An Analytical Evaluation of BPMN Using a Semiotic Quality Framework

Terje Wahl, Guttorm Sindre

Department of Computer and Information Science,
Norwegian University of Science and Technology,
Sem Sælands vei 7-9, N-7491 Trondheim, Norway
{terje.wahl, guttorm.sindre}@idi.ntnu.no

Abstract. Evaluation of modelling languages is important both to be able to select the most suitable languages according to the needs, and to improve existing languages. In this paper Business Process Modeling Notation (BPMN) is presented and analytically evaluated according to the Semiotic Quality Framework. BPMN is a functionally oriented language well suited for modeling within the domain of business processes, but probably also general processes not only within the business domain. The evaluation indicates that BPMN is easily learned for simple use, and Business Process Diagrams (BPDs) are relatively easy to understand. Tools may fairly easily map BPDs into the BPEL4WS format, but executable systems then require creation of Web Services representing the Activities in BPDs. An evaluation according to the BWW ontology is useful for finding ontological discrepancies, and the semiotic framework is useful for evaluating quality on a relatively general level. These methods thus complement each other.

1 Introduction

Currently there exist a large number of different modelling languages. Many of them define overlapping concepts and usage areas, and consequently it is difficult for organizations to select the most appropriate language related to their needs. Traditionally the research community have focused more on creating new modelling languages than evaluating existing ones. However, evaluation of languages is important both to be able to select the most suitable ones, and to improve existing languages.

Conceptual modelling languages can be evaluated analytically and empirically. As Gemino and Wand discuss in [9], analytical and empirical analyses of modelling techniques complement each other. We can also distinguish between analyses of single languages and comparative analyses of several languages. In this paper we present Business Process Modeling Notation (BPMN), and perform an analytical evaluation of the quality of BPMN according to the Semiotic Quality Framework [1] [2]. We also discuss how an analytical evaluation according to the BWW ontology may be performed as a complement to this evaluation.

Section 2 presents BPMN including its notation. It provides some examples of Business Process Diagrams (BPDs), and relates BPMN to the BPEL4WS standard. Section 3 presents the Semiotic framework for evaluation, divided into parts for evaluating the quality of conceptual models and the quality of conceptual modelling languages. Section 4 presents an analytical evaluation of BPMN according to the semiotic framework. Section 5 gives a short summary of what the BWW ontology is, how it may be used to evaluate conceptual modelling languages, and discusses in what ways this can complement the evaluation according to the semiotic framework. Section 6 mentions related work, and Section 7 presents future work. The conclusion is given in section 8.

2 BPMN

2.1 Overview

Business Process Modelling Notation (BPMN) is a notation aiming to be easily understandable and usable to both business users and technical system developers [5]. It also tries to be formal enough to be easily translated into executable code. By being adequately formally defined, it can create a connection between the design and the implementation of business processes.

BPMN defines Business Process Diagrams (BPD), which can be used to create graphical models especially useful for modelling business processes and their operations. It is based on a flowchart technique - models are networks of graphical objects (activities) with flow controls between them.

The BPMN 1.0 specification was developed by The Business Process Management Initiative (BPMI) and released in May 2004. BPMN is based on the revision of other notations and methodologies, especially UML Activity Diagram, UML EDOC Business Process, IDEF, ebXML BPSS, Activity-Decision Flow (ADF) Diagram, RosettaNet, LOVeM and Event-Process Chains (EPCs).

2.2 Basic Notation

The graphical elements that are defined by BPMN for use in BPDs are divided into a small number of categories, so that they can be easily recognized even if a user is not immediately familiar with a specific graphical element [5]. The four basic categories of elements are *Flow Objects*, *Connecting Objects*, *Swimlanes* and *Artefacts* [5]:

Flow Objects contain the three core elements used to create BPDs: *Event* (*Start*, *Intermediate* and *End*), *Activity* (atomic *Task* and compound *Sub-Process*) and *Gateway* (decision-making, forking and merging of paths).

