Investigating Completeness of Coding in Business Process Model and Notation

Carlos Habekost dos Santos[1], Lucinéia Heloisa Thom[1] and Marcelo Fantinato[2]

[1]*Department of Informatics, Federal University of Rio Grande do Sul, UFRGS, Porto Alegre, Brazil*

[2]*School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, Brazil*

Keywords: Business Process Management, Process Implementation, Business Process Model and Notation.

Abstract: One of the ways to represent a business process graphically is using the Business Process Model and Notation (BPMN). One of the things defined by the BPMN specification is a textual rule and a correspondent XML Schema for each notational element. However, there are some limitations regarding textual rules of notational elements and their XML Schema. For example, the XML Schema of end event element do not have any control to not connect any element after it, which can lead to a modeling issue. This paper introduces an approach to increment the XML Schema in a set of notational elements. The approach considers the BPMN textual rules and compares with the current XML schema, proposed by BPMN. To evaluate the approach, we will develop a prototype, to verify the completeness of the developed XML Schema allows better understanding compared with the current schema and will use mathematical formalism to verify the correctness of this new schema. We expect that our approach facilitate the understanding of business process by users and minimize possible implementation problems (e.g. deadlocks, lack of synchronization, livelocks, etc). Altogether, the results of this research can be interesting for users who want develop the BPM tools.

1 INTRODUCTION

Independently of the application domain, the activities of an organization are related to business processes such as buying products, participating in a public tender, manufacturing automobiles, or treating of patients.

The Business Process Management (BPM) allows the representation and management of its activities. In addition, it provides a set of techniques for the analysis and continuous improvement of organizations' business process (Weske, 2012), (Dumas et al., 2013) and (WfMC, 1999).

The life cycle of BPM includes the phases of modeling, configuration, implementation, executing and validation of business process (Weber et al., 2009), (Dumas et al., 2012). In particular, BPMN helps in a better documentation and standardization of business processes, as well as to increase efficiency and quality in execution (Thom, 2012).

In the modeling phase, business processes can be represented graphically by the Business Process Modeling and Notation (BPMN). BPMN provides a very large set of notational elements such as activities, events, gateways, etc. Besides BPMN, there are other notations that also allow process modeling such as Event Process Chain (EPC), Activity Diagram of the Unified Modeling Language (UML) and Petri Nets. However, BPMN is a standard for process modeling, adopted by the Object Management Group (OMG) (BPM, 2013), (Aalst, 2013), (Thom et al., 2009) and used by the majority of vendor's environment. For this reason, BPMN is the object of study in this paper.

The notational elements of BPMN are encoded in the XML Schema. It is also known as XSD (XML Schema Definition), the official recommendation of the World Wide Web Consortium (W3C) for XML document validation, structure definition, content and semantics [1].

For each notational element, BPMN specification offers a textual rule, a graphical representation and its XML Schema (See Figure 1). However, there is a lack of completeness between the textual rule and the corresponding XML Schema of notational elements. For example, in BPMN specification there is a textual rule defining which in the message flow element must connect elements from different pools, but its

[1]More information available in: http://www.w3.org/XML/Schema.html

XMLSchema do not represent this textual rule. Other examples:

- In end event element can not be placed any outgoing sequence flow.
- In loop activity, executes the inner activity as long as the loop condition evaluates as true.
- In parallel gateway, this wait for all incoming flow before triggering the flow through its outgoing sequence flows.

In these examples, all XML Schema do not have any code to represent such textual rules. Only there are basic information about the notational element such as name, type, etc.

In BPMN specification, there exists five categories of notational elements: flow objects, data, connecting elements, swimlanes and artifacts. We investigate the elements of these categories, identifying limitations regarding textual rules and respective XML Schema of the notational elements. There are, depending on the notational element, modeling problems such as deadlock, livelock or lack of synchronization as described below:

- *Lack of Synchronization*: happens when parallel flows, which are started with a gateway dividing the flow, do not converge to another gateway which join the flow.
- *Deadlock*: occurs when it is not possible to perform any activity due to a crash in a point of the process.
- *Livelock*: when the process enters an infinite loop, recurrently performing the same activities.

