

ARE WE THERE YET?

SEAMLESS MAPPING OF BPMN TO BPEL4WS

Marta Indulska
UQ Business School
The University of Queensland
Brisbane, Australia
m.indulska@business.uq.edu.au

Peter Green
UQ Business School
The University of Queensland
Brisbane, Australia
p.green@business.uq.edu.au

Jan Recker
School of Information Systems
Queensland University of Technology
Brisbane, Australia
j.recker@qut.edu.au

Michael Rosemann
School of Information Systems
Queensland University of Technology
Brisbane, Australia
m.rosemann@qut.edu.au

Abstract

Current process management practice strives for better support for mappings between process models in the design and execution phases of process implementation projects. In this context, BPMN was developed as a graphical representation for the process execution language BPEL4WS, with the intention that BPMN can be automatically mapped to executable BPEL4WS specifications. However, there is doubt over whether this translation is actually possible. We use the Bunge-Wand-Weber representation model to analyze the representational capabilities of BPMN and BPEL4WS, and, on this basis, argue that the translation between BPMN and BPEL4WS is prone to difficulties due to inconsistent representational capabilities. Our analysis uncovers representational inconsistencies between the specifications, and thus identifies parts of BPMN that are not readily translatable to BPEL4WS. Our work serves as a theoretical cornerstone on which the development of better mapping support for BPMN and BPEL4WS in particular, and any combination of process modeling techniques in general, can be based.

Keywords: Process design, BPEL4WS, BPMN, process automation, Bunge-Wand-Weber model

Introduction

Business practice shows that, often, different process modeling techniques are being used for defining business requirements and defining technical process specification for workflow execution (Dehnert and van der Aalst 2004). In the latter case, recent years have seen the emergence of a potential standard, Business Process Execution Language for Web Services (BPEL4WS) (Andrews et al. 2003), for specifying processes that can directly be executed via workflow engines. In fact, BPEL4WS has now become a de facto standard for specifying executable processes (Leymann and Roller 2006) and numerous tools support the execution of BPEL4WS processes (see <http://en.wikipedia.org/wiki/BPEL>). Yet, these tools mainly follow the BPEL4WS syntax and fail to provide adequate graphical model editing support. Moreover, the technical orientation and specification rigor of BPEL4WS hinders the intuitiveness of the models, with this situation potentially leading to understandability problems when non-technical stakeholders such as process managers and business analysts are involved.

In order to remedy this gap between business and technical process models, the Business Process Modeling Notation (BPMN) (BPMI.org and OMG 2006) has been developed with the clear intention to provide a standard visual notation for executable BPEL4WS processes and to define a formal mapping between BPMN and BPEL4WS (BPMI.org and OMG 2006, p. 1). Despite being a recent proposal, BPMN is already supported by more than 40 tool vendors (see www.bpmn.org), yet none of the available tools facilitates a direct and automated mapping of BPMN models to executable BPEL4WS specifications.

A closer inspection of the existing BPMN-BPEL4WS translation approaches implemented in commercial tool environments by Ouyang *et al.* (2006b) reveals that available translations fail to meet the following requirements:

- completeness, i.e., applicable translations for any BPMN model,
- automation, i.e., the capability to produce target code without requiring human intervention, and
- readability, i.e., the capability to consistently produce target code that is not only machine-readable but also understandable by human actors.

Given the significant levels of adoption of BPMN and the so far unfulfilled expectations of automatic mapping of BPMN diagrams to BPEL4WS, we see a strong need to comprehensively investigate the nature and capabilities of these two seemingly complementary process specification techniques in order to determine whether they indeed are complementary, moreover, equally expressive, and thus truly translatable. One well established method of evaluating the representational capabilities of modeling techniques that purport to capture real-world scenarios is through the use of a model of representation. A promising candidate is the Bunge-Wand-Weber (BWW) representation model (Wand and Weber 1990; 1993; 1995) that uses the principles of representational analysis for an investigation of a modeling technique's strengths and weaknesses. The BWW model provides a widely accepted means for evaluating conceptual modeling techniques for information systems analysis and design (Green and Rosemann 2004), and is used here as a reference benchmark for our analysis. The selection of this model will allow us to identify scenarios that neither of the two languages can model, scenarios that only one of the languages can model, as well as inconsistencies between their representational capabilities in scenarios that both languages can model. Indeed, our study identifies a number of areas that still need to be further considered. For example, neither BPMN nor BPEL4WS have strong support for business rule modeling. Furthermore, in terms of translation between the two languages, we predict that there will be difficulties when translating models containing BPMN's high level 'event' construct, with the lack of a corresponding general construct in BPEL4WS.

Our research is therefore motivated in three distinct ways:

1. to determine the extent of the representational capabilities of both BPMN and BPEL4WS,
2. to emphasize real-world representational requirements that neither BPMN nor BPEL4WS appear to be able to meet, and
3. to highlight and discuss areas of *representational inconsistency* between the two techniques that in turn indicate expected problem areas with regard to an automatic mapping of BPMN to BPEL4WS.

The *aim of this paper* then is to analyze, based on representation theory, the modeling capacities of BPMN and BPEL4WS, and to investigate whether it is indeed possible to specify a mapping from the former to the latter. As a measurement for the extent of modeling capacity we selected *ontological completeness*, defined as the coverage of constructs as proposed by the BWW representation model. In order to highlight representational inconsistencies between the two techniques, we use the method of *overlap analysis* (Wand and Weber 1995; Weber 1997) of the representational capabilities.