Connecting Objects are used to connect Flow Objects to each other through arrows representing *Sequence Flow*, *Message Flow* and *Association*.

Swimlanes are used to group activities into separate categories for different functional capabilities or responsibilities (e.g. a role/participant). A *Pool* represents a Participant in a Process, and Pools can be divided into *Lanes* (e.g. between divisions in a company). Pools are used when a Process involves two or more business entities or participants. Activities within Pools must constitute self-contained Processes. Because of this, Sequence Flow may not cross from one Pool to another. Instead, Message Flow goes between Pools to indicate the communication between participants. See Section 2.4 for an example of this.

Artefacts may be added to a diagram where deemed appropriate. The following three Artefacts are defined: *Data Object*, *Group* and *Annotation* (to be used for comments and explanations).

For further introduction to BPMN, [5] is recommended.

2.3 Metamodel

A metamodel is defined for BPMN [6]. It contains 55 concepts, some attributes and many relations between the concepts. Because of its relative complexity, its further description is out of scope for this paper.

2.4 Examples

To be better able to understand what BPDs are, two examples are shown here. Fig. 1 below shows a simple process using Flow Objects, Connecting Objects and Annotations.

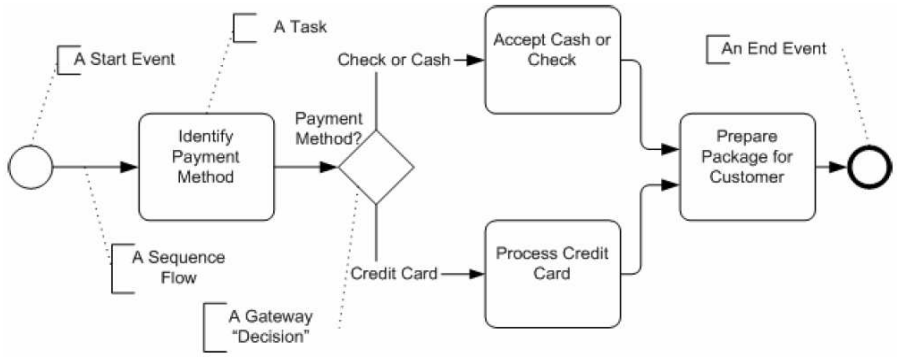

Fig. 1. A simple example of a business process, as shown in [5]

Note how Sequence Flow is used in Fig. 2 only within the Pools, and Message Flow is used for communication between the two Pools:

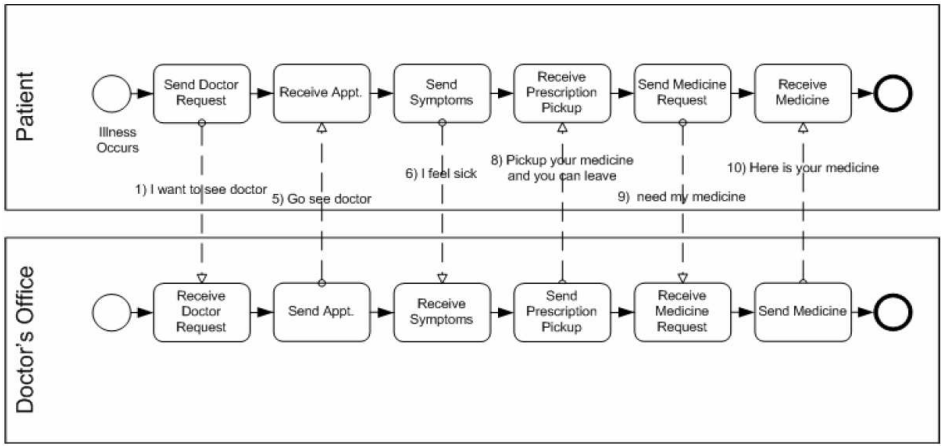

Fig. 2. An example of a BPD which uses two Pools, as shown in [5]

