Why is BPMN not appropriate for Process Maps?

Abstract

Process map is an abstract depiction of all company’s processes and their relations. It provides an overview of how an organization operates without going into process details. It is often used as foundation for the detailed process modeling where BPMN (Business Process Model and Notation) is used for modeling the details of business processes. Regardless of BPMN’s existence, most process maps from practice have been designed with software programs not initially developed for process modeling. Thus, the question we aim to answer is: Why is BPMN not appropriate for designing process maps? To address this, we semantically map process map concepts with BPMN concepts and use the principles of the representation theory introduced by Wand and Weber (1995) to find out whether BPMN is complete and clear in terms of the process map meta-model. By this means, we were able to provide valuable implications for both research and practice.

Keywords: Process map, Process architecture, Representation analysis, Semantic mapping

Introduction

Business Process Management (BPM) is an approach adopted by many organizations for improving their operations and serving their customers more efficiently and effectively. BPM has developed from ideas of BPR (Business Process Reengineering) towards a permanent practice to improve processes in both incremental and radical ways (Kettinger et al. 1997). The benefits mentioned for BPM range from process transparency and standardization, to process improvement and automation. The key method that supports the steps towards reaping these benefits is business process modeling. In this context, a process model is a graphical representation of a business process and is expected to be more intuitive than, for instance, a corresponding textual description as it tends to eliminate the ambiguities of natural language.

A BPM initiative is often operationalized by following the phases of the BPM lifecycle. The BPM lifecycle is a framework that defines a systematic way of managing business processes. The lifecycle has been discussed in many studies (Davenport 1993; Kettinger et al. 1997; Weske 2012). The most recent and consolidated version is introduced in Dumas et al. (2013), which is a lifecycle that comprises six phases. It starts with process identification where the organization identifies all their processes and depicts them in a process map. The process map is a special kind of model that provides a holistic and abstract view of all processes of one organization and the relations between them, and aims to provide an overview of how the company operates as a whole without necessarily going into process details (Malinova et al. 2015; Malinova and Mendling 2013). This process map serves as a foundation for the subsequent phases of the
BPM lifecycle. First, it helps process analysts to identify and focus on important processes. Second, the identified relations between the processes help to better understand the interactions between them in a more detailed analysis (Malinova et al. 2015). In the subsequent phase of process discovery, the current state of a singular process shown in the process map is investigated and captured in terms of an \textit{As-Is process model}. While the process map describes the top-most abstract view of a process architecture (Malinova et al. 2015), the details of the As-Is process models relate to more fine-granular levels that refine parts of the process map (Malinova et al. 2013). In the phase of process analysis, the As-Is process model is analyzed for weaknesses. The identified weaknesses are then addressed in the process redesign phase and modeled in terms of a \textit{To-Be model}, which addresses these weaknesses. Here the process map is used as an aid to ensure that a change made in one process does not affect other processes related to it. The redesigned process models are implemented in the process implementation phase (e.g. employees are made aware that the processes have changed). Last, all relevant processes are continuously monitored and controlled in the sixth phase of the BPM lifecycle (Dumas et al. 2013).

As process modeling is the backbone of any BPM initiative, a number of well-defined process modeling languages, such as UML (Unified Modeling Language), EPC (Event-driven Process Chain) and BPMN (Business Process Model and Notation), have been developed to assist practitioners when modeling their business processes. BPMN has, in essence, integrated the concepts of other languages like UML and EPC, and gained a broad user acceptance in recent years (Chinosi and Trombetta 2012). Prior research has accentuated the capabilities of BPMN as opposed to the other modeling languages (Chinosi and Trombetta 2012; Dijkman et al. 2008; Recker 2010a; Recker 2011; Recker et al. 2009). This might be due to the developer's proactive reaction on current user requirements. Accordingly, BPMN has become the leading standard for business process modeling, and it offers a large list of elements supporting practitioners in modeling all aspects of their business processes (OMG 2013). Despite its wide user acceptance, from our collaboration with many of our industry partners we observe that organizations struggle with the design of process maps on the most abstract level of their process architecture, although they have adopted standard languages like BPMN or a comparable modeling language. It is striking to note that the collection of process maps that we have collected from our industry partners do not use any of the symbols that these languages offer and that no systematically defined modeling language is used at all. As a result, we are faced with a high heterogeneity of process map designs from practice; despite that modeling languages such as BPMN exist. The reason why practitioners seem so reluctant in using BPMN when designing their process maps has been our inspiration for conducting this study. Therefore, the research question we aim to answer is: Why is BPMN not appropriate for designing process maps?

We address this research question by the help of a conceptual analysis that is founded on a qualitative empirical research design. To this end, we complemented a literature review with a case study of a leading software development company. As a method, we use a formal semantical mapping technique from schema transformation for analyzing the relationship between process map requirements and BPMN concepts. This provides the foundation for an analysis that builds on principles of the representation theory introduced by Wand and Weber (1995) to test the extent to which BPMN is complete and clear for designing process maps. The rest of the paper is structured as follows. The next section briefly introduces BPMN and process maps, and gives insights into studies conducted on the representational capabilities of BPMN. The Research Design section elaborates on the methodology and data collection techniques we use to answer our research question. In the Findings section we show the results of our analysis. In the Discussion and Implications section, we discuss the findings of our study and the implications for research and practice. The Conclusion summarizes the paper.

\textbf{Background and related work}

In this section, we describe the essential concepts of BPMN and representational requirements of process maps.

\textit{BPMN}

Process modeling has its roots in flowcharting and has traditionally been used as a tool to support system analysis and software design (Curtis et al. 1992). In the context of business process reengineering, the application of business process modeling has been broadened to process innovation and process automation (Davenport and Short 1990; Hammer and Champy 1993). The notations that were defined
since then (e.g. BPMN) have often focused on exactly these application scenarios. Additional business scenarios such as human understanding, compliance management and standard conformance have been smoothly integrated (Dumas et al. 2013). On the other hand, the increasing adoption of BPM calls for concepts to manage the vast amount of business processes in organizations and consequently reduce their complexity.

