2012 Eighth International Conference on the Quality of Information and Communications Technology

From Business Process Modeling to Data Model: A systematic approach

Estrela Ferreira Cruz
Escola Superior de Tecnologia e Gestão
Instituto Politécnico de Viana do Castelo
Viana do Castelo, Portugal
e-mail:estrela.cruz@estg.ipvc.pt

Ricardo J. Machado and Maribel Y. Santos
Centro de Investigação Algoritmi
Escola de Engenharia da Universidade do Minho
Guimarães, Portugal
e-mail:{rmac,maribel}@dsi.uminho.pt

***Abstract*—Business process modeling and management approaches are increasingly used and disclosed between organizations as a means of optimizing and streamlining the business activities. Among the various existing modeling languages, we stress the Business Process Model and Notation (BPMN), currently in version 2.0. BPMN is a widespread OMG standard that is actually used either in academia and in organizations. BPMN enables business process modeling, but does not facilitate the modeling of the information infrastructure involved in the process. However, interest in the data and its preservation has increased in BPMN's most recent version.**

The aim of this paper is to study BPMN 2.0, particularly on the usage and persistence of data, and present an approach for obtaining an early data model from the business process modeling, which may then be used as a starting data model in the software development process.

I. Introduction

The globalization of the markets and the constant increase of competition between companies' demand constant changes in organizations in order to adapt themselves to new circumstances and to implement new strategies. Organizations need to have a clear notion of their internal processes in order to increase their efficiency and the quality of their products or services. This will enable to increase the benefits for their stakeholders. For this reason, many organizations adopt a business process management (BPM) approach. BPM includes methods, techniques, and tools to support the design, enactment, management, and analysis of such operational business processes [1].A business process is a set of interrelated activities that are executed by one, or several, organizations working together to achieve a common business purpose [2].

There are several languages and tools that can be used to model business processes such as Petri nets, EPC (Event-driven Process Chains), IDEF (Integrated Definition Methods), BPMN (Business Process Model and Notation), among others [3]. In this paper BPMN is used because it is one of the best known standards and it is actually used in organizations. According to Andreas Meyer [4], BPMN is a modeling language really well accepted in companies and that receives the influence of these, as is the case of SAP, Unisys and Oracle.

If from one side the business process management and modeling are increasing their relevance, on the other side the software development teams still have serious difficulties in performing elicitation and defining the applications requirements [5]. In fact, one of the main software quality objectives is to assure that a software product meets the business needs [5]. For that, the software product requirements need to be aligned with the business needs, both in terms of business processes as in terms of the informational entities those processes deal with. This drives us to the question: "How can we use Business Process Models to design software applications?".

Researchers and professionals in information systems have recognized that understanding the business process is the key to identify the user needs of the software that supports it [6]. However, the tasks of business process analysis and software development are managed by different groups of people and commonly use different languages.

BPMN is a business process modeling language developed by OMG with the aim of providing a language easy to understand and usable by people with different roles and training from top managers to information technology (IT) professionals [7], [8].

When dealing with the activities of the business process it is inevitable to mention the data involved, or the information that flows throughout the process. So, to enable process control and business supporting software development, it is needed to store that information. However, as referred by OMG, data modeling is not a BPMN 2.0 goal [8]. Nevertheless, data is a key component whose relevance has increased, not only as a support to the business itself, but also for Business Intelligence (BI) operations [4]. Therefore, the data model is a fundamental model for designing software applications.

This paper addresses data persistence in BPMN 2.0 and presents an approach for obtaining a data model from the business process model, which may then be used in the software development process and this way assure that the data model fits the business process needs.

The remainder of this paper is structured as follows. In the next section, some related work is presented. In section III a brief description of BPMN 2.0 language is made, giving special attention to how data and its persistence is represented. In section IV our approach for obtaining the data model from BPMN is described. Finaly, conclusions and some references to future work are presented.