2.5 Relation to BPEL4WS

Business Process Execution Language for Web Services (BPEL4WS) is a standard for specifying business process behaviour based on Web Services [7]. Processes that are described by BPEL4WS export and import functionality exclusively by using Web Service interfaces, they are stored in a directly executable XML-format and rely on the use of Web Service Description Language (WSDL) and SOAP. BPMN was designed with easy translation into BPEL4WS in mind. Because of this, there are only a few terms in BPMN that cannot be translated into BPEL4WS, and vice versa. The BPMN Specification [8] even contains a section describing how to translate a BPD into BPEL4WS.

3 Semiotic Framework for Evaluation of Quality

In [2], Lindland, Sindre and Sølvberg present a semiotic framework for understanding and evaluating quality of conceptual models. In [1], this semiotic framework has been extended, and also includes a closely related framework for evaluating the quality of conceptual modelling languages. As an example, Krogstie has evaluated UML using this framework [3]. Krogstie's paper also gives a nice introduction to the semiotic framework.

The semiotic quality evaluation framework specifically distinguishes between goals and means, meaning that they separate what you want to achieve from how you achieve it [3]. The framework is based on linguistic and semiotic concepts (such as *syntax*, *semantics* and *pragmatics*) that enable the assertion of quality at different levels, as further described below. The semiotic framework is based on a constructivistic world-

view, meaning that it is recognized that there exists no "absolute truth" in the sense that every participant can always have one common objective agreement on one model. In stead, models are created through dialog as a compromise between the different world views that each participant has.

3.1 Quality of Conceptual Models

The main concepts of the semiotic framework are Model, Modeling domain, Language extension, Participant knowledge, Social actor interpretation and Technical actor interpretation [1]. The relationships between the concepts provide the different quality aspects of the framework. For example, *Syntactic Quality* is based on the relationship between the Model and the modelling language that is used (Language extension).

The seven different relationships represent different aspects of quality: *Physical Quality* regards the physical representation of the model and its externalization and internalization. *Empirical Quality* regards layout and a presentation that is easy to read and write without mistakes. *Syntactic Quality* is about the model being valid and complete with regards to the modeling language being used. *Semantic Quality* is about validity and completeness of the model in relation to the domain being modelled. *Perceived Semantic Quality* of a model is measured like semantic quality above, but in addition it depends on the actor's interpretation of the model and his/her knowledge of the domain. *Pragmatic Quality* regards the audience's comprehension of the model. *Social Quality* has the definition of actors having agreement (relative or absolute) about their interpretation, knowledge and model.

3.2 Quality of Conceptual Modelling Languages

The semiotic framework for evaluating the quality of conceptual modelling languages is based on the framework for quality of conceptual models [1]. It is used to evaluate the modeling language's *potential* for making models of high quality. According to [3], one can evaluate two kinds of criteria: Criteria for the conceptual basis of a language, e.g. the metamodel for the language, and criteria for the external (graphical) representation of the language. The metamodel for a conceptual modelling language can be regarded as a conceptual model in itself, and thus evaluated also according to the framework for quality of conceptual models. It may also be useful to evaluate the specification or other documentation of a language according to the semiotic quality framework.

Five aspects are identified for evaluating the quality of conceptual modelling languages: Domain appropriateness, participant language knowledge appropriateness, knowledge externalizability appropriateness, comprehensibility appropriateness and technical actor interpretation appropriateness.

