

**This is the author-manuscript version of this work - accessed from
<http://eprints.qut.edu.au>**

zur Muehlen, Michael and Recker, Jan C. and Indulska, Marta (2007) Sometimes Less is More: Are Process Modeling Languages Overly Complex?. In Taveter, Kuldar and Gasevic, Dragan, Eds. *Proceedings 3rd International Workshop on Vocabularies, Ontologies and Rules for The Enterprise*, Annapolis, Maryland.

Copyright 2007 IEEE

Sometimes Less is More: Are Process Modeling Languages Overly Complex?

Michael zur Muehlen
Stevens Institute of Technology
mzurmuehlen@stevens.edu

Jan Recker
Queensland University of Technology
j.recker@qut.edu.au

Marta Indulska
The University of Queensland
m.indulska@business.uq.edu.au

Abstract

Modern business process modeling languages such as BPMN or EPC provide users with more constructs to represent real world situations than their predecessors such as IDEF or Petri Nets. But this apparent increase in expressiveness is accompanied by an increase in language complexity. In practice many organizations choose to only use a subset of the available modeling constructs. Using a well-established ontology-based theory of representation, we analyze how this voluntary restriction affects the expressiveness and complexity of the resulting modeling vocabulary. We compare our empirical findings with two notation sets of the popular language BPMN – the core and full set. Our findings indicate that users are willing to accept ambiguity among modeling constructs and that the full element set of BPMN adds little expressiveness at the expense of considerably decreased ontological clarity. The findings are a first step towards an understanding of an optimal cost-effectiveness ratio for process modeling languages- both in theory and practice.

1. Introduction

The design, improvement, and management of processes has been identified as the number one priority of CIOs for a number of years [6]. But

despite the increasing awareness for Business Process Management (BPM) efforts, organizations struggle to effectively implement process improvement programs [5]. Particularly the initial phases of BPM projects, process discovery and documentation, can take up more than a third of the overall project time [9].

A large number of graphical process modeling languages has been developed to aid organizations in the documentation of their processes. These languages range from simple flowcharting techniques to more advanced languages capable of capturing information required for process simulation and execution. A recent study by Rosemann et al. [17] found that, over time, process modeling languages have become more and more expressive and provide richer semantics. At the same time, however, the languages have become more complex and potentially cumbersome since they present users with a large variety of language constructs for process modeling. For instance, while Flowcharts (discussed as early as 1958 [11]) offer 6 basic constructs and 4 extended constructs, BPMN (published in 2006) offers 11 basic constructs and 39 extended constructs.

This increasing number of constructs prompts us to ask: Do users actually make use of the large variety of constructs or do they limit their language use to a potentially less expressive subset in order to reduce language complexity?

A similar situation has previously been witnessed in the case of UML. UML 2.0 contains nine diagramming languages comprising nearly two hundred constructs, thereby affording considerable *theoretical complexity* [19]. Siau et al. [19] found that only few of the UML 2.0 constructs are used in practice, which reduces the *practical complexity* of UML significantly. They correspondingly argue that attention should be paid to whether the organizational (i.e., practical) use of a language is in fact compliant with the original (i.e., theoretical) specification of the language. Similar to this scenario, the goal of our research is to investigate, analytically and empirically, how organizations use process modeling languages. In order to determine the level of language complexity that organizations are willing to accept, we use a metric based on the expressiveness of process modeling languages.

Accordingly, the *aim of this paper* is to report on a measure of complexity for process modeling languages derived from an ontology-based theory of representation [20, 21, 22] and to contrast the theoretical complexity of process modeling languages against their practical complexity, as observed in a case study where BPMN was applied to document the service process of an automotive company.

Our investigation follows the case of the BPMN language [1], an industry standard for the graphical representation of business process models. BPMN shares properties with UML. Both have been ratified by the standardization body OMG. Both contain a larger set of constructs in contrast to competing languages, offering a multitude of options for modeling. Both have been found in analytical studies to not only be semantically richer but also theoretically more complex than others.

We proceed as follows: In the next section we briefly introduce BPMN as our unit of analysis and present the background on the underlying case. In section 3 we introduce the theoretical framework – representation theory - upon which we base our investigation. In section 4 we present our research design. We discuss our findings from the analytical and empirical investigation in section 5. Section 6 concludes this paper with a review of contributions and an outlook to future work.

