process model, system analysts and domain experts work together in creating a process description tightly interconnected with the graphical process model (for details on how we envision such a scenario, see Section 4). The LiProMo editor allows for explicitly linking model elements, i.e., single activities or edges, but also whole process fragments that should be documented, to arbitrary passages in the textual description. No restrictions are imposed on either the size and shape of the process fragments or the linked text fragments. The editor automatically highlights the associated textual descriptions for selected modeling elements and vice versa, e.g., in Fig. 2 the edge labeled “special conditions required” is selected, resulting in the highlighted passage of the textual description on the right. The text gives examples for special conditions that would result in taking this execution path.



Loan application

Loan application process for a consumer loan

The process always starts with a check of the loan application. In case any required information is missing, the employee contacts the customer right away to acquire the missing information.

Depending on the outcome of the review, one of the following tasks is executed. In case the financial status does not allow for an additional loan, the application is rejected and the customer is notified. In most cases loans are rejected because customers do not fulfill the required financial security guarantees.
In case a new loan is granted the customer will be notified and the funds will be disbursed. In some situations it may be necessary to agree on special terms for a new loan. **The most common reason is that the applicant is under aged or that the customer fails to fulfill the required guarantees, but has a guarantor for the loan.**

Fig. 2. Literate Process Modeling Editor

Another benefit of making associations between modeling elements and textual descriptions explicit is the possibility of generating structured documentation. For every association, the textual description and the process fragment are displayed next to each other, resulting in a step-by-step explanation of the process model. The textual information documents the modeling elements and gives additional information that cannot be derived from activity names or edge descriptions.

In order to be able to evaluate the potential benefits of the LiProMo approach, we built the editor on top of Cheetah Experimental Platform [18]. By logging all interactions, i.e., changes to textual descriptions and graphical process models with the modeling environment to a central database, we are able perform a step-by-step replay of the process underlying the creation of the LiProMo model at any point in time, enabling us to perform analysis similar to [4].

4 Our Vision: Literate Process Modeling in Use

This section describes how we envision the interactions among system analysts, domain experts and the LiProMo editor. For this purpose, we provide a short example describing the collaborative creation of a LiProMo model. In particular, assume that a domain expert and a business analyst document the process of consumer loan applications in a banking institution.

Initially, the domain expert indicates that *"the process always starts with a check of the loan application. In case any required information is missing, the employee contacts the customer right away to acquire the missing information"*. So the system analyst enters the textual description and starts to create the graphical process model. To this end, the business analyst creates the first activity *"review loan application"* by selecting the relevant piece of textual information and choosing the activity name. Based on this information, the LiProMo editor creates the activity and links it to the relevant piece of text. This way, the activity name can be kept short, still providing additional information for future users, e.g., *"in case any required information is missing, the employee contacts the customer right away to acquire the missing information"*. The domain experts sees the changes in the process models and the highlighted textual description and explains that *"depending on the outcome of the review, one of the following tasks is executed. In case the financial status does not allow for an additional loan, the application is rejected and the customer is notified"*. The system analyst updates the textual description and decides to model the described scenario using the exclusive choice pattern. Hence, the business analyst creates an XOR split and an activity for rejecting the application and informing the customer, which is linked to the respective passage in the textual description. The domain expert adds *"in most cases we reject loans because customers do not fulfill the required financial security guarantees"*. The system analysts adds an additional comment to the reject activity and the edge connecting the XOR split and the activity including the examples given by the domain expert. The domain expert continues his explanations and states that *"in case a new loan is granted the customer will be notified and the funds will be disbursed. In some situations it may be necessary to agree on special terms for a new loan"*. As before, the business analyst first updates the textual information and adds an activity for agreeing on special conditions. The business analysts asks for examples of these special conditions. The domain experts answers *"there might be several reasons, but the most common are that the applicant is underage or that the customer fails to fulfill the required guarantees but has a guarantor for the loan"*. The business analysts links given examples to the edge between the XOR split and the special terms activity, only adding the phrase *"special conditions required"* to the edge instead of the lengthy examples given by the domain expert. Then the business analyst completes the process model by creating an XOR join and the end event (cf. Fig. 2). The business analyst and the domain expert go through the process model step by step. Since the domain expert is not familiar with BPMN, the system analyst selects the modeling elements they are currently talking about. The LiProMo editor highlights the corresponding textual descriptions, helping the domain expert in understanding the process model and enabling him to identify potential errors.