The relationships in Fig. 3 below represent these five aspects of language quality:

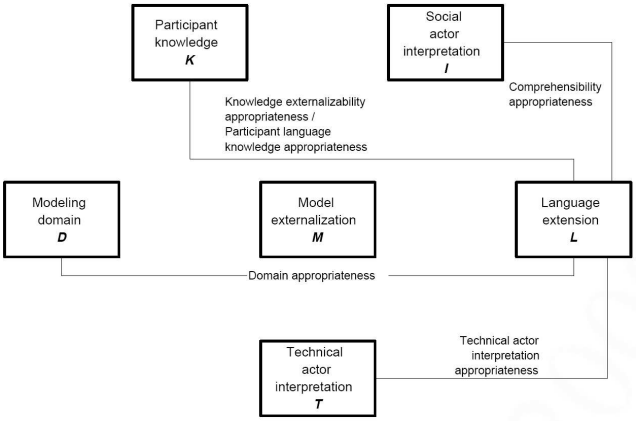

Fig. 3. The framework for quality of conceptual modeling languages, as presented in [1] and [3]

- *Domain Appropriateness*. This deals with how suitable a language is for use within different domains. If "there are no statements in the domain that cannot be expressed in the language" [1], then the language has good domain appropriateness.
- *Participant Language Knowledge Appropriateness*. It's a goal here that the participants know the language and are able to use it. They should have explicit knowledge about all the statements in the language-models of the languages they use [1].
- *Knowledge Externalizability Appropriateness*. This deals with the participants' ability to express all their relevant knowledge using the modeling language. A language has good knowledge externalizability appropriateness if "there are no statements in the explicit knowledge of the participant that can not be expressed in the language" [1].
- *Comprehensibility Appropriateness*. The audience should be able to understand as much as possible of the language. Good comprehensibility appropriateness is achieved if "all the possible statements of the language are understood by the participants in the modeling effort using the language" [1].
- *Technical Actor Interpretation Appropriateness*. It is important for technical actors that the language is suitable for automatic reasoning. This can be achieved if the language is relatively formally defined and reasoning is efficient and practical to use, e.g. by being executable.

4 Analytical Evaluation of BPMN

4.1 Domain Appropriateness

The most central concept in BPMN is the *process*, which is built up from activities.

Because of this, the main *perspective* (as defined in [1]) of BPMN is the *functional perspective*. Data Flow Diagrams (DFD) and UML Activity Diagrams are examples of other conceptual modeling languages with a functional perspective. BPMN is well suited to model processes consisting of activities, with simple and advanced rules for the flow of the sequence of activities. BPDs (that are created using BPMN) can also show which actors or roles perform these activities by using Swimlanes.

But because of its functional perspective, BPMN has clear limitations to its domain appropriateness. It is not well suited for expressing for example models in the object-oriented domain. BPMN lacks concepts like class hierarchies. As stated in [8], BPMN is *not* suitable for modelling organizational structures and resources, functional breakdowns, data and information models, strategy or business rules.

BPMN was created for the main purpose of modelling *business* processes, and is hence well suited for modelling the business domain (e.g. B2B processes). However, the BPMN 1.0 Specification [8] and the BPMN Metamodel [6] do not explicitly limit the usage of the language to business processes. The constructs of the language do not contain any business-specific terms. Because of this, advanced processes can be modelled even if they are not business related. However, BPMN was constructed to support *only* the concepts of modelling that are relevant for business processes [6]. Because of this, some important concepts regarding the specification of processes within other domains are missing from the BPMN language. As an example, BPMN contains no constructs representing valves or pumps for modelling control engineering processes. Those needing to model processes in other domains will in many cases prefer and benefit from using other more domain specific languages. The BPMN specification [8] does however provide possibilities of extending the language to support modelling of different vertical domains, but it is unclear how and to what extent this may be done since the specification is unclear on this point.