In recent years, the BPMN specification has evolved to the de-facto standard for modeling business processes. Its primary goal is to facilitate the creation of process models that are readily understandable to different audiences, ranging from business users to technical users (OMG 2013). More specifically, three main application domains for BPMN have been identified: description, simulation and execution of processes (Chinosi and Trombetta 2012). As such, the focus is on the modeling of singular process in the context of discovering the process in a consistent way, calculating the resources needed during process execution, or translating a process model into a machine readable language in order for it to be automated. It has been argued that BPMN supports these applications fairly well (Chinosi and Trombetta 2012).

The elements included in BPMN to support these applications can be grouped into two categories, namely core graphical elements (12 elements) typically used by business analysts, and the extended set of BPMN elements (69 elements) which are often used by technical users with longer modeling experience. However, it has been empirically evidenced that the core graphical elements are hardly enough for business users to represent all their process requirements (Recker et al. 2006). Similarly, the 69 elements are often overwhelming for business users. In line with this observation, zur Muehlen and Recker (2008) found that typical BPMN users use only 20% of the BPMN vocabulary. The study shows that more than 50% of the BPMN diagrams they investigated used only the sequence flow, task, end event, start event, exclusive gateway, parallel gateway, pool and lane. These all belong to the core set of BPMN elements.

Numerous studies have been conducted on the evaluation of BPMN capabilities in terms of ontological expressiveness (Figl et al. 2013a; Recker et al. 2010; Recker et al. 2011). Most of them use the Bunge-Wand-Weber (BWW) representation model to investigate the capabilities and deficiencies of BPMN for process modeling (Recker et al. 2010). The BWW model specifies a set of well-defined ontological constructs to describe all types of real-world phenomena in the Information Systems (IS) domain (Wand and Weber 1993). These studies suggest that while BPMN is ontologically fairly good, it still has few deficiencies and such deficiencies in process modeling languages may hamper user comprehension (Figl et al. 2013a; Recker et al. 2010; Recker et al. 2011; Rosemann et al. 2006). In particular, the study by Recker et al. (2011) shows that lack of ontological completeness and clarity of BPMN translates to users facing difficulties when trying to understand BPMN process models. The authors suggest the standardization committee of BPMN to create more hands-on guidelines for using the core and extended sets of BPMN elements.

BPMN lacks capabilities to precisely articulate the scope and boundaries of a singular process (Recker et al. 2010). Wohed et al. (2006) argue that while BPMN has the capabilities to represent some aspects of data, the language has deficiencies in modeling the resource perspective of processes. Also, due to the ambiguities of the BPMN constructs “pool” and “lane”, BPMN users tend to use these two elements for purposes which are out of the scope of their defined semantics (e.g. for representing organizational units, business areas, and process grouping) (Recker 2010b). Similarly, the study by Recker et al. (2010) found that BPMN users often use the BPMN element “text annotation” for representing business rules. Thus, it appears as, when BPMN exhibits construct excess BPMN users often compensate by using existing elements which were not originally intended for that purpose. Moreover, it seems that notational deficiencies are common in languages that have matured with respect to their representational capabilities, which indeed is how BPMN has been developed (Figl et al. 2013a; Rosemann et al. 2006). BPMN has also been investigated from a cognitive expressiveness perspective (Figl et al. 2013a; Figl et al. 2013b). For example, according to Figl et al. (2013b) perceptual discriminability of symbols in process modeling languages is positively associated with comprehension accuracy and task difficulty, while semantic transparency and aesthetics are rather related to perceived cognitive load.

Even though different directions for improvement have been described for BPMN, it is the most used language for business process modeling and better in many aspects as compared to older modeling languages. Its extensive set of elements helps experienced process modelers to represent complex business processes at the level of technical details. In this way, BPMN is a valuable tool to support process
automation. As it might be challenging for inexperienced users, it is a powerful tool for complex modeling projects that lead into software implementation.

**Process maps**

A process map is a high-level holistic depiction of a company’s processes and the relations between them (Malinova et al. 2015). The main difference between a process model and a process map is that a process model is a singular process, or part of a process, shown on a process map. Thus, while a process model shows how one business process is done in details, a process map depicts how this process model works together and fits with the rest of the company’s processes. Besides processes and process relations, all process maps from practice also use categories in order to cluster processes that play the same role in the organization. For example, those processes a company considered as most important are typically clustered as Core processes. Often, those processes that generate profit for an organization are considered as core (e.g. Sales, Production). The rest of the company’s processes have either a strategic role (categorized as Management processes) or support the execution of the core processes (categorized as Support processes). Many process maps from practice also include additional concepts, such as process inputs and outputs, main process actors and process resources. Through the inclusion of additional concepts, organizations usually aim at strengthening the message the process map should convey, namely to show how the entire company operates without the need to go into process details (Malinova and Mendling 2013).

Although prior literature illustrates examples of process maps (Dumas et al. 2013; Fischermanns 2006; Harmon 2010; Weske 2012), all are adapted from industry. Moreover, most of these textbooks do not reference any existing modeling language that has been used for the design of the shown process maps. This is often because practitioners use software programs, which have not originally been created for process modeling (e.g. PowerPoint), when designing their process maps. This allegedly happens due to a lack of conceptual foundation for process map elements, hence it might be one of the reasons for the high heterogeneity of process map designs we observe in practice. The one most commonly used concept that comes from literature and can be seen in process maps from practice is the use of the value-chain diagram (Malinova et al. 2015). However, in most process maps only the core processes are represented as a value-chain, which implies sequential execution (Malinova et al. 2015).