978-0-7695-4777-0/12 $26.00 © 2012 IEEE
DOI 10.1109/QUATIC.2012.31

II. Related work

Although previous work about data modeling within BPMN already exists, to our knowledge, all previous work is related with BPMN versions prior to 2.0 and none of them addresses the attainment of a data model that operationally supports the business process. In these studies some flaws have been identified, by the authors themselves, especially to distinguish persistent from non-persistent data.

Arnon Sturm [9] sugests a method for building a data warehouse schema for specifying business processes in order to allow the off-line analysis of business processes' execution. The data warehouse schema is composed of a small set of "snowflakes". A snowflake is the basic structure of the data warehouse. For each data object affected by the process a snowflake schema is created. The method comprises two steps: the first step is the creation of the data schema of each snowflake, the second populates the database with relevant data. The proposed method can generate schemas for all "snowflakes" of the business process specification. However, the creation of multiple schemes will overwhelm the design of the warehouse, so Arnon Sturm argues that one must carefully select the business processes to consider.

Brambilla *et al.* [10] explore BPMN for the generation of the data design, business logic, communication and representation. The authors separate the different concerns in different model types and interpret the BPMN in order to meet the needs of a Rich Internet Application. With respect to the data, the authors use BPMN data objects to identify the data. To distinguish between persistent and volatile data, the authors have chosen to identify, in the process model itself, persistent data with a 'P' and volatile data with a 'V'.

Magnani and Montesi [7], after identifying the gaps in data modeling using BPMN, proposed an extension to BPMN 1.2 with the aim of improving the representation of data. Their extention was named BPDMN (Business Process and Data Modeling Notation). Some of the concepts proposed, namely a way to identify the existence of persistent data were included in BPMN 2.0 [7] with the introduction of the data store, although with a different graphic symbol.

Wohed *et al.* [11] make an assessment of BPMN capabilities, its strengths and weaknesses, to model a business process and conclude that, in BPMN 1.2, data are only partially represented.

III. The BPMN Language

BPMN 2.0 provides three main types of diagrams [8], [12]:

- **Process diagrams** - Defines a set of business activities carried out by an organization for the concretization of a goal (product or service). The business process includes the flow and use of information and resources.
- **Choreography diagrams** - This is a type of process diagram that describes how participants coordinate their interaction. A participant, in general, defines a role in the organization or a business partner and is represented in a pool. The exchanged information is represented by the incoming and outgoing messages.
- **Collaboration diagrams** - This type of diagram focuses on the exchange of information between participants, represented by pools. The processes communicate by exchanging messages.

The basic process models can be grouped into two types of processes [8]:

- **Private Business Processes** - A private process is a process internal to a specific organization. Each private process is represented within a Pool.The process flow must be in one pool and should never cross the boundaries of that Pool. The interaction between distinct private Business Processes can be represented by incoming and outgoing messages.
- **Public Processes** - A public process represents the interactions between a private Business Process and other Processes or Participants. Only activities that are used to communicate with the other participants must be included in the public process.

The Business Process Diagrams use a set of graphical objects that can be grouped into five basic categories [8]:

- **Flow Objects** - are the main graphical elements to define the behavior of a Business Process. There are three kinds of Flow Objects: Events, Activities and Gateways.
- **Data** - represent the data involved in the process.
- **Connecting Objects** - model the connection between the several process elements. There are four types of connecting objects: Sequence Flows, Message Flows, Associations and Data Associations.
- **Swimlanes** - represent the participants in the process. A participant is a person, or something, involved in the process. Participants in the process can be grouped into pools or, more particularly, in Lanes. A pool can be divided into several Lanes, for example, to represent the different departments of an organization.
- **Artifacts** - are used to provide additional information to the process, such as a note ("Text Annotation").

The following subsection addresses data in BPMN 2.0, mainly its representation and flow.

A. Data on BPMN

Process modeling must be able to model the data items (physical documents or electronic information) that are created, manipulated, and used during the execution of a process [8]. The data involved in the process can be considered persistent or not persistent (volatile). The persistent data is the one that remains beyond the life cycle, or the scope, of the process [8].

