Bringing together BPM and Social Software

Completed Research Paper

Stephan Bögel
Virtimo AG
boegel@virtimo.de

Stefan Stieglitz
University of Muenster
stefan.stieglitz@uni-muenster.de

Christian Meske
University of Muenster
christian.meske@uni-muenster.de

ABSTRACT

Traditional BPM approaches are well understood for hierarchical organizations that cope with highly repetitive and well-defined processes. Unfortunately, they do not work well for the most prevalent types of processes and exclude most of the employees from participation. Especially collaborative ad-hoc processes in a team-based organization are not covered by BPM methods. Social software is a widely accepted and promising approach to support participation but lacks mechanisms for controlling processes. The paper presents a novel approach using roles. Based on a systematic literature review, this paper contributes a solution-architecture for collaborative BPM and presents collaboration patterns expressed through role models. This helps overcoming the model-reality-divide and lost innovation in the business process modeling area. Furthermore, role-model based collaboration patterns can be used to monitor and control collaborative ad-hoc processes.

Keywords

Social BPM, virtual communities, social software, ad-hoc processes, knowledge intensive processes, collaborative processes, conceptual modeling.

INTRODUCTION

Enterprise modeling and business process management are approaches to plan, control, and improve organizations with the use of conceptual models. Processes, organizational structures, and resources are modeled to manage the organization and/or the supporting IT-infrastructure. In this context a lot of modeling notations, languages, methods and tools have been developed in academia and adapted in larger companies. In this paper it will be shown that these approaches work well for hierarchically organized companies with a strict top-down management where business processes are highly repetitive, well-structured and allow a high degree of automation. Unfortunately, they do not work well for the most prevalent types of processes and exclude most of the employees from participating.

Gilbert (2010) shows in an example of a large bank, that only 2,5% of the business processes are highly complex, repetitive and allow a substantial automation. More than 60% of the processes are e-mail-based ad-hoc processes that are not covered by classical BPM methods. The remainder of the processes has an average complexity and is only partly manageable with prevailing process modeling methods. In contrast to well-defined business processes, ad-hoc processes emerge spontaneously, have a short lifespan, and are executed only a few times (Huth, 2004). This does not justify the effort of creating a process model. Moreover, ad-hoc cannot be planned in advance and have a high interactivity, which makes it difficult to automate them (Huth, 2004). Such ad-hoc processes can be seen as to what Bruno et al. (2011) calls the accelerated “pace of changes” as well as the “spreading of context information and the demand for quickly created process solutions” of BPM.

Knowledge intensive processes are a second type of processes that evade classical BPM methods. A process that is knowledge intensive typically appears in many scenarios, has deliberations and tradeoffs, is unstructured, and optimal sequences are not known in advance. Furthermore, these processes are evolutionary and involve distributed as well as
evolving knowledge. They are characterized by unknown participants and have infrequent performers, who value discretion over work methods (Markus et al., 2002).

Whereas traditional software focuses on productivity and process support, social media applications focus on enabling communication, cooperation, and collaboration of individuals and groups over the Internet. Social media is based on different services for setting up networks and supporting the distribution of information within the network (e.g. e-mail, instant messaging, chats, microblogging, social network sites, or blogs) (Stieglitz & Dang-Xuan, 2012). Following Hippenr and Wilde (2005) social software, also described as web 2.0 applications, are characterized by the following attributes: (1) it focuses on individuals or groups, (2) it relies on the self organization of the participants, (3) individuals contribute voluntarily, (4) actors’ roles change from an information consumer to an information provider, and (5) linkage of information is of crucial importance, instead of the information of individuals. Social media are defined as “a group of internet-based applications that build on the ideological and technological foundations of Web 2.0 that allow the creation and exchange of user-generated content.” (Kaplan and Haenlein, 2010). Internet forums, social network sites, microblogging, wikis, web logs, instant messaging, RSS, pod casts, and social bookmarking are understood as social media.

Social software is also heavily discussed as instrument which enable participation of all of the members of an organization especially in the context of improvement for company’s knowledge management (Stieglitz & Meske, 2012). As Vanderhaeghen et al. (2010) point out, social media can be understood as a paradigm that is based on self-organization and collective intelligence and that it allows a bottom-up peer-to-peer knowledge flow that is under the control of the user, informal and easy to use (Tuten, 2010).