5 Related Work

We relate our work to four streams of research: research on understandability and maintainability of process models, the process of process modeling, the automatic generation of process models from natural language and research targeting communication between domain experts and system analysts.

Understandability and Maintainability of Process Models. The impact on model understanding and model maintenance has already been examined from various angles. For instance, [19] looks into the effect of modeling expertise, [20] discusses the influence of domain information, [16] investigates hierarchy, whereas [21] describes the impact of activity names. Similarly, [12, 22] discuss the relation between cognitive aspects and the understanding of process models. Like LiProMo, all these works deal with understandability and maintainability of process models, however, LiProMo rather focuses on the *process of process modeling* than on the outcome of process modeling, i.e., the resulting process model.

The Process of Process Modeling. The LiProMo approach focuses on improving the process of process modeling. Similarly, [23, 24] discuss the interaction of system analysts and domain experts. However, these works focus rather on the negotiation than on the creation of the process model, as done in LiProMo. The process of process modeling was investigated in [6, 18]. In contrast to LiProMo, they embrace a descriptive point of view on the process of process modeling, rather than trying to improve it.

Process Model Generation from Natural Language. There has been considerable research in the area of automatically deriving process models from natural language. [25] proposes a technique to automatically create BPMN models from natural language. [26] describes the creation of BPMN models based on group stories. Even though the automated generation of process models seems promising it is not clear—as argued in [27]—in how far these process models are well understandable. Moreover, several approaches impose restrictions on textual descriptions to provide sufficient structure to be able to derive a process model. Hence, we do not aim to automatically create process models from natural language, but rather support modelers in creating well documented process models in an iterative process involving domain experts and system analysts.

Improving Communication between Domain Expert and System Analyst. As motivated in this work, communication between domain experts and system analysts is often impaired by a different set of skills and vocabulary. For declarative process models, this problem has been tackled by the Test Driven Modeling (TDM) methodology [28, 29]. TDM combines test cases and process model to provide a common vocabulary for domain experts and system analysts. Besides improving communication such a combination improves the maintainability of declarative process models, as shown in [30]. In contrast, LiProMo focuses on supporting the creation and maintainability of imperative process models. In the upcoming evaluation we will strive for similar effects when utilizing LiProMo.

6 Summary and Outlook

In this paper we presented LiProMo—a technique that tightly interweaves graphical process models with their textual description. Presumably, the advantage of such an integration is threefold. First, according to dual coding theory, it allows for more efficient processing of information, hence directly supporting system analysts in creating the process models. Second, the tight integration of the textual description provides a common vocabulary, hence improving communication between domain experts and system analysts. Third, during model evolution, the interweaved availability of visual process model and textual description presumably lowers the chance of misinterpretation, hence improving process model maintenance.

So far, however, these conjectures are based on theoretical considerations only. To corroborate them, we are currently planning an empirical evaluation, in which the prototypical LiProMo editor will be used in real-world modeling sessions. Therein, we will specifically investigate the communication patterns between domain experts and system analyst, allowing us to taylor the editor towards specific usage scenarios. Additionally, we will conduct controlled experiments to assess the impact of LiProMo on the maintainability of process models.

Acknowledgements. Gefördert aus Mitteln des vom Land Tirol eingerichteten Wissenschaftsfonds.
This research was funded by the Austrian Science Fund (FWF): P23699-N23.