In BPMN 2.0, the elements that manipulate data are "ItemAware elements", i.e., are elements that allow the storage and transmission of items during the process flows [8]. As stated in [8], "the data structure these elements hold is specified using an associated ItemDefinition". The item definition involves the specification of the data that are stored or transferred. On the "itemDefinition" properties there is a reference to the structure definition (structureRef). This reference could point to a file

location (usually a XML schema[1]) where the data structure is defined. This specification is optional.

In BPMN 2.0 the data can be represented in a process diagram by the elements presented in Figure 1. Data manipulation elements can be grouped into:

	Name	Symbol	Description
Data Objects	Data Object		Data objects represent the information needed or produced by the activity. A Data object can be referenced by **DataObjectReference**
	Data Object Collection		Represents a data objects' collection. (Attribute **isCollection**: boolean = true)
	Data Input		Represents the information needed for the process execution
	Data Input Collection (input set)		Represents a data input collection (Attribute **isCollection**: boolean = true)
	Data Output		Represents the information produced by the process execution
	Data Output Collection (output set)		Represents a data output collection. (Attribute **isCollection**: boolean = true)
	Data Store		Represents the persistent data stored or retrieved by one activity. A Data Store can be referenced by **DataStoreReference**

Fig. 1. Data representation elements (adapted from [8])

- **Data Objects** - data objects represent the information that flows through a process. A data object can be referenced by DataObjectReference. A data object reference is a way to reuse one data object in the same diagram. A data object reference can represent a different state of the same data object at different points in the process. On a process diagram this is represented by $< DataObjectName >$ [$< DataObjectReferenceState >$] [8].
- **Data Store** - a data store is a mean to handle persistent data. It provides a mechanism for an activity to store information or use the information stored. A data store can represent paper documents (a file folder, an agenda, a notebook, etc.) or an electronic database [12].

Data objects and data stores are exclusively used in process diagrams [8].

During the process execution, resources and/or data are consumed and produced. The transmission of the data created or used during a process execution can be represented by:

- **Messages** - A Message is used to represent the contents of a communication between two participants. Each participant is represented by a different pool. So, a message crosses the pool boundary to show the interactions between separate private Business Processes [8]. A message can represent any kind of information like an email, a fax, a letter, a phone call, etc. [12]. Graphicaly, an initiating message is represented by a white envelope. An non-initiating message is represented by a gray envelope [8].

[1]The default XML schema is presented in: http://www.w3.org/2001/XMLSchema.

- **Data Associations** - A Data Association can be used to model how data comes and goes from activities or events within a pool. This way is possible to identify the activity that sends data to a data store and the activity that gets data from a data store.

The next section presents an approach to obtain the persistent data model that supports a process modeled with the BPMN 2.0 Business Process diagram.

IV. Data modeling

Business process management initially focused its attention on designing and documenting business process, in order to describe which activities are performed and the dependencies between them [4]. Data did not receive much attention. Evolution and practice have extended BPMN attention to data. An example of this are the changes made from BPMN 1.2 to BPMN 2.0, where there has been an increase of the data relevance, particularly with the introduction of new model constructs such as "Object data collection" and "Data store". However, BPMN aims to support the modeling concepts that are applicable to business processes, leaving out aspects such as data models. As stated in [8], BPMN does not provide model elements for describing the data structure nor a language for data manipulation.

If one wants to model the data involved in the business process, then it is needed to include in the business process diagram the information about all data involved in the process. However, to avoid increasing the model complexity with this extra data modeling perspective, which could turn business process essential features vaguer, this approach should be applied as a further step to modeling the business process and the resulting model should be kept as a branch.

As stated by Brambilla *et al.*, one issue to be considered in the creation of a BPMN diagram is the level of granularity, or detail, which shall be applied [10]. In our approach, the level of detail should be large with respect to the data. In fact, our approach is based on the private business process, where messages exchanged with other participants, or business partners, can and shall be highlighted, but without going into detail on the private processes of other participants involved.

To define a persistent data model one needs to identify the domain entities, their attributes, and the relationships ($(1:n)$, $(m:n)$ or $(1:1)$) between entities [3].