The combination of BPM and social software is discussed under the terms subject-oriented BPM (Fleischmann, 2010), social BPM (Nurcan and Schmidt, 2009) and BPM 2.0 (Roychowdhury and Dasgupta, n.d.; vom Brocke et al., 2011). Based on previous research in this field we are seeking to answer the research question of how to overcome model-reality divide and lost innovation of BPM modeling. To answer this question we argue that a main driver to overcome this deficit is that real participation has to be achieved. This leads to the requirement that process design and process execution have to be merged. Furthermore, process-modeling software should be as easy to use as social software. If such a participation is achieved, a still unanswered question is how to monitor and control ad-hoc and knowledge intensive processes.

The proposed approach applies the concept of roles to allow the use of communication “atoms” as first class model-elements and to represent collaboration patterns through role models. This allows merging process controlling and execution as well as the monitoring and control of ad-hoc and knowledge intensive processes. The contribution of the paper is an architecture for social BPM relying on the role concept and the demonstration of collaboration patterns expressed by role models.
aspect of the approach. They also allow to dynamically type the relevant objects related to the collaborative business processes. In the evaluation phase we use a case study to show that the results of our research are of general interest, innovative, and well founded.

To build a fundament for our later proposed role based collaborative BPM approach, in the next three sections we, review and discuss relevant literature on the challenges of business process modeling in general, the meaning and importance of ‘roles’ within the conceptual modeling of BPM as well as the main opportunities of collaboration and users involvement.

CHALLENGES OF BUSINESS PROCESS MODELING

As van der Aalst (2013) has shown in his survey, business process management involves the four key activities model, enact, analyze, and manage. For the modeling of business processes, process models are created using modeling languages like BPMN, petri-nets or EPC. In the BPM approach Vanderhaeghen et al. (2010) point out that some implicit assumptions are made, which we will analyze in detail. First of all, process models are an abstraction from the real world to reduce complexity. As a result, exceptions are often not covered by the models as well as tasks that are difficult to be modeled.

A reason for this is the allocation of tasks in the modeling process. In conceptual modeling two major roles are distinguished, the model creator and the model user (Thomas, 2005). From the constructivist understanding of models, the model creator perceives the relevant part of reality and abstracts it to a conceptual model (Schütte, 1998). The model user, who communicates with the model creator until both found a consensus about the model, ideally defines what is relevant and what is not. Typically the model creator is a system analysis and modeling expert who gets the relevant information through surveys and interviews (Frank, 1999). The model user has no direct access to the process models and change request have to be treated by the modeling experts. As (Schmidt & Nurcan, 2009) point out, this separation can lead to a deficient acceptance of BPM models among employees who only act in the role of model users.

The same pattern of task allocation exists between the design of business processes (at build-time), the execution of the processes (at runtime) and the viewpoint conflict between the two. In summary, insufficient feedback is a major drawback of BPM (Vanderhaeghen et al., 2010).

One reason for this is the semantics problem, the disregarding of the language game. According to Wittgenstein (1990) the semantics of a language are constituted while using the language. Most of the model users are not involved in the creation of modeling languages and thus languages like BPMN seem to appear out of nowhere. In addition to this, most of the modeling languages are over-engineered. A majority of the modeling constructs are not used during modeling. As shown by Recker (2010) by the example of BPMN, the common core of the BPMN encompasses six elements out of 50. Another problem is the integration of the community language (domain language) and the modeling language (Kugeler and Rosemann, 1998). Ontologies or modeling conventions can be used to overcome this problem (Fallmann et al., 2010; Delfmann et al., 2009), but raise the same challenge of letting users participate in the ontology creation or formulation of conventions.

THE ROLE OF ROLES

A reason for the lack of innovation and model reality divide, showing in the absence of user participation, can be found in the architecture of modeling languages. Classical modeling languages, like UML (OMG, 2007) or BPMN, are specified using four layers of abstraction. These layers M3 to M0 are described by Rubaugh et al. (2010): On the topmost layer M3, the meta-meta-model is specified. This model is used to specify the subjacent layer M2, the meta-model. Every element of the meta-model is an instance of the overlying model-layer. The meta-model defines the grammar of the modeling language that is the abstract syntax of the language. It describes the concepts of the domain and their relations. On the basis of the modeling language a concrete user model can be defined on the layer M1 using the same instantiation mechanism. In the realm of business process management, the layer M0 represents a concrete process instance.