In this context, the main goal of this paper is to report an on-going research that aims to increase the XML Schema completeness, proposed by the BPMN specification, in order to have more adherence to the textual rules described for each notational element. The work occurs based on the analysis of these rules, identifying and implementing those that are not included in the current XML Schema. In addition, the research also aims to create XML Schema variants, containing the textual rules already defined by the BPMN specification (BPM, 2013), and modeling rules suggested by the literature, such as those proposed in (Mendling et al., 2010).

We believe that with our approach users will have a better understanding of the XML Schema of notational elements, from the most complete codification of BPMN.

For the development of the XML Schema completeness of notational elements, we used this methodology:

- Selection of notational elements: definition of a set of notational elements that will have the XML Schema incremented.
- Validation Perspective: definition of validation perspectives. We plan to do a prototype for users interact with the XML code generate from XML Schema, through forms that show a textual rule, the original code and the incremented code. The users will identify the code best represent the textual rule.

The remainder of this paper is structured as follows. Section 2 provides the state of art in business process modeling and implementation. Section 3 shows the proposed approach for the XML Schema completeness and perspectives for validation. Finally, Section 4 concludes the paper.

2 STATE OF ART

To obtain the state of the art related to modeling and implementation of business process we conducted a systematic literature review using (Kitchenham and Charters, 2007). A systematic literature review is a means to identify, assess and interpret research papers available and relevant to a specific research question, a thematic area, or a phenomenon of interest (Kitchenham et al., 2010).

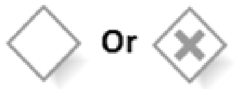
Our aim with the systematic review was to gather a better understanding of modeling and implementation so that we could identify areas for improvement quote (Santos and Thom, 2014).

The systematic literature review occurred in the first semester of 2014. We analyzed the title, keywords and abstract of works considering the period from 2009 until 2014. Our approach focused on the digital libraries:

- ACM Digital Library;
- IEEE Xplore Digital Library;
- Springer Link;
- Google Scholar;
- Author's page with related publications to BPM area and h-index relevant.

To select the works, we considered "string search" to get it. Terms with quotation marks (" and ") indicates these should be considered the complete term:

1. "business process" AND modeling AND execution;
2. "business process execution" OR "workflow execution";

Textual Rule	Each *token* arriving at any incoming Sequence Flows activates the gateway and is routed to exactly one of the outgoing Sequence Flows. In order to determine the outgoing Sequence Flows that receives the *token*, the conditions are evaluated in order. The first condition that evaluates to true determines the Sequence Flow the *token* is sent to. No more conditions are henceforth evaluated.
Graphical Representation	

XML Schema	`<xsd:element name="exclusiveGateway" type="tExclusiveGateway" substitutionGroup="flowElement"/>`
	`<xsd:complexType name="tExclusiveGateway">`
	`    <xsd:complexContent>`
	`        <xsd:extension base="tGateway">`
	`            <xsd:attribute name="default" type="xsd:IDREF" use="optional"/>`
	`        </xsd:extension>`
	`    </xsd:complexContent>`
	`</xsd:complexType>`

Figure 1: A example modeling rule defined by BPMN.

3. ("business process" OR workflow) AND (model OR modeling OR design) AND (implementation OR implemented OR execution OR executable OR enactment);

From the string search in the digital libraries, adding search in author's BPM page, it reached a set of 350 works.

We performed the analysis with the title and abstract of each article. As result, we selected 30 candidate articles, which the approach was related to the string search. Through the analysis of these articles, 19 were selected as best works to represent the string search. This article, we select the works which represents the results of systematic literature review.

The works studied started on the BPMN specification (BPM, 2013). For each notational element, there exists a textual rule, the graphical representation and the XML Schema. Figure 1 illustrates an example of this definition in the BPMN specification. The described example shows the exclusive gateway (XOR) element used for routing the process flow.