In practice, we observe that organizations that typically use BPMN for modeling their business processes, or a similar process modeling language, do not use the same when designing their process maps. This might be due to a number of reasons. For example, while BPMN has been developed and is used for modeling business processes with their according details, the purpose of a process map is to show an abstract view of all processes that have been modeled using BPMN, however excluding the process details. Accordingly, the range of elements BPMN offers might be too overwhelming when trying to depict processes on a process map. Secondly, while process models are often used for more technical reasons, such as process improvement and process automation, the process map plays a more strategic role in organizations, such as providing an abstract overview of all company’s processes. Furthermore, whereas process models are normally used by employees who have some modeling experience, the process map is a document which often circulates among all company employees, including those with no modeling experience.

Regardless of the studies conducted about the capabilities and deficits of BPMN for process modeling, there are hardly any studies done on the reason why the design of process maps is far from the typical appearance of a BPMN model. We cannot yet state the real reasons behind why practitioners do not use this language when designing their process maps. It might be due to the ontological deficits BPMN exhibits. Nevertheless, the purpose of BPMN not fitting to that of process maps could be the cause of it. Conceptual modeling theory describes a model being bound to a specific purpose. Accordingly, a modeling language is bound to an entire class of purposes. Therefore, if a process model depicted using BPMN is commonly used to describe, improve or automate a singular business process, BPMN is accordingly used for all these listed purposes. In that regard, while BPMN is used to represent the fine-granularity of business processes, it simply might not be appropriate for the purpose of abstract process modeling. Presumably, BPMN does not fit the purpose of a process map and is therefore ignored by practitioners when undertaking this task. In view of that, we require deeper understanding of why process maps from practice contain elements which do not come from well-established modeling languages such as BPMN.
Accordingly, we conduct this study in order to give more insights into the suitability of BPMN for designing process maps.

**Research design**

In this section we elaborate on the methods we used for data collection and analysis.

**Data collection**

To be able to test the suitability of BPMN for designing process maps, we need all concepts used on a process map and concepts used for BPMN. The data for this study, concerning the process map meta-model, has been gathered during a one-week, in-depth qualitative study we conducted with one of our industry partners. Our industry partner is one of the world’s leading software development companies with more than 60,000 employees. During our 5-day stay at their premises in September 2014, we conducted 15 interviews. Our interview partners were mainly employees which are responsible for processes shown in their process map (e.g. process managers). These are typically employees responsible for the end-to-end execution of one or more processes. In addition we interviewed employees who are members of the BPM governance team of the company. They are involved during all stages of the BPM initiative, such as setting the BPM standards, the phases of the BPM lifecycle and ensuring strategic alignment. We also interviewed the company’s main process analysts, who are employees that lead the process modeling project and are thus directly involved with modeling the company’s business processes. The interviews can be grouped as follows: BPM governance team (6 interviews), process managers (8 interviews) and process analysts (1 interview).

The interviews were conducted following an interview guideline. Prior to our stay, we were made aware of most of our interview partners. Accordingly, we created two types of interview guidelines. One guideline consisted of questions regarding characteristics of singular processes or group of processes, such as questions revealing the relations between processes shown in the process map and process triggers. We used this interview guideline in the case of our interview partners being process managers. The second interview guideline included more strategic questions, such as questions regarding their process map, the role the process map plays within the organization and any means of guidelines the process map needs to comply too. We used the questions of this guideline to lead the interviews we conducted with the members of the BPM governance team. The interviews were semi-structured, thus they ended up in open discussions about the company’s processes that are part of their process map, the relations between them, and any concepts directly connected to a process and should be shown on their process map. As a result we ended up with approximately 10h of interview material (5h with process managers, 4h with BPM governance team, and 1h with process analysts). The 10h of material was transcribed using the transcription software F4 which resulted in just about 100 pages (font size 11) of interview transcripts.

**Data analysis (Process map concepts)**

To be able to argue the applicability of BPMN for designing process maps, first we need a list of all concepts necessary for designing process maps, and a list of all BPMN concepts available and used by practitioners for modeling business processes. Therefore, as foundation for process map concepts we take the process map meta-model provided by Malinova et al. (2015). Having already a process-map meta-model, the main purpose for conducting the case study with our industry partner and interviewing the 15 employees was to identify any material pertinent to the concepts shown in the meta-model in Malinova et al. (2015). We did this in order to be able to saturate the process map meta-model from Malinova et al. (2015). In view of that, we had three main targets for conducting the case study. First of all, as a result of the interviews we validated the concepts shown in the process map meta-model introduced by Malinova et al. (2015). Secondly, we were able to empirically assess the extent to which the concepts included in the meta-model are exhaustive and to make sure that the meta-model is saturated. In this way, we aimed to ensure that there are no other process maps from practice that will offer an additional concept that is not yet included in the meta-model introduced by Malinova et al. (2015). Since our industry partner is a very well-established software developer worldwide, and because of the company’s size, we found it suitable to ensure the saturation of the process map meta-model. Besides the validation of the existing concepts, we were also able to identify additional concepts that were not yet included in the meta-model, but indeed...
belong to the set of concepts for designing process maps. Third, we were able to define the semantics of each process map concept. As basis for the concept semantics we use the definitions presented in Malinova et al. (2015). The case study enabled us to validate the provided process map concept definitions, refine them when necessary, and also define the newly added concepts.