As one can clearly see, languages are developed and used in a strict top-down manner. They also focus on the description of the elements and their relations but not on the collaboration of the entities. In contrast, roles specify the collaboration of entities and thereby the important context of the entities.

A role is a concept that is founded and not semantically rigid. The first characteristic demands that a role is related to another role and thus describes the collaboration aspect of roles. For example the role “father” is related to the role “child” and expresses the relation between two entities. Semantically rigid means that the identity of an entity depends on a certain characteristic and cannot exist without it. For example “animal” and “dog” are characteristics that constitute the identity of an entity. In contrast, a role is not semantically rigid. An entity can play a certain role in a specific context and dynamically change the role played in a different context (Guarino, 1992).
A role can further be specified in delimitation to the concepts of the natural concept and the environmental role. In contrast to a role, a natural concept is not founded and semantically rigid (Guarino, 1992). “Dog” is an example for a natural concept. Less interesting are concepts that are founded as well as semantically rigid. “Color” is such an example and may be implemented as an attribute of a natural concept.

Roles are defined using role types (Riehle and Gross, 1998). A role type is a type that subsumes a set of similar roles which are assumed by various entities. Role types are defined using role models (Riehle and Gross, 1998). Inside the role model, restrictions between roles are defined. If R is the set of role types in a certain role model and A and B are two roles of R, Riehle and Gross (1998) define the following roles restrictions:

Role-implied: An entity that is playing the role A has also to play the role B. In the role model this is symbolized using an arrow with a white arrowhead. In the example an entity that is playing the role “root” also has to play the role “parent”.

Role-equivalent: Expresses a role-implied that is directed bi-directionally. An entity that plays role A has also to play role B and vice versa. In a role model this is symbolized using an arrow with white arrowheads at both ends.

Role-prohibited: An entity that plays role A cannot play role B. In the example an entity that plays the role “parent” cannot play the corresponding “child”-role. As notation a line with a “dead end” at each side is used.

Role-don’t-care: There is no restriction concerning role A and B.

Important for the understanding of role-restrictions is that they are valid in the context of the role model only (Riehle and Gross, 1998).

Roles are a well-known concept in conceptual modeling. On the meta-level M2, e.g. UML, the infrastructure roles are defined as association roles or classifier roles (OMG, 2007). They are used to further describe class relations in UML models. Following this understanding, roles are named places (Steimann, 2000). On the model layer M1, role-types are used to annotate class-diagrams and thus specify collaborations between classes (Riehle and Gross, 1998). For an example see Fehler! Verweisquelle konnte nicht gefunden werden.2b). Three classes “Customer View”, “Person” and “Employee View” are annotated with role-types to specify how they interact. Following this annotation approach, on the real-world layer

---

**Figure 2: Example of a) a Role-Model and b) an Annotated Class Diagram (Riehle and Gross 1998)**

Roles are a well-known concept in conceptual modeling. On the meta-level M2, e.g. UML, the infrastructure roles are defined as association roles or classifier roles (OMG, 2007). They are used to further describe class relations in UML models. Following this understanding, roles are named places (Steimann, 2000). On the model layer M1, role-types are used to annotate class-diagrams and thus specify collaborations between classes (Riehle and Gross, 1998). For an example see Fehler! Verweisquelle konnte nicht gefunden werden.2b). Three classes “Customer View”, “Person” and “Employee View” are annotated with role-types to specify how they interact. Following this annotation approach, on the real-world layer
M0 role instances are annotated to objects. This understanding is based on roles as adjunct instances (Steimann, 2000). For the proposed concept in this paper, we are using roles as adjunct instances to merge process design and process execution.

Table 1 summarizes our comparison of classical meta-model based modeling and role-based modeling. In meta-model based modeling, model-elements in the meta-hierarchy are related by an “instance-of”-relationship while in the role-based approach an object on layer M0 assumes a role defined on layer M1. This allows that an aggregated type characterizes an object and these types can be changed at runtime. Furthermore, the role models on layer M1 allow to express collaboration in a straightforward manner.

<table>
<thead>
<tr>
<th></th>
<th>Meta-model based modeling</th>
<th>Role-based modeling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relation to the model</td>
<td>Instance of</td>
<td>Plays role</td>
</tr>
<tr>
<td>Typing</td>
<td>Every element has one type</td>
<td>Aggregated types</td>
</tr>
<tr>
<td>Dynamic extensibility</td>
<td>Not possible</td>
<td>Possible</td>
</tr>
<tr>
<td>Focus on</td>
<td>Entities</td>
<td>Collaboration, Context</td>
</tr>
</tbody>
</table>

Table 2. Comparison of Concepts Class and Role

Role models have another advantage: they can be used to express patterns. In the context of software development, Riehle (1996) shows that role models can be used to express design patterns. We adapt this idea for the definition of workflow patterns.