From this example, it is possible to note that there is low consistency between the default XML Schema of BPMN and the textual rule. There are not some rules allowed in the XSD, for example, the evaluation of only a condition which have no corresponding code in the "XML Schema". Furthermore, BPMN defines rules indicating which elements can be connected by a sequence flow (e.g. a gateway can not connect to a start event), however, there is no XSD solution to allow this control.

In (Mendling et al., 2010), was presented a research on the relationships between structured models and their error probability and understanding. As a result, seven modeling rules were proposed to guide users to avoid problems of modeling and understanding business processes models.

In the approach of (Stroppi et al., 2011) was proposed an extension of the BPMN notation called BPMN + X. This work extends the XSD encoding of notational elements. Also it discusses about the lack of guidelines and methodologies to enable the extension, by the BPMN specification.

In (Dijkman et al., 2008) is proposed an analysis of business process models, which are mapped to petri nets in order to identify possible problems such as deadlock and livelock with the help of some techniques.

Altogether, the studies analyzed aim to verify issues related to modeling problems. Currently, these works focus on the use of external techniques such as petri nets to solve these problems, rather than focusing on improvements directly in the encoding of the BPMN specification.

3 COMPLETENESS OF XML SCHEMA

In this section we discuss the XML Schema completeness of notational elements. Initially, we studied the structure of the XML Schema available in the BPMN specification version 2.0 (BPM, 2013). We proposed, then to increase the completeness of the XML Schema concerns by adapting this schema to represent conditions and other rules that are missing in the original XML Schema. The adaption follows three phases:

1. Definition of the group of notational elements to be incremented.
2. XML Schema increment to make it more complete regarding the textual rules inherent to each notational element being studied.
3. Definition of possible perspectives of how to validate the XML Schema incremented.

First, we defined the group of notational elements that will be incremented. For this selection, we considered only the basic set of elements defined in the BPMN specification: events, activities and gateways. In this paper, we present the proposal of the exclusive gateway. However, we are also investigating the inclusive gateway (OR) and parallel gateway (AND). We started the investigation on these notational elements because there are studies in the literature that highlight them as the most problematic in terms of user understandability (Kossak et al., 2012), (Figl et al., 2013), (Dehnert and Aalst, 2004) and modeling (Mendling et al., 2010).

Next, we modified the current XML Schema, adding new code to it. We developed a new XML Schema based on the analysis of the textual rules described in the Chapter 13 of the BPMN specification which describes the semantics of the notational elements (BPM, 2013). The change occurs in the XML Schema of the notational element. In (Stroppi et al., 2011) it is proposed a way to modify the XML schema to extend the notational elements. The proposed completed XML Schema is illustrated in Figure 2.

Figure 2a shows the XML Schema suggested for the XOR element (see "Extension code XSD"). The completeness is achieved by adding the <xsd:element> tag. Figure 2b shows the corresponding XML code generated.

The code added, in this example, refers to the textual rule defined in Figure 1 (see "Textual Rule"). This rules says that only one condition evaluates to true. The branch corresponding to this condition receives the token and no other condition is tested. If no condition evaluates to true, the token is passed to the default sequence flow. In case there is no default flow, an exception is generated. The logic used to develop the code shown in "XML code" is (See Figure 2b):

- While the counter true condition is zero (*trueConditions == 0*):
  - If the tested condition is true (*condition == true*), then the counter true conditions is incremented (*trueConditions = 1*), otherwise it verifies if all conditions were tested (*allConditions == 0*).
  - If all conditions were tested, then it checks if there is a default condition (*conditionDefault*). If so, then the counter true conditions is incremented (*trueConditions = 1*), otherwise an exception is generated informing that the conditions are not valid.

The number of iterations of the algorithm varies according to the number of conditions.

The variables used in the example, *trueConditions*, *condition*, *allConditions* and *conditionDefault*, were added exclusively for this example. In practice, the tool that implements this XML Schema must also implements their conditions so that the operation occurs as expected.

In addition to the XML Schema completeness, we plan to provide alternative XML Schema based, for example, on the seven rules and standardization of the labels format defined by (Mendling et al., 2010). The goal of alternative XML schema is to provide suggestions to the user who wants to avoid potential problems with deadlock, livelock, or enable a correct modeling of the elements labels.