For the analysis, we followed the grounded theory approach, which fits well our purpose, as it is used to describe, explain and interpret collected data (Williams 2007). In particular, the Straussian grounded theory method offers three coding steps that lead to theory building (Strauss and Corbin 1998). However, since we are not aiming in building theory but only in validating the process map concepts, adding any missing concept, and defining their semantics, we only use the first two coding steps, namely the open and axial coding. During open coding we examined, conceptualized and categorized our interview transcripts (Strauss and Corbin 1998). We did this by going through our interview data and identifying all concepts our interview partners discussed as either already part of their process map or necessary to be included. This resulted in a list of concepts our industry partner considers important to be shown on a process map level. During the axial coding step, we categorized the derived concepts in line with the concepts from the existing process map meta-model. Hence, we matched the derived list of concepts to those shown in the process map meta-model by Malinova et al. (2015). In this way, we were able to validate the process map meta-model (Malinova et al. 2015). Additionally, those concepts that could not be matched to any of the concepts shown in the meta-model were concepts that extended the meta-model. During our data analysis, we were also able to confirm the validity of the provided concept definitions (Malinova et al. 2015). However, our interview material pointed us to additional attributes of existing concepts defined by Malinova et al. (2015). In such cases we refined the concept definition accordingly. We used the specialized qualitative analysis tool NVivo to assist us with the analysis of our data and to keep track of all resultant concepts.

As result of the data analysis, we arrived at a saturated list of concepts for process map design, along with their corresponding semantics. In order to discuss the capabilities of BPMN for designing process maps, we also need a list of BPMN concepts. Since BPMN is a very rich modeling language with an immensely long list of concepts, we only focus on the process modeling conformance class and use the list of the extended BPMN elements from this class (OMG 2013). We selected these mainly because the other BPMN conformance classes focus on process execution matters, thus their purpose is rather more technical. Given both lists, we were able to match the semantics of the process map concepts to the semantics of the BPMN extended list of elements.

Semantic mapping

We use semantic mapping in order to be able to match process map concepts to BPMN concepts. Semantic mapping is used for schema matching, which has been motivated by schema integration, a method used to integrate two independently developed schemas into a single schema (Rahm and Bernstein 2001). Researchers have identified various types of semantic relationships (Magnani et al. 2005; Rizopoulos and McBrien 2005). For our study, we use the four most commonly used semantic relationships: equivalence, subsumption, intersection and disjointness. We use these because they are well-grounded and well-defined in the research area of schema matching. They are also used to formalize the representational relationships that Wand and Weber (1993) describe. We adopt their definitions as formalized based on the construct domains D by Rizopoulos and McBrien (2005). These are:

- **Equivalence**: two schema constructs A and B are equivalent, \( A \cong B \), if and only if \( D(A) = D(B) \)
- **Subsumption**: schema construct A subsumes B, \( A \sqsupset B \), if and only if \( D(A) \subseteq D(B) \)
- **Intersection**: two schema constructs A and B are intersecting, \( A \cap B \), if and only if \( D(A) \cap D(B) \neq \emptyset \), \( \exists C : D(A) \cap D(B) = D(C) \)
- **Disjointness**: two schema constructs A and B are disjoint, \( A \cap B \), if and only if \( D(A) \cap D(B) \neq \emptyset \), \( \exists C : D(A) \cup D(B) \leq D(C) \)

Imagine A is a process map concept, and B is a BPMN concept. The relationships explained in natural language would be as follows: the equivalence relationship occurs between concepts A and B if their semantics are the same. When the semantics of concept A subsumes the one of concept B, it means that concept A includes the semantics of concept B and more. The semantics of concepts A and B are
**Why is BPMN not appropriate for Process Maps?**

*intersected* if they share some commonalities, but they are not entirely equal. Last, concepts A and B are *disjoint* if they have no elements in common. We manually matched the semantics of each process map concept to the semantics of each BPMN concept. The result is a table depicting all concepts that are related by one of the first three shown semantic relationships. The rest of the concepts that could not be matched are naturally those related by the disjointness relationship. We did this the following way: first, one researcher read the descriptions of all BPMN concepts from the extended list of elements and based on the semantic relationships mapped these to the process map concepts. A second researcher went through the list and modified the mappings which in the researcher’s opinion were not passable. At the end, after discussing the mapping results both researchers were able to reach a consensus.

**Representation theory**

As a result of the semantic mapping, we can use the representation theory principles to argue about the capabilities of BPMN for abstract depiction of business processes (Wand and Weber 1993; Wand and Weber 1995). In particular, we use the representation model as proposed by Wand and Weber (1995) as a basis for evaluating BPMN in terms of its ability to generate process maps that are good representations of the company's operations. Accordingly, we evaluate the language using two criteria: completeness and clarity (Wand and Weber 1995). In terms of completeness, BPMN would be complete, hence suitable for designing process maps, if and only if it contains all constructs that are required to depict all concepts that are shown in our extended process map meta-model. Thus, if all concepts shown in the process map meta-model entail the same semantics of concepts that BPMN already offers, we would consider BPMN as complete, thus suitable for designing process maps.

The second criterion, clarity, states that BPMN would be clear if and only if each of its constructs has a one-to-one correspondence with one of the concepts shown in the extended process map meta-model. This criterion points to three conditions that need to be satisfied in order for it to hold. Therefore, there should be no *construct overload* i.e. one process map meta-model concept maps to two or more BPMN constructs, no *construct redundancy* i.e. two or more process map meta-model concepts map to one BPMN construct, and no *construct excess* i.e. a process map meta-model concept does not map to any BPMN construct, BPMN would be clear, thus suitable for process map design. An expressive modeling language is one that satisfies both quality criteria completeness and clarity. We argue that if BPMN is not expressive in terms of the process map meta-model, the process maps that are designed using BPMN will be deficient (Wand and Weber 1995). Specifically, if BPMN was not complete, it might lack the constructs needed to convey meaning about some aspect concerning the company's operations. If BPMN was not clear, then it has constructs that convey ambiguous meaning about the company's operations. The following section shows our findings as result of our data analysis.

**Findings**

In this section, we discuss the findings of our study. First, we present the extended process map meta-model as result of the conducted case study and we explain the included concepts. Next, we provide the results of the semantic mapping between the process map concepts and the BPMN concepts. We illustrate the semantic mapping in a tabular form, where we indicate a semantic relationship, if such exists, between the concepts. Last, we show the results of the representational analysis where we elaborate on the representational completeness and clarity of BPMN for designing process maps as result of the semantic mapping.