2. Background

2.1. Introduction to BPMN

BPMN [1] is a graphical process modeling language that originated from the revision of other

notations including UML, IDEF, ebXML, RosettaNet, LOVeM and EPCs. The development of BPMN stemmed from the demand for a graphical notation that complements the BPEL4WS standard for executable business processes. The specification document differentiates the BPMN constructs into two sets, *viz.*, core set and full set. The *core set* contains eleven basic graphical elements. The designers of BPMN speculated that these basic elements are sufficient for creating simple and easy-to-understand process models even for inexperienced users and audiences. The *full set* defines thirty-eight distinct language constructs plus attributes, grouped into four basic categories: Flow Objects, Connecting Objects, Swimlanes and Artifacts. The BPMN designers intended for these extensions to afford BPMN an expressive power that allows for the capture of advanced and complex process scenarios as well as process execution. For more information on BPMN refer to [1].

2.2. Case background

We have gathered BPMN diagrams from a process improvement project at a truck dealership in Connecticut, USA. The family-owned company was acquired by a group of investors in 2006 and a substantial business improvement program was put in place. One reason for the acquisition was a generational transition in ownership; the other was the need to modernize operations in an increasingly competitive market.

In order to improve business performance, the company commissioned an analysis of their core service management process, which starts when a customer delivers a truck for scheduled or unscheduled maintenance, and ends when the customer is billed for the services rendered and drives the truck off the premises of the company. BPMN was employed to graphically capture the as-is process and to propose a to-be scenario. Over the course of the project, 13 versions of the as-is process model, and 6 versions of the to-be process model were generated. During these iterations we sought validation of the content of the process models and also solicited feedback on the understandability of the constructs used for modeling. Constructs that were perceived as cumbersome to understand and/or ambiguous were eliminated, amended or replaced in order to improve the readability of the models, which we found improved end user acceptance. At the end of the project, a total set of fourteen BPMN constructs was used in the creation of the as-is and to-be models.

3. Representation theory

We seek to develop a measure of process modeling language complexity. Modeling languages in general are concerned with providing a user with graphical constructs to articulate real-world domains for the purpose of understanding and communication [18]. A natural starting point for the analysis of modeling languages is thus ontology, the branch of philosophy that studies the most pervasive features of reality, such as real existence, change, time, causation, chance, life, mind, and society [3].

Wand and Weber [20, 21, 22] developed and refined a theory of representation based on an ontology defined by Bunge [2] for the evaluation of the *representational capability* of modeling languages and the models produced with these languages. The procedure of applying their theory has become known as representational analysis [16]. More specifically, Wand and Weber [20, 21, 22] derived from Bunge's ontology a set of representational constructs that, at that time, they deemed sufficient for articulating real-world domains, including relevant things, events and transformations, in the form of conceptual models. These constructs are shown on the left in Table 2. While other ontologies can be used as a basis of the representational analysis, we choose the BWW model for its formal specification, Information Systems domain focus, and demonstrated utility in the process modeling domain in particular [16]. The BWW model has been applied in a large number of studies, both analytically and empirically, to a variety of data [21], object-oriented [10], business [24], structured analysis [14] as well as process modeling languages such as EPCs [7], BPEL [8] or Petri nets [12]. Its demonstrated usefulness in these studies serves as a justification for its selection in our research. For more information about the BWW model, its development, and its constructs (listed in Table 2) please refer to [20, 21, 22].

Wand and Weber's theory suggests that process modeling languages should exhibit two characteristics in order to allow for the development of (good) descriptions of real-world domains.

First, they should be *complete*. Users should be able to articulate all real-world phenomena that they deem necessary to have represented in their process model. This principle is known as *ontological completeness* and is indicated in any language by the extent of construct deficit that the language exhibits. Second, a modeling language should be *clear*. Users

should be able to unambiguously and effortlessly articulate the real-world phenomena that they seek to represent without causing confusion to the end users. Wand and Weber labeled this principle *ontological clarity* and elaborate that ontological clarity in a language can be reduced by three situations, *viz.*, construct redundancy, construct overload and construct excess. Construct redundancy exists when two or more language constructs share the capacity to model the same real-world phenomenon. Construct overload exists when a language has the capacity to represent more than one real-world phenomenon. Construct excess exists when a language construct does not have the capacity to model any relevant aspect of a real-world domain. Figure 1 shows the main concepts and premises of representation theory.