3.1 Perspectives of Validation

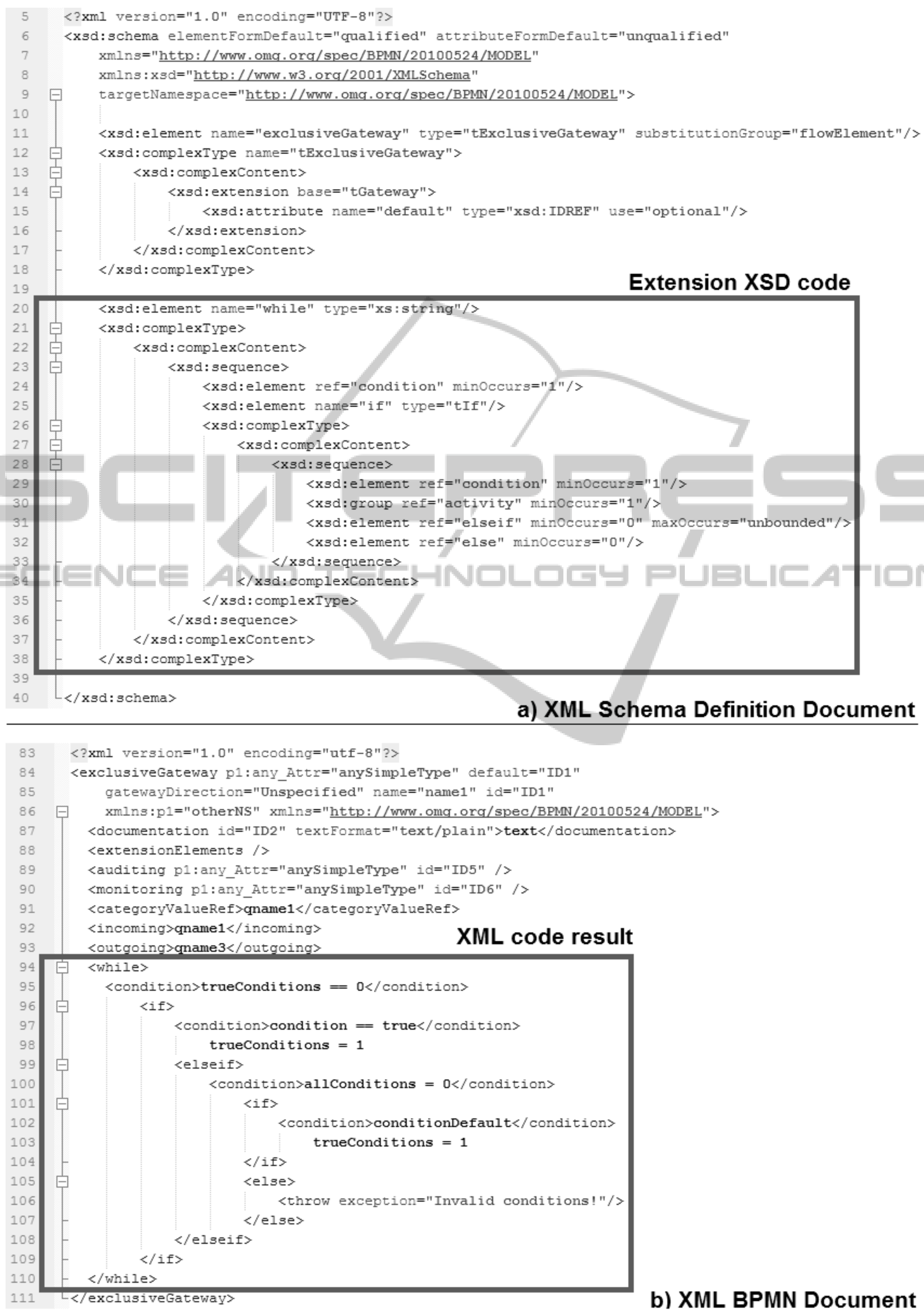
In this section, we describe the case study designed to validate the proposed XML Schema completeness. For this purpose, a prototype in which users can interact with the XML code generate from this schema, will be developed. This will occur through forms that show the textual rule, the original XML code of BPMN specification and the incremented XML code. The purpose of this case study is to identify, through the users, which XML Schema best represents the textual encoding rule.