A. The proposed approach

In [14], Borger claims that a notion of state is missing in BPMN, and consequently the specification of data dependence conditions is poorly supported. To overcome this fact, we opted by involving BPMN participants in the process. A BPMN participant is always related with all activities where he/she participates and, consequently, with the data stores manipulated by these activities.

Our approach is based on the following considerations:

- The information about the participants in the process is relevant to the process, especially for its control,

so all participants involved in data exchanges shall be represented in the data model.
- The pool representing the organization (company, department, etc.) that is designing the process shall disclose all internal roles involved. The other pools, as they represent entities outside the organization, do not have that need. It is only needed to show the input and output flow of information, e.g. through messages.
- Since a data store represents the persistent data, all data stores involved in the process shall be represented in the data model.
- If one wants to focus on different states in the same data store, the data store name, similarly to what happens with the data object, shall be given by $< DataStoreName >$ [$< DataStoreReferenceState >$].
- If there is involvement of data within a sub-process, the data flow shall be viewed with the subprocess extended.
- Data objects may represent electronic data as well as physical data. However, in both cases data must be stored. For example, if a data object represents the arrival of a shipment, the data about the shipment must be stored.

Our approach is organized into a set of three groups of rules. Each group is devoted to a particular goal: the first group (R1) identifies the data model entities, the second group (R2) identifies the relationships between entities, and in the third group (R3) the entities' attributes are identified.

The first group of rules is explained below:

- R1.1: Each data store, identified by a name, must be represented by an entity in the data model. The entity name is the name of the data store.
- R1.2: When two data stores have the same name, it is considered that they represent the same data store, so it will be represented by the same entity in the data model.
- R1.3: A role played by a participant (represented by a Lane or a Pool) must be represented by an entity in the data model: - If the pool is divided into several lanes, each lane will be represented by a data model entity. The name of each entity will be the name of the corresponding lane; - If the pool is not divided, the pool will result in an entity. The entity name will be the Pool name.
- R1.4: If a participant has the same name as a data store it will be represented by the same entity in the data model.

The second group of rules is explained below:

- R2.1: When a participant is responsible for an activity that manipulates a data store, the entity that represents the participant must be related with the entity that represents the data store. Each participant can perform the same activity several times, so the relationship between the entity that represents the participant and the entity that represents the data store, by default, will be $(1 : n)$.
- R2.2: If, in R2.1, the activity that handles the data store is cyclical (denoted by a circle arrow), or "Multi-instantiable" (denoted by three bars), i.e., it may be repeated several times within the process instance, then the relationship between the entity that represents the participant and the entity that represents the data store is $(m : n)$.
- R2.3: If the activity that handles the data store sends or receives information to/from another participant, this means that the participant provides or uses information from that data store. Therefore, there must be an "indirect" relationship between the entity that represents this participant and the entity that represents the data store. By default, the relationship type will be $(1 : n)$.
- R2.4: If, in R2.3, the activity fulfills the conditions presented in R2.2, the relationship type will be $(m : n)$.

Two rules have been created to identify the entities attributes. One to identify the attributes of the entities that represent the participants and the other to identify the attributes of the entities that represent the data stores:

- R3.1: A data store is an "Item-Aware element", so, as mentioned before, the data structure definition of these elements could be specified as a XML file. The definition of the structure will be used to identify each item that belongs to the data store. Each item identifies an attribute of the entity that represents the data store.
- R3.2: In Swimlanes only the name is identified. Consequently, the attributes of an entity that represents a role played by a participant are static: *ID* and *Name*. The ID represents the participant identification (code number). It could be the number of the employee, the code of business partner, etc. The Name represents the participant name, for example the name of the employee.

In the presented approach, the data stores and the roles played by participants give origin to entities in the data model. The relationship between the identified entities is deduced from the information exchanged between participants and the activities that manipulate the data store in two ways: directly by the participant that performs the activity and indirectly by the participant that sends or receives information to/from the activity that operates the data store.

In the next two subsections, two examples demonstrating the application of the proposed approach are described.