OPPORTUNITIES OF COLLABORATION AND USER INVOLVEMENT

Since distributed work and virtual teams are getting more and more important, most business processes involve several collaborating individuals (Ho et al., 2009). The fluctuation of employees is also accelerating, which requires the generation of BPM theories and instruments that consider the increasing complexity of collaborative work. Furthermore, collaborative tools in the domain of Internet social networking (Riemer and Richter, 2010), such as wikis, social networking sites, or blogs, CAN be used to support the design, execution, and management of business processes.

Based on user-generated content, the upcoming web 2.0 communication enhances the possibilities and enables new kinds of individualized information sharing and interaction among employees. As Happ et al. (2007) point out, companies start to adapt social software for external usage as well as for supporting internal tasks (e.g. internal communication and collaboration among employees). However, social software is not only used for unstructured communication processes (e.g. finding an expert for a certain problem) but also to support well-defined processes which may affect workflows internally or those which are directed to external stakeholders.

Literature identifies two major related problems that BPM faces, which might be solved by adopting social software: model-reality divide and lost innovation (Erol et al., 2010). The notion of model-reality divide refers to differences between process modeling and execution. Even if business processes are adequately designed, there is a tendency that over time, the modeled processes and the real processes fall apart. The notion of lost innovation considers the high transaction costs that are associated with the transfer of improvement ideas between different process stakeholders (e.g. users, managers, and IT professionals).
Following Erol et al. 2010, the model reality-divide as well as the loss of innovations problem originates from the information pass-on threshold. Caused by high transaction costs (time and effort) for passing innovations, process owners and/or the user are less motivated to communicate them. Even if they externalize their ideas, the following process is often not transparent and employees get no feedback whether their idea has been realized. Strong regulations and slow change management also lower the motivation to actively communicate innovations. Therefore, important and valuable information is lost and potential improvements remain unrealized which increases the model-reality-divide. Additionally, there is another reason for the model–reality divide and lost innovations: the information fusion. Usually employees are not familiar with business process modeling. Therefore, users are forced to passively accept the processes created for them. Furthermore, the inadequate definition of terms by a top management-driven approach instead of a peer-to-peer basis may create a model–reality-divide (Erol et al., 2010). Such findings also suggest the usage of social software in BPM. The following proposition can be put forward: The use of social software in BPM lowers the threshold when information is passed, because it allows employees to continuously inform themselves about process updates. Particularly in the context of design accompanying control and customization activities in virtual communities, the deployment of a platform aimed at supporting necessary coordination processes for process modeling leads to reduced transactions costs (vom Brocke and Buddendick, 2004). Thus the transactions costs that are associated with the transmitting of innovations may further be lowered significantly (compare Erol et al., 2010).

We believe that research in this area must extend existing frameworks of BPM. Except for a few examples such as the framework for collaborative business engineering (vom Brocke and Thomas, 2006), collaborative work seems to be insufficiently represented in both current BPM research and practice (Niehaves and Plattfaut, 2011, vom Brocke et al., 2011). First, the application of design science research seems promising in order to fill this gap and learn more about the way business processes can be designed collaboratively. The analysis of best practices may well inform the iterative search process underlying any design science study (Hevner et al., 2004). Some software products have already integrated social software applications into their BPM suites (e.g. ARISalign), and may thus constitute a good starting point for related research. At the individual level in particular, there are a number of more specific issues that fellow IS researchers should consider. Employees are, for instance, typically not modeling experts. Empirical studies, both of qualitative and quantitative nature, can help to better understand the enablers and barriers of collaborative process modeling from both an employees’ and decision makers’ perspective. Additionally, methods are required that are appropriate for the measurement of the costs and benefits associated with collaborative business process modeling (e.g. Lattemann and Robra-Bissantz, 2010).

A ROLE-BASED COLLABORATIVE BPM APPROACH

Architecture

A collaborative BPM (C-BPM) platform is connected to social software services using a service-oriented architecture (see 4). Central element of the C-BPM platform is the interactive, collaborative workbench that is instantiated for every collaborative business process. Inside the workbench users can add microblogging-messages and shared documents as first class model elements. Furthermore tasks, goals, pictures, e-mail conversations et cetera can be created. All these elements can be connected, arranged and accessed graphically. These elements are natural concepts.