**Process map (extended) meta-model**

Figure 1 depicts the extended process map meta-model. It is extended because, we took as foundation the meta-model for process map design from the study done by Malinova et al. (2015), we validated the included concepts by conducting the case study and extended the meta-model accordingly. As a result of the case study, we were able to validate all concepts included in the meta-model by Malinova et al. (2015). In addition, we found three concepts which should belong on a process map, but were not part in the original meta-model. These are *service* and *object*, which are concepts directly connected to a process, and a new type of trigger relation, namely the *specialization* relation, which is a relation between a standard process and its variants. In the following, we explain the meta-model concepts. In particular, we explain
the three newly added concepts in detail, whereas for the rest of the concepts, we provide a detailed description only in the cases of concept refinement i.e. new information due to the case study, otherwise they are briefly explained, as their detailed description can be found in Malinova et al. (2015).

**Process** is the main concept of process maps. A process belongs to a category depending on the role the process plays for an organization. A process could belong to a phase, depending on the time the process should be performed. A process is triggered by an input and when a process has finished, it gives a result in a form of an output. An actor is often assigned to a process. Some processes need resources in order for them to be performed while some offer services to employees. An object is commonly assigned to a process, which is something that is conquered due to the process performance. Processes are related to each other via different types of relations. Some process relations could occur only between processes coming from the same category, while some relations can only occur between processes that come from different process categories.

An **Input** triggers a process to start. According to the study by Malinova et al. (2015), an input triggers the execution of an end-to-end process i.e. a sequence of core processes. Whereas this is true, as result of our case study we found that there are two different types of inputs, namely external and internal inputs. External inputs come from a customer external to the company (not a company employee), while an internal input is one which is provided either by an employee or by another process. An external input can trigger an end-to-end process; however it can also trigger a singular process which is part of the process map (e.g. the core process Sales). On the other hand, an internal input can only trigger singular processes within the process map (e.g. all processes shown on a process map). Examples of external inputs are: a customer request to purchase a new product, a customer request to maintain an existing product, and a contractual requirement to ensure process quality. Examples of internal inputs are: an employee request to implement a new idea, and data outputted by a process needed for another process to start. One input can trigger one or more processes.

---

Figure 1. Extended Process map meta-model (using UML class diagram notation)
An **Output** is a result generated during process execution. Similar to input, there are external and internal outputs. **External output** is one which is used by an external customer. Any process that has been triggered by an external input produces an output which is sent to an external customer, which is typically the one that has triggered the process in the first place. Examples of external outputs are: a signed contract, a consulting service, and a customer training. In contrast, an **internal output** is one which is used by an internal customer, either being that a company employee, or another process. One process could produce one or more outcomes. In the process map meta-model from Figure 1, we consider the different types of inputs and outputs as attributes of both.

A **Resource** supplies processes with means they need in order to be performed. Example of a resource is “wood” needed for the production of furniture. A process does not necessarily need a resource in order for it to be performed, but it could also have as many as necessary.

An **Actor** is a person responsible for a process shown on the process map. They are typically called process managers and are those who are familiar with the entire end-to-end process. Process managers are also the ones who communicate with the employees that are involved with the process, usually called process performers. A process could have none, one or more assigned actors.

One process could offer **services**, which is an act provided by the company or a particular process to employees and is typically considered as a benefit. For example, the HR (Human Resources) process provides employees services such as sport courses and medical assistance. All these belong to services, which are not necessarily processes, but are provided by processes. One process need not offer services, but it could also provide access to many.

An **Object** is a thing which is associated with only one process. A process must not have an object, however when a process is assigned an object, the performance of this process must be able to attain the assigned object. Thus, the process performers during process execution must take into consideration, at all times, the object’s presence. The object resembles the *flow unit* in business processes. A flow unit is an item being analyzed. Depending on the process, it can be a unit of input, a unit of output or a financial value of the input or output (Anupindi et al. 2004). In addition to a flow unit, an object could also be a more abstract concept that must not necessarily be seen, but is at all times present. An example of an object for the process “production” would be “product quality”. This means that, during the process “production” is executing, all products that are produced have to be of high quality, as this is the object that should be kept in mind constantly during process performance.

A **category** is a cluster of processes which have a similar purpose. A process is always assigned to only one category. One process cannot belong to two categories. One category contains more than one process. Most common process categories are: management, core and support process categories.

A **phase** is a temporal category of processes. This means that all processes that are assigned to one phase are executed at the same time. A typical example of a company which includes phases comes from the manufacturing industry, where the production line of a product is strict and processes always need to be executed at a specific time before the next stage of the production can start. A phase could contain processes coming from different process categories.

There are four main types of **process relations**: trigger, dataFlow, manage and support. The **Trigger** relation could be used between processes coming from the same process category. It could also happen that processes from different process categories are related by a sequence relation. When this relation is used between two processes, it commonly indicates that there is one process that, when executed, triggers the other process which is related to. There are three types of trigger relations: sequence, decomposition, and specialization. One process could be related to another process, or more processes, with the **sequence** relation. When two processes are sequentially related, this can mean two things. First, the second process can start performing only after the first process has finished. This is the case presented in Malinova et al. (2015). Compared to the description by Malinova et al. (2015), we found that a sequence relation between two processes does not necessarily mean a strict sequential relationship. When two processes are sequentially related, the second process could also start without the first process having completed immediately before the second process has started. However, the first process must have been executed at least once before the second process could start executing. A suitting example to illustrate this case would be to consider the two core processes “production” and “sales”. Of course a product must be produced for it to be sold. However, a product that has already been produced can be sold at any time in
the future, as long as there are still products in stock. Only in the case there are no products left for selling the “production” process must be executed at least once before the “sales” process could start again. In addition, one process could trigger at the same time more processes. In this case, the processes that are triggered are performed in parallel. Similarly, the processes performed in parallel could in turn trigger one or more processes.