B. Doctor's Office example

The diagram shown in Figure 2 represents a process of appointment scheduling and attendance at a doctor's office. The diagram, adapted from [8], was complemented in order to highlight the data. Based on this diagram, the following entities can be identified: Receptionist, Patient and Doctor originated from participants (R1.3) and Appt, Symptoms and Prescription originated from data stores (R1.1 and R1.2).

The relationship between the entities is described next:

- The *Receptionist* reads and writes the data store *Appt*, so by R2.1, the relationship type is $(1 : n)$. A receptionist can make several appointments. An appointment is made by one receptionist.
- The *Doctor* participant handles *Symptoms*, *Prescription* and *Appt* data stores, so there is a relationship between the entity Doctor and each one of the entities Symptoms,

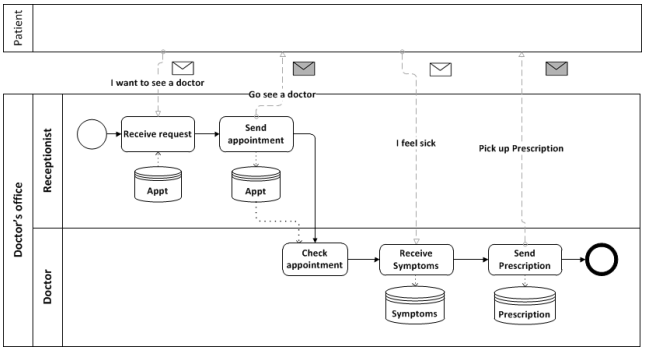

Fig. 2. Business process diagram focused on data

Prescription and Appt. Since a *Doctor* can perform the same activities several times, by R2.1, the relationship between Doctor and Symptoms entities is $(1:n)$ as well as between Doctor and Prescription and between Doctor and Appt.

- The *Patient* participant has an "indirect relationship" with the *Appt* and *Symptoms* data stores, since the activities that manipulate these data stores are activated by messages sent by the *Patient*. Once a patient can perform the same activities several times, by R2.3, the relationship between the entity Patient and the entity Appt is $(1:n)$. For the same reason, the relationship between the entities Patient and Symptoms is $(1:n)$.
- The *Patient* participant receives a message as a result of the activity that manipulates the data store *Prescription*. So, by R2.3, the relationship between the entity Patient and the entity Prescription is $(1:n)$.

Figure 3 shows the data model resulting from the application of our approach to the diagram shown in Figure 2.

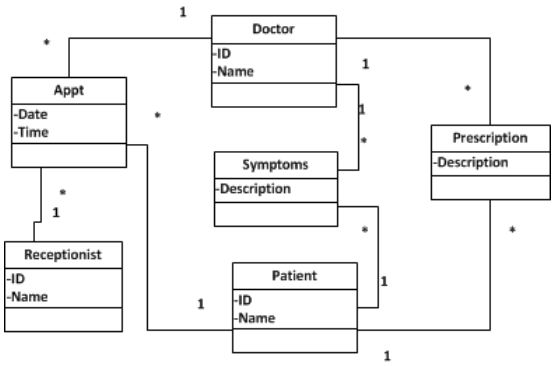

Fig. 3. Doctor's Office Data Model

By R3.2, the entities Receptionist, Patient and Doctor will have the same attributes: ID and Name; By R3.1, the attributes of the entities appt, Symptoms and Prescription are detailed in a XML file.

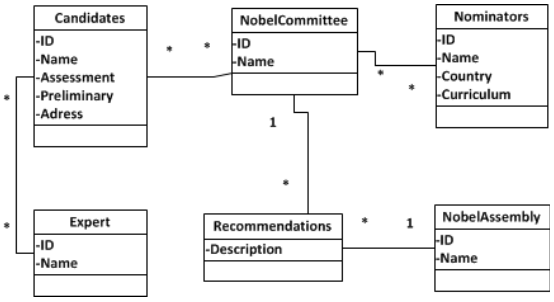

Fig. 5. Nobel Prize Data Model

C. Nobel Prize example

The diagram shown in Figure 4, represents the Nobel Prize in Medicine Process Diagram. As we can see in Figure 4, there are four roles, or participants, involved (Nobelcommittee, NobelAssembly, Nominators and Experts) and three different data stores (Nominators, Candidates and Recommendations). By R1.3, each participant will be represented by an entity in the data model. By R1.1, each data store will be represented as an entity in the data model. But, the *Nominators* participant has the same name as the *Nominators* data store, so, by R1.4, both are represented by the same entity in the data model.