An access management component handles the users of the platform and restricts the access to authorized users. With the help of a role model editor, the users can create role models graphically. Inside the workbench, role instances can be attached to the process elements to further specify the collaboration inside the business process. With the help of the defined role
models, collaboration patterns are defined and the process instances can be checked for soundness. A collaboration pattern miner analyzes the role-annotated process instances to make suggestions for the improvement of a process. Furthermore similar process instances can be discovered and transformed into a well-defined process for reuse. This is a major step in monitoring and controlling collaborative business processes.

![Collaborative BPM platform](image)

**Figure 4: Solution Architecture**

The standard procedure to use the workbench is as follows:

1. The platform is started and configured by the service orchestration. This defines the existing natural concepts like “document”, “person”, “microblog”, etc.
2. Users instantiate the workbench, i.e. create a new collaborative process.
3. Users add information “atoms” (these can be tasks, people, goals, documents, pictures, e-mails, microblogs, wiki pages…) to the workbench and work on the workbench instance.
4. The information “atoms” are than annotated with role instances. Now the “atoms” become typed. The types depend on the role model that is applied (e.g. a role-model for customer management or the development of a new product)
5. Due to the restrictions formulated in the role model, the collaborating elements can be checked for soundness.

**Collaboration patterns**

Monitoring and controlling of the collaborative business processes is achieved by applying collaboration patterns expressed by role models. De Moor (2006) describes various collaboration patterns. A goal pattern for example shows the collaboration of community- and individual objectives. A communication pattern defines a desired communication interaction inside the community. A task pattern determines how certain entities need to collaborate to fulfill a specific task (De Moor, 2006).

Team communication patterns are called “genres” by Riemer and Richter (2010). They distinguish the following (very fine grained) collaborations:

- Ask question (ask for status-update, how-to question, decision)
- Coordinate others (provide feedback, delegate task, note to-dos)
- Share information (post links and references, post ideas, solutions)
• Discuss and clarify (offer opinion, clarification)
• Record information (Record data, Post team protocol)
• Provide update (event updates, task-status update, upon a decision, notify of upcoming events)

In the proposed approach a collaboration pattern would look as depicted in Figure 5. When considering for example goal patterns (taken from (De Moor, 2006)), a community objective can be to write a joint advisory report to be used in a policy making process. An individual objective for somebody can be to do the editing of particular sections of that report.

![Collaboration pattern diagram](image)

**Figure 5: Example Role-Model "Generate Report"**

In the upper part of Figure 5, the natural concepts are defined. This is automatically done during social software service orchestration. In the lower part, the involved roles are specified. The role model makes the following statements if it is applied to a workbench instance “create advisory report”:

- The auditor and the persons who work on the report need to be different persons (role-prohibited).
- The advisory report is created respectively validated by the experts respectively the auditor.
- The auditor is related to the goal “validate report”.

By annotating the workbench elements with roles, the process can be monitored (Is every role attached?, Is the goal reached?) and controlled (Is every necessary role involved?, Are auditors and experts different persons?).

**CONCLUSION**

As major drawbacks in applying existing BPM solutions to the problem of handling ad-hoc and collaborative processes, the model-reality-divide and lost innovations were identified. A reason for this is the separation of model design and execution. Furthermore, the lack of monitoring and controlling mechanisms for those types of processes has been observed during the literature review. A role-based solution was presented to counter this problem. It allows the creation of processes in a straightforward manner using relevant elements of social software as well as monitoring and controlling them by applying collaboration patterns.
The main contribution of this work is the application of the role-concept on the problem of combining BPM and social software for the management of collaborative, ad-hoc processes. A solution-architecture was outlined, a novel modeling procedure defined and the definition of a collaboration pattern shown. By using the concept of roles we are able to merge the process design and execution layer. Communication atoms as first class model elements realize a living model and the annotation of roles allows the monitoring and controlling of those processes. This is a prerequisite for real user participation.

Although this gives a new perspective on collaborative (or social) BPM, some work remains. We are currently building a software prototype, which then needs to be evaluated in a real-world case study. During the case study a set of collaboration patterns needs to be modeled and checked for its ability in monitoring and controlling such processes.

REFERENCES