The decomposition relation relates a superordinate process with its subprocesses. This is a one-to-many relation, which means that one superordinate process could have a number of subprocesses. When the decomposition relation is used between processes it means that, once the superordinate process has started, all its subprocesses are triggered and must be performed in order for the superordinate process to finish its performance. The subprocesses are commonly related to each other with the sequence relation.

The specialization relation also relates one process to more processes. Contrary to the decomposition relation, the specialization relation relates a standard process to its variants. Process variants are processes which slightly vary from the normal flow of execution of its standard process. For example, in a furniture company a chair is produced differently than a table, however for both to be produced the process “production” is triggered. Thus, the process “production” has two process variants, one process “produce chair” and one process “produce table”, respectively. Both relations decomposition and specialization occur between processes within one process category.

Contrary to the other relations, manage and support are relations that can be seen between processes coming from two categories. As the name implies, the relation manage relates the management processes with all processes coming from the other process categories, (e.g. core and support processes). The management processes are there to manage the performance of all other processes from the process map. Similarly, the support relation is one between the support processes and the processes from the other process categories (e.g. core and management processes). When this relation is used, it means that the support processes are there to support all other processes from the process map. What was not included in the study by Malinova et al. (2015) is that, there could be subcategories of processes. For instance, there are cases where the support processes are categorized into “local” and “global”. This means that, the “local” support processes support only a certain number of processes from another category (e.g. the support process “audits” supports only the core process “production” by ensuring its quality), whereas the “global” processes are there to support all processes (e.g. the process “HR” supports all other processes of the company). Subcategorization can occur within the other process categories as well.

**Semantic mapping**

Table 1 depicts the results of the semantic mapping between the process map meta-model concepts and relationships from Figure 1 and the extended list of BPMN elements from OMG (2013). As seen from Table 1, in addition to the process map concepts, we also included the relationships between the concepts we see in Figure 1. We did this because some BPMN concepts could only be matched to the semantics of a relationship between two process map concepts. The table is organized in the following way. The first column is for the BPMN concepts, the second column is for the process map concepts and relationships, while in the third column the semantic relationship between the BPMN concept and the corresponding process map concept is defined. For example, the first row can be read as follows: the BPMN concepts “multiple start event”, “parallel multiple start event”, “intermediate catching message event” and “intermediate catching signal event” are semantically equivalent to the process map concept “input”. This means that, the concept “input” could be represented by all of the four aforementioned BPMN concepts. We sorted the rows of the table according to the semantic relationship. Hence, the concepts in the first eight rows (excluding the first row) are semantically equivalent, in rows 8 and 9 the BPMN and process map concepts are semantically intersecting. Rows 10-16 show the process map concepts which are subsuming BPMN concepts, while rows 17-22 the BPMN concepts are subsuming process map concepts. The last two rows are for all BPMN and process map concepts that could not be matched, hence they are semantically disjoint.

The semantic mapping results point to a number of interesting observations. First of all, only two BPMN concepts have a one-to-one mapping to one process map concept, and only one of them share the same name, that is the BPMN and process map concept “category”. Accordingly, these share the same semantics. The second one-to-one mapping occurs between the BPMN element “expanded sub-process” and the process map concept “decomposition-trigger relation”. Although these two concepts do not
Why is BPMN not appropriate for Process Maps?

belong to the same category of concepts, one being a type of process, the other a type of relation, both concepts relate a superordinate process to its subprocess, hence they are semantically equivalent. Most of the rest of equivalence mappings are many-to-one which means that there are usually more than one BPMN concepts that entail equal semantics with only one process map concept. The only semantic intersections occur between the “none start/end events” and the “input/output” process map concepts. This is because both BPMN and process map concepts are defined in a way that they share some commonalities, however both have additional attributes that the other cannot be matched to.

One of the most interesting findings from Table 1 is that the majority of mappings between BPMN and process map concepts are semantic subsumptions. This means that, most concepts of the one are specialization of the other. Considering the purpose of process maps as compared to the purpose of BPMN models, especially the fact that process maps are abstract representations of processes, while BPMN models are detailed descriptions of the processes shown in process maps, it simply makes sense that there are many subsumption relationships. For example, the “message/timer start events” and “data input” are subsumed by the process map concept “input”. Hence, both “start events” and the “data input” are specializations of “input”. This was likely, because a process map “input” can take the form of a message, but a process in a process map can also be triggered by a contractual requirement, which might be in form of a reminder. In the same manner, all three BPMN concepts “message/signal end events” and “data output” are specializations of “output”. Similarly, the semantics of the BPMN element “data object” is subsumed by the semantics of the process map concept “object”, because an “object” could be a data object, but not necessarily.
Why is BPMN not appropriate for Process Maps?