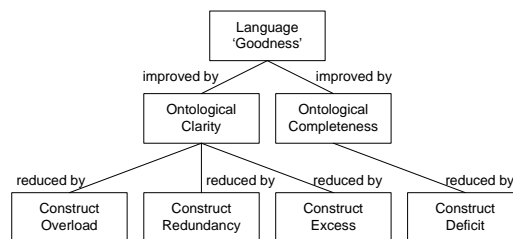


Figure 1: Premises of representation theory [23]

Clearly, the premises of representation theory appear conducive to developing a measure of the complexity of a process modeling language. A high degree of ontological completeness (i.e., a low degree of construct deficit) would mean that users are able to depict all relevant aspects of real-world domains. This coverage capacity, however, can be negatively affected by a lack of ontological clarity. The clarity of a language describes how unambiguous the meaning of its constructs is specified and thus, how much effort is needed to apply desired real-world meaning to the constructs.

While a certain language may provide sufficient coverage (indicated by a low degree of construct deficit), it may be complex to use because this scope of coverage may come at the expense of redundant, overloaded or excessive constructs (indicated by high degrees of construct redundancy, overload and excess).

4. Research design

Representation theory offers the theoretical foundation for measuring the complexity of a language: given a certain scope of coverage, as defined by the degree of deficit in a language, one is able to gauge, and compare, the complexity of the

language by establishing its degrees of redundancy, overload and excess.

We seek to apply these measures to three scenarios: first, the core BPMN set, the full BPMN set and the set of BPMN constructs that was used in our case study – an ‘empirically valid’ set of BPMN constructs.

In order to establish these measures, first, the four degrees of deficiencies have to be identified. At the forefront of representation theory is the so-called Bunge-Wand-Weber representation model, which specifies a set of representation constructs that conceptual modeling grammars should provide language constructs for in order to be able to develop faithful models of real-world domains.

The procedural model by Rosemann et al. [15] is followed in creating a representation mapping between the constructs in the BPMN language to the ontological constructs specified in Wand and Weber’s representation model. Based on this mapping, the four scenarios of construct deficit, redundancy, overload and excess can be identified within the three BPMN sets (see Figure 2).

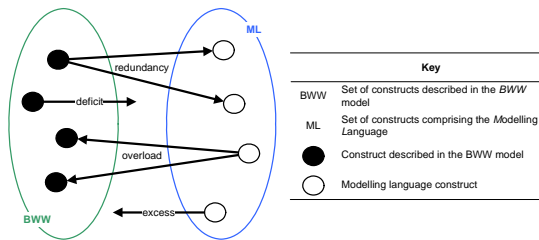


Figure 2: Mapping relationships [21]

Following the representation mapping, four measures can be established:

1. The *degree of completeness* (DoC) is the ratio between the number of BWW constructs found to have a mapping to language constructs (#C) divided by the total number of constructs defined in the BWW representation model (#M). It is thus the inverse relation of the extent of construct deficit in a language.
2. The *degree of redundancy* (DoR) is the ratio between the number of language constructs found to have a mapping to the same BWW construct (#R) divided by the total number of constructs in the modeling language (#L).
3. The *degree of overload* (DoO) is the ratio between the number of language constructs found to have a mapping to more than one BWW construct (#O) divided by the total number of constructs in the modeling language (#L).

4. The *degree of excess* (DoE) is the ratio between the number of language constructs found not to have a mapping to any BWW construct (#E) divided by the total number of constructs in the modeling language (#L).

5. Findings

5.1. Results

Design and conduct of the representation mapping of the full set of BPMN constructs was described by Recker et al. [13]. We present this full mapping in Table 2, and extract from it mappings of just the core BPMN set and also the case study BPMN set.¹

From Table 2 it is possible to compute the four measures of complexity derived from representation theory. Table 1 presents the results.