4.2 Participant Language Knowledge Appropriateness

The graphical elements of BPMN are defined in a clear and concise way, to avoid confusions and ease the learning of the language. The language is also made to have similar notation to other languages like Flowcharts, UML Activity Diagrams, Event Process Chains, Petri Nets and Data Flow Diagrams (DFD). For example, a diamond shape is used both in BPMN, UML Activity Diagrams and Flowcharts to express a decision point, and both in BPMN and Activity Diagrams to express a merge. BPMN also have a striking resemblance to Activity Diagrams regarding the notational representation of events (small circles) and activities (rounded rectangles). In addition, the concept of Swim Lanes and its graphical representation are very similar in Activity Diagrams. Sequence flows are represented by arrows with solid lines and solid arrowheads in BPMN, Activity Diagrams, Flowcharts and Petri Nets. These similarities are helpful, at lest for IT professionals that are already familiar with the other languages. There are however also some graphical elements that are used differently in BPMN compared to other languages. For example, BPMN use a diamond shape also to represent forks or joins

(of parallel activities), but in Activity Diagrams a thick horizontal line is used for this. Flowcharts use rounded rectangles to represent start- or end-states, and not to symbolize activities. These differences make it more difficult to learn BPMN.

It is a goal for BPMN that it should be understandable not only by IT professionals, but also for business analysts and other non-technical people [8]. However, due to the complexity of the more advanced aspects of BPMN, these authors find it a bit unrealistic that normal business users without training should be able to understand advanced business processes modelled using BPMN. As an example of the complexity of BPMN, there are 23 different predefined diagram elements representing different types of events.

4.3 Knowledge Externalizability Appropriateness

This area is highly dependable on the specific knowledge of the actors that are using the language, and is therefore difficult to evaluate in a general way. We can however make assumptions about the typical participants involved in the modelling process. BPMN probably appeals the most to business users, since it was created especially for modelling business processes. The term "business users" is however very broad and include a wide variety of actors. If the actors want to model a process purely within the business domain, BPMN has very good support for this. But if they desire to create models involving other domains as well, this may be difficult (ref. subsection 4.1 above) and supplements to the BPMN-models may be needed. Thus it may be hard for actors to externalize their relevant knowledge using only Business Process Diagrams (BPDs) if that knowledge goes beyond the domain of business processes. As already mentioned, the BPMN specification [8] provide possibilities of extending the language to better support modelling of several vertical domains, but it is unclear how and to what extent this may be done.

4.4 Comprehensibility Appropriateness

Comprehensibility of a conceptual modelling language can be divided into understanding of language concepts and understanding of notation. Regarding notation, BPMN provides a small number of notational categories so that the readers can easily recognize the basic types of elements that constitute the diagrams [5]. In addition, these basic categories contain variations that may be used when creating more complex BPDs. This categorization helps with the comprehensibility of BPDs. It also helps that the notational categories are easily distinguished from one another, and look partially familiar to other languages like UML Activity Diagrams. In some cases the notation is very intuitive, e.g. envelopes are used for symbolizing message events and clocks are used for symbolizing timer events. The BPMN specification [8] gives some helping guidelines on how to create clear and understandable diagrams. But on the other hand, it has few strict requirements on how to layout diagram elements and connect flow arrows between them, so the potential for creating BPDs with poor empirical quality (and thus worsened

comprehensibility) is present despite the guidelines. Regarding the concepts defined for BPMN, the authors think that the basic concepts used in the language is descriptive, accurate, easily understandable and well defined in the specification [8]. The more detailed and advanced concepts in BPMN will however require training to fully understand what they mean when used in connection with BPDs.

BPMN support aggregation by allowing activities that are "collapsed" and contain sub-activities. This helps to understand and get an overview over complex models.

4.5 Technical Actor Interpretation Appropriateness

BPDs are with a few exceptions easily mapped into the BPEL4WS-format. Guidelines for doing this can be found in the BPMN 1.0 Specification [8], and this relatively easy mapping helps the technical actors that want to implement a BPD into an executable Information System (IS). Mappings to other more formally defined languages are not defined, though it is possible to do this. But if the technical actors don't want to implement the models using BPEL4WS processes and Web Services, more work is probably required to convert the BPD into an executable IS.

BPEL4WS requires the use of WSDL and Web Services to be executable. Because of this, it is not so easy to do automated reasoning about processes that are not suitable to be implemented using a combination of Web Services.