<table>
<thead>
<tr>
<th>BPMN (B)</th>
<th>Process map meta-model (P)</th>
<th>Semantic relationship</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start events</strong> (multiple, parallel multiple); Intermediate catching events (message, signal)</td>
<td>Input</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>Multiple end event; Intermediate throwing events</strong> (message, signal)</td>
<td>Output</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>Expanded sub-process</strong></td>
<td>Decomposition-trigger relation</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>Gateways</strong> (exclusive, event-based, inclusive)</td>
<td>Specialization-trigger relation</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>Flows (normal, uncontrolled)</strong></td>
<td>Sequence-trigger relation</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>Message flow</strong></td>
<td>triggers; yields</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>Category</strong></td>
<td></td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>None Start event</strong></td>
<td>Input</td>
<td>B ∩ P</td>
</tr>
<tr>
<td><strong>None End event</strong></td>
<td>Output</td>
<td>B ∩ P</td>
</tr>
<tr>
<td><strong>Start events</strong> (message, timer); Data input</td>
<td>Input</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>End events</strong> (message, signal); Data output</td>
<td>Output</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>Gateways</strong> (parallel, parallel event-based)</td>
<td>Sequence-trigger relation</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>Data object</strong></td>
<td>Object</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>Lane</strong></td>
<td>Actor</td>
<td>B ⊆ P</td>
</tr>
<tr>
<td><strong>non-atomic, compound Activity</strong></td>
<td>Process</td>
<td>P ⊆ B</td>
</tr>
<tr>
<td><strong>Group</strong></td>
<td>belongs to</td>
<td>P ⊆ B</td>
</tr>
<tr>
<td><strong>Category</strong></td>
<td>Phase</td>
<td>P ⊆ B</td>
</tr>
<tr>
<td><strong>Participant</strong></td>
<td>Actor</td>
<td>P ⊆ B</td>
</tr>
<tr>
<td><strong>Association</strong></td>
<td>Relations (dataFlow, manage, support); triggers, yields; uses; is responsible for; provides access to; attains; is performed in; belongs to</td>
<td>P ⊆ B</td>
</tr>
<tr>
<td><strong>Directional association</strong></td>
<td>internal Input; internal Output; Relations (manage, support); triggers, yields; uses; is responsible for; provides access to; attains; is performed in; belongs to</td>
<td>P ⊆ B</td>
</tr>
<tr>
<td><strong>/</strong></td>
<td>Resource; Service</td>
<td>P ⊆ B</td>
</tr>
<tr>
<td><strong>Start events</strong> (conditional, signal); Intermediate throwing &amp; catching events (none, timer, escalation, compensation, conditional, link, multiple, parallel multiple); End events (escalation, error, cancel, compensation, terminate, multiple); Task (atomic); Choreography task; Sub-process (collapsed, nested/embedded); Sub-choreography (collapsed, expanded); Complex gateway; Flows (conditional, default, exception); Compensation association; Looping (activity, sequence flow); Multiple instances (sequential, parallel); Process break; Transaction; Off-page connector;</td>
<td>/</td>
<td>P ⊆ B</td>
</tr>
</tbody>
</table>

Table 1. BPMN – Process map meta-model semantic mapping

There are a similar number of subsumption mappings showing the case when BPMN concepts subsume process map concepts. For example, the main concept of process maps “process” is a specialization of the BPMN concept “activity”, because an “activity” could be atomic and non-atomic, but a process map “process”, as defined, can only be atomic. Similarly, a “phase” is subsumed by a BPMN “category”, because a “category” does not restrict the time when its contained processes will be executed, while a “phase” does exactly that. Moreover, an “actor” is a specialization of “participant”, because a “participant” could be a person that is responsible for the execution of a process just as an “actor” is, however a “participant” could also be a buyer, a seller, or a manufacturer, while an “actor” cannot.

The last semantic subsumptions that could easily be spotted from Table 1 are between the BPMN concepts “association” and “directional association” and the process map relations “manage/support” and all of the relationships between the process map concepts. These are one-to-many relationships since the semantics
of both “association” and “directional association” elements entail that they can be used to link information and artifacts with any BPMN graphical element and other artifacts. Thus, these two BPMN elements could be used to relate a “process” with all other process map concepts. In particular, it is interesting to mention that these are the only BPMN elements that could be used to express the semantics of the relations “manage” and “support” in a process map.

Last, we can see that there are only two process map concepts (“resource” and “service”) that cannot be mapped to any of the BPMN concepts we investigated, while there are many more BPMN concepts that cannot be mapped to any of the process map concepts from the meta-model on Figure 1.

**Representation analysis**

The semantic mappings from Table 1 also help us assess the completeness and clarity of BPMN and its suitability for designing process maps in terms of the representation theory and the principles of completeness and clarity as defined by Wand and Weber (1995). Accordingly, BPMN is complete if and only if it contains all concepts that enable to depict all process map concepts from Figure 1. If BPMN is complete, it is suitable for designing process maps. However, as we can observe from Table 1, BPMN is not complete in terms of the process map meta-model, because the two process map concepts “service” and “resource” cannot be semantically mapped to any of the investigated BPMN elements. Accordingly, we can claim that BPMN does not satisfy the principle of completeness because it does not offer all elements that could be used to represent all process map concepts.

In terms of clarity, BPMN is clear if and only if each of its constructs has a one-to-one correspondence with one of the process map concepts. To test for clarity, we must check for cases of construct overload, redundancy and excess. In Table 1, we spot that all three cases have occurred. First, one process map concept maps to more than one BPMN concepts, which is the case of construct overload. Similarly, there are cases when one BPMN element is mapped to more than one process map concept, hence construct redundancy occurs. Last, there are two process map concepts that do not map to any BPMN element, and vice versa, hence construct excess is present. Accordingly, we can conclude that BPMN is not clear in terms of the process map meta-model.

**Discussion and implications**

The following two subsections discuss the findings of our study, and state the implications for practice and research, respectively.

**Discussion**

First of all, as result of the semantic mapping and the representational analysis, we found that BPMN is not complete with respect to process map design. Therefore if an organization would use BPMN to design their process map, they might lack the necessary concepts needed to represent all requirements they aim to depict. This will most likely happen in case an organization intends to include the concepts “service” or “resource” in their process map, as there are no BPMN concepts that even remotely relate to the semantics of these two process map concepts. Furthermore, we also found that BPMN is not clear in terms of the process map meta-model. Thus, when an organization uses BPMN to design their process map, they will possibly face difficulties in choosing between the various BPMN elements that entail the same semantics as one process map concept. For example, a process map designer has the option to choose between three different types of BPMN gateways (exclusive, event-based, and inclusive) in order to represent process variants (“specialization-trigger relation”).