Measure	Full BPMN set	Core BPMN set	Case study BPMN set
DoC	55.26%	47.37%	50.00%
DoR	51.28%	18.18%	28.57%
DoO	25.64%	25.64%	54.55%
DoE	38.46%	45.45%	28.57%

Table 1: Complexity measures for BPMN sets

Table 1 suggests that the full BPMN set provides the best coverage in light of representation theory. Case study set and core set follow closely. In the following, we discuss the *costs* of this coverage, i.e., the complexity of the three language sets in relation to the scope of coverage they provide.

5.2. Findings & discussion

The results indicate that in terms of Degree of Completeness there is little difference between the core, full, and case study sets of BPMN. All three sets correspond roughly to half of the representation constructs defined by the BWW model. This situation indicates that even the core set, with only 25% of the constructs of the full set, has a comparable expressiveness in light of the theory.

¹ Note that in Table 2, the core BPMN construct ‘Event’ is listed as ‘Start Event’. This is because the definition of the core ‘Event’ construct corresponds to the definition of the ‘Start Event’ in the full set. See [1].

BWW construct	Full BPMN set	Core BPMN set	Case study BPMN set
THING	Pool, Lane	Pool, Lane	Pool, Lane
PROPERTY			
in general	Attributes of Pools, Attributes of Lanes	N/A	N/A
in particular	N/A	N/A	N/A
hereditary	N/A	N/A	N/A
emergent	N/A	N/A	N/A
intrinsic	N/A	N/A	N/A
non-binding mutual	N/A	N/A	N/A
binding mutual	N/A	N/A	N/A
attributes	N/A	N/A	N/A
CLASS	Lane, Data Object	Lane, Data Object	Lane
KIND	Lane	Lane	Lane
STATE	N/A	N/A	N/A
CONCEIVABLE STATE SPACE	N/A	N/A	N/A
LAWFUL STATE SPACE	N/A	N/A	N/A
STATE LAW	N/A	N/A	N/A
STABLE STATE	N/A	N/A	N/A
UNSTABLE STATE	N/A	N/A	N/A
HISTORY	N/A	N/A	N/A
EVENT	Start Event, Intermediate Event, End Event, Message, Timer, Error, Cancel, Compensation, Terminate	Start Event	Start Event, End Event, Intermediate Message
CONCEIVABLE EVENT SPACE	N/A	N/A	N/A
LAWFUL EVENT SPACE	N/A	N/A	N/A
EXTERNAL EVENT	Start Event, Intermediate Event, End Event, Message, Timer, Error, Cancel, Compensation	Start Event	Start Event, End Event, Intermediate Message
INTERNAL EVENT	Start Event, Intermediate Event, End Event, Message, Error, Cancel, Compensation, Terminate	Start Event	Start Event, End Event, Intermediate Message
WELL-DEFINED EVENT	Compensation, End Event	Start Event	End Event
POORLY-DEFINED EVENT	Message, Timer, Error, Cancel, Terminate, Start Event, Intermediate Event	Start Event	Start Event, Intermediate Message
TRANSFORMATION	Activity, Task, Collapsed Sub-Process, Expanded Sub-Process, Nested Sub-Process, Transaction	Activity	Task
LAWFUL TRANSFORMATION	Default Flow, Uncontrolled Flow, Exception Flow	Default Flow	Default Flow
stability condition	Rule, Conditional Flow	N/A	N/A
corrective action	'Exception Task', Compensation Activity	N/A	N/A
ACTS ON	Message Flow	Message Flow	Message Flow
COUPLING	Message Flow	Message Flow	Message Flow
SYSTEM	Pool, Lane	Pool, Lane	Pool, Lane
SYSTEM ENVIRONMENT	Pool, Lane	Pool, Lane	Pool, Lane
SYSTEM COMPOSITION	Pool, Lane	Pool, Lane	Pool, Lane
SYSTEM DECOMPOSITION	N/A	N/A	N/A
SYSTEM STRUCTURE	Pool, Lane	Pool, Lane	Pool, Lane
SUBSYSTEM	Pool, Lane	Pool, Lane	Pool, Lane
LEVEL STRUCTURE	Pool, Lane	Pool, Lane	Pool, Lane
EXCESS	Link, Off-Page Connector, Gateway Types, Association Flow, Text Annotation, Group, Activity Looping, Multiple Instances, Normal Flow, Event (super type), Gateway (super type)	Association Flow, Text Annotation, Group, Gateway (super type)	Association Flow, Text Annotation, Gateway (super type), Gateway Types

Table 2: Representation mapping outcomes

Similarly, the set of BPMN constructs used in the case study had comparable scope of coverage to the full BPMN set whilst using only 14 of the 39 constructs. Turning to the complexity that is afforded by these scopes of coverage, Figure 3 shows how the Degrees of Completeness obtained by the three sets are positioned in relation to the Degrees of Redundancy, Overload and Excess.