Atomic Activities in BPDs are supposed to usually represent a Web Service. How to specifically implement these Web Services may be difficult to interpret, especially if the Activity is vaguely defined using only a short textual description.

4.6 Quality of the BPMN Language Model

The BPMN Metamodel (and its evaluation) is too complex for a detailed analysis in the scope of this paper. But its sheer size and complexity suggests that it might have a less than perfect pragmatic quality. This further strengthens our claim that normal business users without training will have difficulties understanding advanced business processes modelled using BPMN.

5 BWW Ontology for Evaluation

The Bunge-Wand-Weber model (BWW) is an ontological model of information systems, and can be used to analyze and evaluate conceptual modeling languages [4]. By comparing the constructs of the BWW ontology to the constructs of the modeling language, one can analyze the meaning of the language constructs to help see if they are appropriate with regards to being well defined and fitting well together. In [4], four

"ontological discrepancies" are identified for evaluating ontological clarity of languages: Construct Overload, Construct Redundancy, Construct Excess and Construct Deficit.

5.1 Comparing the BWW Ontology to the Semiotic Framework

Both methods are analytical in that they evaluate languages based on a theoretical framework. The results are lists of language features that correspond to the recommendations of the frameworks, and other features that have room for improvement.

The semiotic framework is well suited for evaluating quality on a relatively general level. The BWW ontology complements this by being more concrete as to evaluating and suggesting which concrete language constructs should be used. The BWW ontology looks at the conceptual basis of modelling languages, and can not be used to evaluate other aspects of a modelling language (as for example the diagram notation). Notation and other aspects can be evaluated using the semiotic framework, giving this technique a broad focus. The BWW ontology on the other hand has a narrower focus, but evaluations thus become more thorough. Evaluations using BWW are easily made more objective than when using the semiotic framework with its more general concepts. The semiotic framework and the BWW ontology complement each other as methods to analyze conceptual modelling languages.

UML is an example of a modelling language that has been evaluated both using the BWW ontology and the semiotic framework. In [10], Opdahl and Henderson-Sellers evaluated UML using the BWW ontology. They found that many constructs in UML match well with the BWW ontology, but also suggest some concrete improvements based on identified problem areas. In [3], Krogstie evaluated UML using the semiotic framework. He suggests different but useful improvements, based on issues identified in e.g. problems of comprehensibility. Despite their different findings, both these papers have the same basic conclusion that UML is a useful language but with some weaknesses.

6 Related Work

The semiotic quality framework has been used to evaluate several conceptual modelling languages. In [3], Krogstie evaluated UML using the semiotic framework. The framework was also used by Su and Ilebrekke [11] to compare the quality of ontology languages and tools. Flon Arnesen and Krogstie [12] tailored the framework for a concrete organization's needs and used it to evaluate the quality of five enterprise process modelling languages for use in this organization.

The authors have not been able to find any other published papers evaluating BPMN. However, some evaluations of BPEL4WS have been performed. This is relevant because models created using BPMN in many cases can be mapped directly into a corresponding model in BPEL4WS and vice versa. However, BPEL4WS-models are represented in

XML and have no graphical notation. In [13], Wohed, Aalst, Dumas and Hofstede do an analysis of BPEL4WS using a framework composed of workflow and communication patterns. They conclude that although being a relatively powerful and flexible language, BPEL4WS is a complex language with partially unclear semantics. A similar conclusion is reached by Aalst in [14]. These findings correspond to this paper's suggestion that diagrams utilizing advanced features might be difficult to comprehend, especially for non-technical business users.

7 Future Work

To further elaborate on the evaluation done in this paper, the quality of the documentation and tool support for BPMN should be analyzed using the semiotic framework.