Similarly, since there are two or more process map concepts that relate to only one BPMN concept, if one BPMN concept is used to represent different process map concepts, the resultant process map would be prone to misinterpretation and would likely convey ambiguous meaning about the company’s operations. Let’s take as an example the inclusion of the BPMN concept “association” in a process map to capture the semantics of both process map relations “manage” and “support”. First of all, when two processes are related by a “manage” relation, it means that processes from one category manage the performance of all processes from the other categories. If the “support” relation is used, this means that processes from one category support the execution of all processes from the other process categories. Thus, using the same
Why is BPMN not appropriate for Process Maps?

BPMN concept to represent both would be a subject of misinterpretation, as “association” does not imply neither manage nor support. To make sure that process map users would infer the correct semantics of both, the BPMN element “text annotation” might be attached to each giving extra explanation. Nevertheless, our intention is to avoid exactly such cases of users mitigating construct excess by using supplementary elements. In addition, if “association” is used to relate a process map “process” with the rest of the concepts included in the map, the meaning of “manage” and “support” might even get more unclear.

As result of the semantic mappings, it is reasonable to assume that BPMN, primarily used for modeling the fine-granularity of business processes, is not applicable for abstract modeling. If BPMN would be used for designing process maps, first, a user with modeling experience would likely misinterpret the concepts on the process map, as the user has probably seen and used the same concepts in detailed process models. Second, many studies point to the complexity of BPMN due to its richness of elements, thus a user with less or no modeling experience would likely not understand a process map designed with BPMN, which could cause potential user resistance. Nonetheless, even semantically equivalent concepts might be misinterpreted when used for different purposes. For example, using the BPMN element “activity” to represent a process on a process map might be misleading because it could infer a single activity, when indeed it is an end-to-end process which does not expand to its constituent parts.

Implications

The findings of this paper point to implications for both research and practice. In terms of implications for practice, our findings show that BPMN is not fitting for process maps, thereby making the necessity for a dedicated language for process map design apparent. Such a language would support practitioners when designing their process maps as it will release them from the burden of, first relying on their own creativity when designing process maps, and second choosing among the numerous elements existing process modeling languages offer in order to be able to capture their requirements. Most importantly, having a language which offers appropriate elements for all concepts shown in the process map metamodel from Figure 1 would assist practitioners in ensuring they depict a correct overview of their company’s operations, accordingly decreasing threats of potential misinterpretation. Furthermore, as consequence of BPMN initiatives, organizations are typically faced with a vast amount of process models. Accordingly, a process map could be used as a tool to abstract from the details of many process models to a single model which captures the essence of their performance.

In terms of implications for research, finalizing the syntax, semantics and notation of such a language for process map design is still a subject that needs to be addressed. Our findings show that, while BPMN is deficient in terms of the principles for completeness and clarity as defined by Wand and Weber (1995), it still has potential for representing certain aspects of process maps using its elements. In view of that, the process map language should be developed such that it is user-friendly and familiar; hence it offers elements easy to understand by both experienced and non-experienced users. Second, it should be integrated with the existing process modeling languages in order for practitioners to utilize it as complementary to their existing process models. This will in turn enable linking the details of process models to their superordinate process shown on the process map. Moreover, research about the design of process maps could complement prior research on managing process model collections (Dijkman et al. 2012). However, compared to the existing literature which has a more technical focus, process maps could be used to understand process model collections from a more strategic perspective. In this way, managing process collections could take the turn of adopting a top-down approach, rather than the bottom-up one followed until now.

Prior research has raised awareness of the significance of good quality of modeling languages in terms of the BWW representation model (Recker et al. 2010; Recker et al. 2011). However, while such quality is indeed necessary, it might not be sufficient for the uptake of a language. The cognitive fit theory states that, for a model to be used for problem solving a cognitive fit must exist between the way a model has been created and the goals the model should accomplish (Vessey 1991). As a model is bound to a specific purpose, these can be lifted to the modeling language which is bound to all purposes of the models created by it. Thus, although BPMN has been developed for the purposes of discovering, simulating and executing business processes (Chinosi and Trombetta 2012), process modeling in abstract terms has hardly ever been considered as one of the reasons for using BPMN. Hence, another complementary perspective of this
Why is BPMN not appropriate for Process Maps?

Conclusion

We conducted a study to assess the suitability of BPMN (the standard for process modeling today) for designing process maps. We addressed this by the help of a conceptual analysis that is founded on a qualitative empirical research design. First, we complemented a literature review by conducting a case study with a leading software development company. As result, we provide a meta-model with saturated concepts for process map design. We use a formal semantical mapping technique from schema transformation for mapping the process map concepts with BPMN concepts. We do this in order to analyze the relationship between both, hence test for the capabilities of BPMN to represent all process map requirements. Last, we use the semantic mapping as foundation to analyze BPMN in terms of the principles for completeness and clarity introduced by Wand and Weber (1995). Based on this, we were able to test the extent to which BPMN is complete and clear for designing process maps. We found that BPMN is neither complete, nor clear for this purpose. Thus, if organizations use BPMN for designing their process map, they will encounter multiple BPMN elements which entail the same semantics as one process map concept, and vice versa, one BPMN element could also be used to represent multiple process map concepts. Our findings illustrate many concepts as specializations of others. This makes sense mainly because of the differing purposes of BPMN models and process maps. That is, while BPMN models show the details of a process, the purpose of a process map is to depict an abstract overview of all company's processes, hence they show how BPMN models fit together, excluding their details.

We point to implications for research and practice. For practice, due to our study we were able to show that BPMN is indeed not suitable for designing process maps. Thus, we point to the necessity of developing a dedicated language for process map design which will support practitioners when undertaking this task. In terms of research, the method we used to test the suitability of BPMN for process maps could be generalized to a method for conceptual fit analysis of modeling languages and their suitability for a particular purpose.

References


Why is BPMN not appropriate for Process Maps?