References

1. Becker, J., Rosemann, M., Uthmann, C.: Guidelines of Business Process Modeling. In: Business Process Management, Models, Techniques, and Empirical Studies, Springer-Verlag (2000) 30–49
2. Rittgen, P.: Quality and Perceived Usefulness of Process Models. In: Proc. SAC '10. (2010)
3. Scheer, A.W.: ARIS - Business Process Modeling. Springer (2000)
4. Pinggera, J., Zugal, S., Weidlich, M., Fahland, D., Weber, B., Mendling, J., Reijers, H.A.: Tracing the process of process modeling with modeling phase diagrams. In: Proc. ER-BPM '11. (2012) 370–382
5. Hoppenbrouwers, S.J., Proper, E.H., van der Weide, T.P.: Formal Modelling as a Grounded Conversation. In: Proc. LAP'05. (2005) 139–155
6. Soffer, P., Kaner, M., Wand, Y.: Towards Understanding the Process of Process Modeling: Theoretical and Empirical Considerations. In: Proc. ER-BPM '11. (2011) 357–369

7. Arlow, J., Emmerich, W., Quinn, J.: Literate Modelling - Capturing Business Knowledge with the UML. In: Proc. UML '98. (1998) 189–199
8. Cordes, D., Brown, M.: The Literate-Programming Paradigm. Computer **24** (1991) 52–61
9. Mayer, R.E., Moreno, R.: Nine Ways to Reduce Cognitive Load in Multimedia Learning. Educational Psychologist **38** (2003) 43–52
10. Bertin, J.: Semiology of Graphics: Diagrams, Networks, Maps. Univ. of Wisconsin Press (1983)
11. Paivio, A.: Mental Representations: A Dual Coding Approach. Oxford Univ. Press (1986)
12. Moody, D.L.: The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. IEEE Transactions on Software Engineering **35** (2009) 756–779
13. Ottensooser, A., Fekete, A., Reijers, H.A., Mendling, J., Menictas, C.: Making sense of business process descriptions: An experimental comparison of graphical and textual notations. Journal of Systems and Software **85** (2012) 596–606
14. Dengler, F., Vrandecic, D.: Comparison of wiki-based process modeling systems. In: Proc. i-KNOW '11. (2011) 31–34
15. Sweller, J., Chandler, P.: Why Some Material Is Difficult to Learn. Cognition and Instruction **12** (1994) 185–233
16. Zugal, S., Pinggera, J., Mendling, J., Reijers, H., Weber, B.: Assessing the Impact of Hierarchy on Model Understandability-A Cognitive Perspective. In: Proc. EESSMod '11. (2011) 18–27
17. Knuth, D.: Literate Programming. The Computer Journal **27** (1984) 97–111
18. Pinggera, J., Zugal, S., Weber, B.: Investigating the Process of Process Modeling with Cheetah Experimental Platform. In: Proc. ER-POIS'10. (2010) 13–18
19. Mendling, J., Reijers, H.A., Cardoso, J.: What Makes Process Models Understandable? In: Proc. BPM '07. (2007) 48–63
20. Mendling, J., Strembeck, M.: Influence Factors of Understanding Business Process Models. In: Proc. BIS '08. (2008) 142–153
21. Mendling, J., Reijers, H.A., Recker, J.: Activity Labeling in Process Modeling: Empirical Insights and Recommendations. IS **35** (2010) 467–482
22. Zugal, S., Pinggera, J., Weber, B.: Assessing Process Models with Cognitive Psychology. In: Proc. EMISA '11. (2011) 177–182
23. Constantine, L., Lockwood, L.: Structure and style in use cases for user interface design. Object Modeling and User Interface Design (2001) 245–280
24. Rittgen, P.: Negotiating Models. In: Proc. CAiSE '07. (2007) 561–573
25. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: Proc. CAiSE '11. (2011) 482–496
26. de A. R. Goncalves, J.C., Santoro, F.M., Baiao, F.A.: A case study on designing business processes based on collaborative and mining approaches. In: Proc. CSCWD '10. (2010) 611 –616
27. Glinz, M., Seybold, C., Meier, S.: Simulation-Driven Creation, Validation and Evolution of Behavioral Requirements Models. In: Proc. MBEES '07. (2007) 103–112
28. Zugal, S., Pinggera, J., Weber, B.: Toward Enhanced Life-Cycle Support for Declarative Processes. Journal of Software: Evolution and Process **24** (2012) 285–302
29. Zugal, S., Pinggera, J., Weber, B.: Creating Declarative Process Models Using Test Driven Modeling Suite. In: Proc. CAiSE Forum '11. (2011) 1–8
30. Zugal, S., Pinggera, J., Weber, B.: The Impact of Testcases on the Maintainability of Declarative Process Models. In: Proc. BPMDS '11. (2011) 163–177