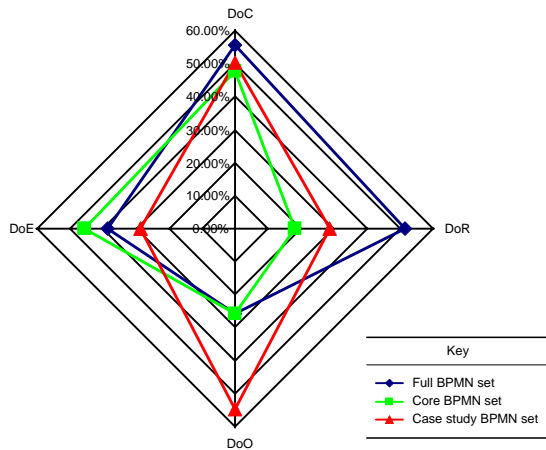


Figure 3: Complexity diagram

In light of representation theory, Figure 3 suggests that the full set of BPMN offers a wide range of representationally redundant constructs. The Degree of Redundancy, which increases from 18% in the core set to 51% in the full set, indicates this. This means that an increase of 8% in degree of completeness comes at the expense of 33% more redundancy.

In terms of Degree of Overload the core and full set are identical, while the case study set is nearly twice as overloaded. This indicates that in practice the degree of ambiguity introduced by constructs that have multiple representational capacities does not appear to affect the decision of users to choose modeling constructs. In fact, it would appear that users would favor a flexible and potentially ambiguous specification of constructs (such as Lane or Pool) over rigid and fixed semantics.

Considering the Degree of Excess, the case study set contains the least number of constructs that have no mapping to the BWW constructs. This situation indicates that the BWW representation model in itself is a suitable collection of constructs to represent real world domains. In fact, case study users avoided BPMN constructs that in light of the theory did not

offer representational capacity, as shown by the decreased Degree of Excess.

A closer look at the BPMN constructs contained in the case study set reveals that the set is comprised of 8 of the original 11 core set constructs. These are complemented by 4 constructs from the full BPMN set. The missing elements from the core set are Group, Data Object, and generic events. While the lack of the group object may be attributed to the specific needs of the project at hand, the absence of the generic event symbol is easily explained by the inclusion of specific event constructs from the full set, which together make up 3 of the 4 extended constructs. The final construct from the full set is the specialized gateway construct (XOR and AND gateways). This would indicate that users opted to *refine* constructs from the core set, rather than *introducing* constructs that have no relation to the core set constructs.

For BPMN designers, the implications of our findings are twofold: On the one hand, users prefer the different event and gateway types that are part of the full set to the generic types contained in the core set. On the other hand, elements that are used to structure the graphical representation of the diagrams (grouping, off-page links etc.) seem to be of lesser importance. Our findings in this regard align with the arguments offered by Rosemann et al. [17] who speculated that some of the BPMN constructs, such as off-page connectors, group, text annotation etc., would be of limited use to process modelers due to existing tool support, which often provides similar means to assist process modeling exercises. One conclusion would thus be to externalize these constructs from a process modeling language into a modeling tool to thereby reduce the complexity of a language without decreasing representational capacities.

While we concede that the *purpose* for which BPMN is used in the organization may have an impact on the appropriateness of certain constructs, our initial results indicate that more constructs may not necessarily improve the representational 'goodness' of a language. Indeed, in our case study, in which BPMN was used for the purpose of communicating with process stakeholders, we observe a preference for lower DoE and DoR but higher DoO.

This finding is in line with the differences between business and workflow process modeling suggested by Dehnert et al. [4]. They state that process modeling for business purposes has the requirement of understandable models suitable for multiple interpretations, while technical workflow

models impose restrictions regarding ambiguity and machine-readability.