The forms will be designed using Google Forms[2]. The target audience is students and people from the industry, who have some knowledge in programming logic, but not necessarily in XML. The idiom for development will be Portuguese and English.

We decided to use this type of case study because it can helps to identify if the the users can have a better understanding of XML Schema, and if have more adherent to the textual rules. Therefore, the goal with the validation is to compares the current XML Schema and the improved XML Schema that we are proposing, to identify which one best applies the notational elements textual rules.

Another way to validate the work, refers to the use of mathematical formalism to verify the correctness

[2]http://www.google.com/forms/about/



```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://www.omg.org/spec/BPMN/20100524/MODEL"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.omg.org/spec/BPMN/20100524/MODEL">

    <xsd:element name="exclusiveGateway" type="tExclusiveGateway" substitutionGroup="flowElement"/>
    <xsd:complexType name="tExclusiveGateway">
        <xsd:complexContent>
            <xsd:extension base="tGateway">
                <xsd:attribute name="default" type="xsd:IDREF" use="optional"/>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <xsd:element name="while" type="xs:string"/>
    <xsd:complexType>
        <xsd:complexContent>
            <xsd:sequence>
                <xsd:element ref="condition" minOccurs="1"/>
                <xsd:element name="if" type="tIf"/>
                <xsd:complexType>
                    <xsd:complexContent>
                        <xsd:sequence>
                            <xsd:element ref="condition" minOccurs="1"/>
                            <xsd:group ref="activity" minOccurs="1"/>
                            <xsd:element ref="elseif" minOccurs="0" maxOccurs="unbounded"/>
                            <xsd:element ref="else" minOccurs="0"/>
                        </xsd:sequence>
                    </xsd:complexContent>
                </xsd:complexType>
            </xsd:sequence>
        </xsd:complexContent>
    </xsd:complexType>

</xsd:schema>
```

```
<?xml version="1.0" encoding="utf-8"?>
<exclusiveGateway p1:any_Attr="anySimpleType" default="ID1"
    gatewayDirection="Unspecified" name="name1" id="ID1"
    xmlns:p1="otherNS" xmlns="http://www.omg.org/spec/BPMN/20100524/MODEL">
  <documentation id="ID2" textFormat="text/plain">text</documentation>
  <extensionElements />
  <auditing p1:any_Attr="anySimpleType" id="ID5" />
  <monitoring p1:any_Attr="anySimpleType" id="ID6" />
  <categoryValueRef>qname1</categoryValueRef>
  <incoming>qname1</incoming>
  <outgoing>qname3</outgoing>
  <while>
    <condition>trueConditions == 0</condition>
        <if>
            <condition>condition == true</condition>
                trueConditions = 1
            <elseif>
                <condition>allConditions = 0</condition>
                    <if>
                        <condition>conditionDefault</condition>
                            trueConditions = 1
                    </if>
                    <else>
                        <throw exception="Invalid conditions!"/>
                    </else>
            </elseif>
        </if>
  </while>
</exclusiveGateway>
```

Figure 2: Proposal for completeness of BPMN encoding.

of the proposed XML Schema. The aim of this validation is to verify to the reliability and robustness of the proposed schema. The formalism to be applied is being analyzed.

4 CONCLUSIONS

This paper presented an ongoing work regarding the completeness XML Schema of notational elements. The current XML Schema proposed in the BPMN specification does not cover all the defined textual rules, as a result each vendor's environment implements it in a different way.

Though the BPMN notation should not have restrictions on the XML Schema of notational elements to implement different scenarios of the business process models, the textual rules should be implemented, to reduce the possibility of problems to happen in modeling, such as deadlocks and livelocks.

The completeness XML Schema proposed allows to increase semantic in the schema of notational elements. Textual rules that could be only viewed in the BPMN specification, can now be also viewed in the XML Schema, so that users can have a more adherent schema.