Figure 5 shows the data model resulting from the application of the aproach to the diagram shown in Figure 4.

The relationship between the entities is described next:

- The activities that manipulates the *Nominators* data store are cyclic, so by R2.2, the relationship between *Nobelcommittee* and *Nominators* is $(m:n)$. The same happens between the entities *Nobelcommittee* and *Candidates*.
- By R2.1, the relationship between the entities *Nobelcommittee* and *Recommendations* is $(1:n)$.
- The activities that manipulates the *Candidates* data store are cyclic and exchange information with the *Expert* participant, so by R2.4, the relationship between the entities Candidates and Expert is $(m:n)$.
- By R2.3, the relationship between the entities *NobelAssembly* and *Recommendations*, is $(1:n)$.

By R3.1, the attributes of the entities Nominators, Candidates and Recommendations are detailed in a XML format file. By R3.2, the entities Nobelcommittee, NobelAssembly, Nominators and Experts will have the same attributes: ID and Name. Since *Nominators* is a participant and a data store, its attributes will be the joint of all.

V. Conclusions and Future Work

The growing organizations' interest in BPM has been accompanied by the increasing number of theories, modeling languages and supporting software applications in this area. Among the various tools and languages for business process modeling, BPMN has been used here because it is a well-known OMG standard and it is effectively used in business organizations [16], [4].

From BPMN 1.2 to BPMN 2.0 one can note a growing concern with data, including the possibility of identifying data

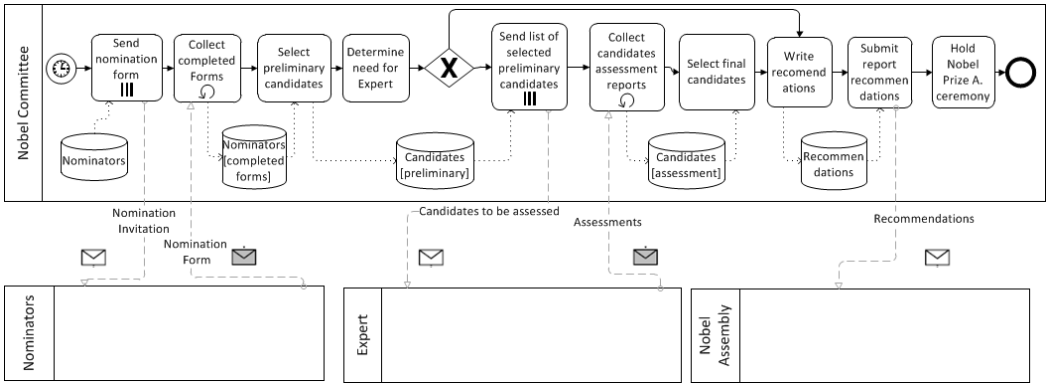

Fig. 4. The Nobel Prize Process Diagram (adapted from [15])

persistence. In BPMN 2.0 data persistence can be identified through the use of a data store. Thus, it becomes possible to identify data that are maintained in a persistent manner. However, to enable the ulterior obtention of the data model, it is necessary that the business process modeling is made taking data into account, i.e. the modeler must monitor the data throughout the process. Moreover, it is necessary to identify the activities that write or make use of the information stored in the data store, and ensure that the roles responsible for performing those activities are identified.

Thus, this paper presents an approach to model data involved in a business process from a private BPMN process diagram. In the proposed approach, data stores and process participants in the business model originate entities in the data model. The relationship between those entities can be deduced from the information exchange between participants and the activities that manipulate the data stores. Nonetheless, there may be relationships between the entities that represent the various data stores whose identification isn't possible to achieve here. For this reason, we can say that this approach only partially identifies the relationship between entities. On the other hand, information about the participants is scarce, leading to all entities that are obtained from the participants to have the same attributes. For the mentioned reasons, we may conclude that the proposed approach allows the identification of an early data model that may serve as a basis for further development.