Additional evaluation of BPMN should also be performed by comparing the BWW ontology to the BPMN Metamodel. The comparison should look for construct overload, redundancy, excess and deficit [4]. One might for example find that the BPMN Metamodel lacks some general concepts relevant when modelling outside the business domain. This might be the case because BPMN was created with mainly business processes in mind. An evaluation according to the BWW ontology is useful for finding the ontological discrepancies as described above. While the correctness and completeness of the BWW ontology or any other ontology can always be debated, the use of such an approach to evaluate modelling languages does provide an anchoring point for the discussion and has shown useful application results [10].

In addition to further analytical evaluation, empirical evaluation is needed to validate the results of the analytical investigations. Future work should also include comparative studies of BPMN and several other business process modelling languages.

8 Conclusion

BPMN is a functionally oriented language designed for easily modeling business processes, and is well suited for this domain. BPMN has limitations to usage within other domains (e.g. the object-oriented domain), but can be used to model general processes also outside the business domain. BPMN has a familiar and easy basic graphical notation, but also include complex and advanced features that probably require a fair amount of training for non-technical users to learn. BPDs have relatively good comprehensibility appropriateness due to categorization of the types of graphical elements and support for aggregation of Activities. Technical actors may fairly easily map BPDs into the BPEL4WS format, but creation of Web Services representing the Activities is required to make an executable system in this case.

It has been discussed how BPMN may be evaluated according to the BWW ontology,

and in what ways this may supplement the evaluation according to the semiotic framework. An evaluation according to the BWW ontology is useful for finding ontological discrepancies, and the semiotic framework is useful for evaluating quality on a relatively general level. The semiotic framework and the BWW ontology complement each other as methods to analyze conceptual modelling languages.

References

1. Krogstie, J., Sølvberg, A.: Information Systems Engineering: Conceptual Modeling in a Quality Perspective. Kompendiumforlaget, Trondheim, Norway, 2003
2. Lindland, Sindre, Sølvberg: Understanding Quality in Conceptual Modeling. IEEE Software (pp. 42-49), 1994
3. Krogstie: Evaluating UML Using a Generic Quality Framework. UML and the unified process (pp. 1-22), 2003
4. Wand, Weber: On the ontological expressiveness of information systems analysis and design grammars. Journal of Information Systems, 3 (pp. 217-237), 1993
5. Stephen A. White: Introduction to BPMN. IBM Corporation, http://www.bpmn.org/Documents/Introduction to BPMN.pdf, 2004
6. BPNI.org's webpages for Business Process Modeling Notation (BPMN) Information. URL: http://www.bpmn.org, 2005
7. Andrews, Curbera, Dholakia, Goland et al: Specification: Business Process Execution Language for Web Services Version 1.1. IBM Corp, 2003
8. Stephen A White et al: Business Process Modeling Notation (BPMN) Version 1.0. BPMI.org, http://www.bpmn.org/Documents/BPMN V1-0 May 3 2004.pdf, 2004
9. Gemino, Wand: Evaluating Modeling Techniques Based on Models of Learning. Communications of the ACM, Volume 46, Number 10 (pp. 79-84), 2003
10. Opdahl, Henderson-Sellers: Ontological Evaluation of the UML Using the Bunge-Wand-Weber Model. Software and System Modeling 1 (1) (pp. 43-67), 2002
11. Su, Ilebrekke: A Comparative Study of Ontology Languages and Tools. Proceedings of the 14th International Conference on Advanced Information Systems Engineering (CAiSE'02) (pp 761-765), 2002
12. Arnesen, Krogstie: Assessing Enterprise Modelling Languages using a Generic Quality Framework. Proceedings of the 7.th CAiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD'02), 2002
13. Wohed, Aalst, Dumas, Hofstede: Analysis of Web Services Composition Languages: The Case of BPEL4WS. In 22nd International Conference on Conceptual Modeling (ER 2003), Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2003
14. Aalst: Don't go with the flow: Web services composition standards exposed. IEEE Intelligent Systems 18(1) (pp. 72–76), 2003