This suggests that higher DoO would potentially not be allowed in situations where BPMN is used for technical specification, rather than for communication purposes. This aspect remains to be tested but it might indicate that there is a need for BPMN designers to explicitly define sets of constructs for different purposes, rather than defining a full and core set. Such explicit specification would help organizations in the correct selection of the required constructs, thus reducing complexity of modeling for their purpose.

6. Conclusions

6.1. Contributions

In this study we were motivated to examine the differences in expressiveness and complexity of three sets of BPMN constructs, *viz.*, full set, core set, and a set obtained via a process improvement case study. This motivation stemmed from the recognition that organizations impose or discover their own sets of ‘allowed’ BPMN constructs, while the BPMN specification itself also defines different sets of constructs. Our initial study, within the context of modeling for stakeholder communication purposes, shows that actual BPMN users tended to strive for maximum possible scope of coverage within BPMN, however, they also favored a set that had lowest levels of excess constructs and less redundancy than observed in the full BPMN set. At the same time, the users were content to increase the level of construct overlap (i.e. potentially increasing ambiguity) in order to increase their ability to model more concepts. Our set of case study BPMN constructs was derived over nineteen versions of process models, with each iteration striving for more accurate and less complex representation of the process. Hence, overall, our results would indicate that in some cases an increase in constructs does not always lead to better modeling results. What impacts such decisions, be it modeling purpose or modeler experience for example, is an open question that requires further investigation.

References

- [1] BPMI.org and OMG, *Business Process Modeling Notation Specification. Final Adopted Specification*. 2006, <http://www.bpmn.org>.

6.2. Outlook

We do not consider our research to be complete. Indeed, we used the investigation reported in this paper as a starting point for a more comprehensive study of representational capacities of process modeling languages before the background of different usage and purpose scenarios. The goal of our research remains to identify an optimal cost-effectiveness ratio for process modeling in order to be able to guide both vendors and end-users of process modeling solutions in their endeavors. Following the initial investigation reported in this paper, our research is currently taking on a number of directions. First, we have obtained over fifty ‘real world’ BPMN models from a large variety of case settings. Through the analysis of this large set of data we aim to identify the set of BPMN constructs actually in use. This analysis is currently underway and will allow us to replicate our findings from the initial complexity measurement and obtain generalizable statistically validated findings.

Second, we will then contrast our findings in relation to the varied process modeling purposes for which the obtained models were developed. This will allow to us to obtain insights to what extent the premises of representation theory hold for different process modeling scenarios.

Third, we will aim to derive from the analytical and empirical scenarios the ideal subset of BPMN constructs that maximizes DoC while minimizing DoE, DoO and DoR.

A related stream of research will also focus on the suitability of the BWW model for analysis of process modeling techniques. Previous research (e.g. [13]) has shown that the BWW model is able to uncover deficiencies in a modeling technique, which are then confirmed by practitioners. However, some analyses also resulted in a number of unsupported propositions. Hence, we are interested to investigate whether the BWW model can be further focused for the process modeling domain. Such investigation will include an interpretation mapping (i.e., a mapping from process modeling language to BWW model) in order to better understand how the BWW fares with respect to popular process modeling methods.

- [2] M.A. Bunge, *Treatise on Basic Philosophy Volume 3: Ontology I - The Furniture of the World*, Dordrecht, The Netherlands: Kluwer Academic Publishers, 1977.