As current work limitation refers to the fact that not all textual rules can be applied. For example, in Figure 1, "Textual Rule" defines that each token arriving in a gateway is sent to exactly one the gateway output streams. This textual rule can not be represented in XML Schema because there is not a tag that can express this rule.

As main contribution, the present work will provide a more adherent XML Schema from textual rules, adding more semantic of the notational elements. Users who plan to develop a new tool for modeling business processes will be using a more accurate XML Schema, reducing the possibility of potential problems.

As future work, we plan to complete the development of alternative XML Schema. After, we will develop a parsing, which enables a more detailed analysis of the coding notation, allowing a more effective scrutiny of possible failures of implementation.

Finally, we propose graphical representation of notational elements that facilitate user understanding of the business process, so that can represent when a gateway is fork or join, using different graphics. This work is focused on enabling the minimization of possible modeling problems.

ACKNOWLEDGEMENTS

The authors would like to thank Capes and CNPq, Brazil, for partially supporting this work.

REFERENCES

(2013). *Business Process Modeling Notation (BPMN), V. 2.0.2.* BPMN.

Aalst, W. M. P. v. d. (2013). Business Process Management: A Comprehensive Survey. *ISRN Software Engineering*, 2013:1–37.

Dehnert, J. and Aalst, W. M. P. v. d. (2004). Bridging the Gap between Business Models and Workflow Specifications. *Journal of Cooperative Information Systems*, pages 1–39.

Dijkman, R. M., Dumas, M., and Ouyang, C. (2008). Semantics and analysis of business process models in BPMN. *Information and Software Technology*, 50(12):1281–1294.

Dumas, M., Rosa, M. L., Jan, M., and Reijers, H. A. (2013). *Fundamentals of Business Process Management.* Springer-Verlag, Berlin, Germany, first edition.

Dumas, M., Rosa, M. L., and Mendling, J. (2012). Understanding business process models: the costs and benefits of structuredness. *Advanced Information ...*, pages 31–46.

Figl, K., Recker, J., and Mendling, J. (2013). A study on the effects of routing symbol design on process model comprehension. *Decision Support Systems*, 54(2):1104–1118.

Kitchenham, B. and Charters, S. (2007). Procedures for performing systematic reviews. Keele university. technical report tr/se-0401, Department of Computer Science, Keele University, UK.

Kitchenham, B., Pretorius, R., Budgen, D., Pearl Brereton, O., Turner, M., Niazi, M., and Linkman, S. (2010). Systematic literature reviews in software engineering – A tertiary study. *Information and Software Technology*, 52(8):792–805.

Kossak, F., Illibauer, C., and Geist, V. (2012). *Event-Based Gateways: Open Questions and Inconsistencies*, pages 53–67.

Mendling, J., Reijers, H. A., and Aalst, W. M. P. v. d. (2010). Seven process modeling guidelines (7PMG). *Information and Software Technology*, 52(2):127–136.

Santos, C. H. and Thom, L. H. (2014). Uma revisão sistemática sobre modelagem e execução de processos de negócio. Individual work, Institute of Informatics, Federal University of Rio Grande do Sul, BR.

Stroppi, L. J. R., Chiotti, O., and Villarreal, P. D. (2011). Extending BPMN 2.0: Method and Tool Support. *Business Process Model and ...*, pages 59–73.

Thom, L. H. (2012). Gerenciamento de Processos de Negócio e Aplicabilidade na Saúde e na Robótica. *lbd.dcc.ufmg.br*.

Thom, L. H., Reichert, M., and Iochpe, C. (2009). On the Support of Workflow Activity Patterns in Process Modeling Tools: Purpose and Requirements.

Weber, B., Sadiq, S., and Reichert, M. (2009). Beyond rigidity – dynamic process lifecycle support. *Computer Science - Research and Development*, 23(2):47–65.

Weske, M. (2012). *Business Process Management*. Springer Berlin Heidelberg, Berlin, Heidelberg.

WfMC (1999). Workflow Management Coalition Terminology & Glossary. (3):1–65.