Typically, in a real situation, a software product does not support only one process, but rather a set of processes. So, in order to generate a data model useful for the development of such software product, it will be necessary to assemble all data models resulting from the application of the proposed approach to each process being supported. One way to solve this problem is to group on the same entity all entities with the same name, gathering its attributes and maintaining the relationships.

An alternative way is to use a use case driven method. For that, a method like 4SRS (4-step rule set) [17] may be used. The 4SRS method generates a logical architecture and corresponding class diagrams [18] from user requirements, represented as use cases. It employs successive transformations of the software architecture in order to satisfy the elicited user requirements. In order to obtain a data model from a set of business processes, the 4SRS needs to be extended so that the use cases can be derived from the BPMN.

REFERENCES

[1] W. van der Aalst, "Business process management demystified: A tutorial on models, systems and standards for workflow management," vol. 3098 of LNCS, pp. 21–58, Springer, 2004.

[2] R. K. L. Ko, "A computer scientist's introductory guide to business process management (bpm)," *Crossroads*, vol. 15, pp.11-18, June 2009.

[3] M. Weske, *Business Process Management Concepts, Languages, Architectures*. Springer, 2010.

[4] A. Meyer, "Data in business process modeling," in *5th PhD Retreat of the HPI Research School on Service-oriented Systems Engineering*, 2010.

[5] P. Jalote, *A concise Introduction to Software Engineering*. Springer, 2008.

[6] H. Mili, G. B. Jaoude, ric Lefebvre, G. Tremblay, and A. Petrenko, "Business process modeling languages: Sorting through the alphabet soup," in *OOF 22 NO. IST-FP6-508794 (PROTOCURE II)*, 2003.

[7] D. M. Matteo Magnani, "Bpdmn: A conservative extension of bpmn with enhanced data representation capabilities," in *CoRR*, 2009.

[8] OMG, "Business process model and notation (bpmn), version 2.0," tech. rep., OMG, 2011.

[9] A. Sturm, "Enabling off-line business process analysis: A transformation-based approach," in *BPMDS*, 2008.

[10] M. Brambilla, J. C. Preciado, M. Linaje, and F. Sanchez-Figueroa, "Business process-based conceptual design of rich internet applications," *Web Engineering, International Conference on*, vol. 0, pp. 155–161, 2008.

[11] P. Wohed, W. van der Aalst, M. Dumas, A. ter Hofstede, and N. Russell, "On the suitability of bpmn for business process modelling," in *Business Process Management*, vol. 4102 of *LNCS*, pp. 161–176, Springer, 2006.

[12] T. Allweyer, *BPMN 2.0 - Introduction to the standard for business process Modeling*. Books on Demand GmbH, Norderstedt, 2010.

[13] B. List and B. Korherr, "An evaluation of conceptual business process modelling languages," in *Proceedings of the 2006 ACM symposium on Applied computing*, SAC06, pp. 1532–1539, ACM, 2006.

[14] E. Borger, "Approaches to modeling business processes: a critical analysis of bpmn, workflow patterns yawl," in *Software and Systems Modeling - Springer*, 2011.

[15] OMG, "BPMN 2.0 by example," tech. rep., OMG, 2010.

[16] M. Muehlen and J. Recker, "How much language is enough? theoretical and practical use of the business process modeling notation," in *AISE*, vol. 5074 of *LNCS*, pp. 465–479, Springer, 2008.

[17] R. Machado, J. Fernandes, P. Monteiro, and H. Rodrigues, "Refinement of software architectures by recursive model transformations," in *Product-Focused Software Process Improvement*, vol. 4034 of *LNCS*, pp. 422–428, Springer, 2006.

[18] M. Y. Santos and R. J. Machado, "On the derivation of class diagrams from use cases and logical software architectures," in *2010 Fifth International Conference on Software Engineering Advances*, 2010.