- [3] M.A. Bunge, *Philosophical Dictionary*, New York, New York: Prometheus Books, 2003.
- [4] J. Dehnert and W.M.P. van der Aalst, "Bridging The Gap Between Business Models And Workflow Specifications," *International Journal of Cooperative Information Systems*, vol. 13, no. 3, 2004, pp. 289-332.
- [5] Gartner Group, *Delivering IT's Contribution: The 2005 CIO Agenda*, EXP Premier Report January 2005, Gartner, Inc, 2005.
- [6] Gartner Group, *Creating Enterprise Leverage: The 2007 CIO Agenda*, EXP Premier Report January 2007, Gartner, Inc, 2007.
- [7] P. Green and M. Rosemann, "Integrated Process Modeling. An Ontological Evaluation," *Information Systems*, vol. 25, no. 2, 2000, pp. 73-87.
- [8] P. Green, M. Rosemann, M. Indulska, and C. Manning, "Candidate Interoperability Standards: An Ontological Overlap Analysis," *Data & Knowledge Engineering*, vol. 62, no. 2, 2007, pp. 274-291.
- [9] M. Indulska, S. Chong, W. Bandara, S. Sadiq, and M. Rosemann, "Major Issues in Business Process Management: An Australian Perspective," in *Proceedings of 17th Australasian Conference on Information Systems*, Adelaide, Australia, 2006, Australasian Association for Information Systems.
- [10] A.L. Opdahl and B. Henderson-Sellers, "Ontological Evaluation of the UML Using the Bunge-Wand-Weber Model," *Software and Systems Modeling*, vol. 1, no. 1, 2002, pp. 43-67.
- [11] R. Péter, "Graphschemata und rekursive Funktionen," *Dialectica*, vol. 12, no. 3-4, 1958, pp. 373-393.
- [12] J. Recker and M. Indulska, "An Ontology-Based Evaluation of Process Modeling with Petri Nets," *Journal of Interoperability in Business Information Systems*, vol. 2, no. 1, 2007, pp. 45-64.
- [13] J. Recker, M. Indulska, M. Rosemann, and P. Green, "How Good is BPMN Really? Insights from Theory and Practice," in *Proceedings of 14th European Conference on Information Systems*, Goeteborg, Sweden, 2006, Association for Information Systems, pp. 1582-1593.
- [14] F. Rohde, "An Ontological Evaluation of Jackson's System Development Model," *Australian Journal of Information Systems*, vol. 2, no. 2, 1995, pp. 77-87.
- [15] M. Rosemann, P. Green, and M. Indulska, "A Reference Methodology for Conducting Ontological Analyses," in H. Lu, W. Chu, P. Atzeni, S. Zhou, and T.W. Ling, ed., *Conceptual Modeling – ER 2004*, Vol. 3288, Lecture Notes in Computer Science, Shanghai, China: Springer, 2004, pp. 110-121.
- [16] M. Rosemann, P. Green, M. Indulska, and J. Recker, "Using Ontology for the Representational Analysis of Process Modeling Techniques," *International Journal of Business Process Integration and Management*, forthcoming, pp. in press.
- [17] M. Rosemann, J. Recker, M. Indulska, and P. Green, "A Study of the Evolution of the Representational Capabilities of Process Modeling Grammars," in E. Dubois and K. Pohl, ed., *Advanced Information Systems Engineering - CAiSE 2006*, Vol. 4001, Lecture Notes in Computer Science, Luxembourg, Grand-Duchy of Luxembourg: Springer, 2006, pp. 447-461.
- [18] K. Siau, "Informational and Computational Equivalence in Comparing Information Modeling Methods," *Journal of Database Management*, vol. 15, no. 1, 2004, pp. 73-86.
- [19] K. Siau, J. Erickson, and L.Y. Lee, "Theoretical vs. Practical Complexity: The Case of UML," *Journal of Database Management*, vol. 16, no. 3, 2005, pp. 40-57.
- [20] Y. Wand and R. Weber, "An Ontological Model of an Information System," *IEEE Transactions on Software Engineering*, vol. 16, no. 11, 1990, pp. 1282-1292.
- [21] Y. Wand and R. Weber, "On the Ontological Expressiveness of Information Systems Analysis and Design Grammars," *Journal of Information Systems*, vol. 3, no. 4, 1993, pp. 217-237.
- [22] Y. Wand and R. Weber, "On the Deep Structure of Information Systems," *Information Systems Journal*, vol. 5, no. 3, 1995, pp. 203-223.
- [23] R. Weber and Y. Zhang, "An Analytical Evaluation of NIAM's Grammar for Conceptual Schema Diagrams," *Information Systems Journal*, vol. 6, no. 2, 1996, pp. 147-170.
- [24] H. Zhang, R. Kishore, and R. Ramesh, "Semantics of the MibML Conceptual Modeling Grammar: An Ontological Analysis Using the Bunge-Wand-Weber Framework," *Journal of Database Management*, vol. 18, no. 1, 2007, pp. 1-19.