This paper is structured as follows. The next section provides an overview of the background to our study - it introduces BPMN and BPEL4WS and reviews earlier studies on their correspondence and translation. The following section discusses the research methodology. Then, the subsequent section presents the findings of the comparison from the viewpoint of ontological completeness, and it discusses representational capabilities that appear to be missing. It also presents a discussion of some of the identified inconsistencies in the representational capabilities of the two techniques and consequences of these inconsistencies for an automatic mapping from BPMN to BPEL4WS. We conclude the paper with a summary of results, and a discussion of limitations and future research.

Background & related research

An introduction to BPEL4WS and BPMN

BPEL4WS

The Business Process Execution Language for Web Services (Andrews et al. 2003) is an executable language extended with constructs specific to the BPM domain, in particular, Web Service implementations. Version 1.1 of BPEL4WS was released in 2003 and is already supported by a large number of BPM tool vendors.

BPEL4WS defines the technical details of a workflow that offers a complex Web Service built from a set of elementary Web Services. The most important concepts of a BPEL4WS process are *variables*, *partnerLinkTypes*, *basic* and *structured activities*, as well as *handlers*. Variables store process data and messages that are exchanged with Web Services. PartnerLinkTypes define the mutual required interface types of a message exchange by declaring which partner acts according to which role defined in a partner link. Basic Activities specify the operations that are performed in a process. These Web Service operations include invoke, receive, or reply. There are further activities for assigning data values to variables (assign) or waiting to halt the process for a certain time interval. Structured Activities are utilized for the definition of control flow (for example, to specify concurrency of activities via the flow construct), alternative branches (for instance, via the construct switch), or loops (while). Structured activities can be nested and links can be used to express synchronization constraints between activities. Handlers can be defined in order to respond to the occurrence of a fault, an event, or if a compensation has been triggered.

BPMN

The nature of technical, programming-oriented languages such as BPEL4WS renders them less suited for direct application by business users to design, manage, and monitor the specified business processes. Clearly, what is needed is a standard visual notation for business processes defined in executable process languages. The Business Process Management Initiative (BPMI) (www.bpmi.org) recognized this need and started work on the Business Process Modeling Notation (BPMI.org and OMG 2006) in early 2003. Version 1.0 was released in May 2004 with the intention to deliver a notation for standardized process modeling. Currently, BPMN is being considered by the Object Management Group (OMG) for standardization purposes.

The development of BPMN was mainly driven by two objectives: (a) to develop a modeling language that supports typical process modeling activities both for business and technical users, and (b) to provide a standard visualization mechanism for executable process specifications that also supports the automatic mapping from BPMN models to BPEL4WS specifications. The complete BPMN specification defines thirty-eight distinct language constructs plus attributes, grouped into four basic categories of elements. *Flow Objects*, such as events, activities and gateways, are the most basic elements used to create Business Process Diagrams (BPDs). *Connecting Objects* are used to inter-connect Flow Objects through different types of arrows. *Swimlanes* are used to group activities into separate categories for different functional capabilities or responsibilities (for example, roles, systems or departments). *Artefacts* may be added to a diagram where deemed appropriate in order to display further related information such as processed data or other comments.

Correspondence between BPMN and BPEL4WS

In an attempt to support the claim that BPMN provides a visualization mechanism for BPEL4WS, the BPMN specification (BPMI.org and OMG 2006, pp. 137-204) presents a mapping between BPMN and BPEL4WS. However, this mapping is informally written. A precise algorithm and a definition of the required structural properties is missing. An example of how a mapping could work is given by White (2005), however, the scenario used is a simplistic one, and, even then, the feasibility of such a mapping in a general case has not yet been demonstrated.

Regarding related academic research, a number of researchers have proposed transformation strategies for process models, especially in the case of BPMN and BPEL4WS. In (Ouyang et al. 2006a), a general approach is presented to translate a specific type of process models, namely, standard workflow models, to BPEL4WS by exploiting the BPEL4WS construct 'event handler'. However, as the authors admit, this approach only holds for this particular process model type and in consequence, only for a reduced set of modeling constructs of techniques such as BPMN. This approach was later adapted to the specific context of BPMN and BPEL4WS (Ouyang et al. 2006b), also relying on the discovery of process patterns in BPMN models that are then to be mapped onto BPEL4WS structured activities. This approach, too, is not yet at a stage where it holds for more advanced or complex BPMN models. Another approach is presented in (Mendling et al. 2006), where the authors discuss different strategies for translating graph-oriented models (like BPMN) to block-oriented specifications (like BPEL4WS). These strategies have different advantages and disadvantages, and also do not work in the general case.

As our brief literature review shows, existing transformation strategies falter when it comes to defining a mapping that holds in the general case for all scenarios. This situation raises the question whether BPMN and BPEL4WS really share the same representational capabilities, *i.e.*, whether they are in fact able to capture the same aspects of a modeled domain, and, if they are able to, whether they capture relevant aspects of the modeled domain in a similar, *i.e.*, complementary, fashion that

Indulska et al.

Seamless Mapping of BPMN to BPEL4WS

would allow the derivation of algorithms that are able to automatically map BPMN models to executable BPEL4WS specifications.

Research design

Representation theory in Information Systems

The research area of representational analysis is based on the understanding that Information Systems are representations of real-world systems (Wand and Weber 1995) and consequently, models of Information Systems need to contain the necessary representations of phenomena in real-world domains that are relevant to information systems analysis and domains (Wand and Weber 1993). Wand and Weber (1990; 1993; 1995) developed and refined a theory of representation based on an ontology defined by Bunge (1977) for the evaluation of the representational capability of the modeling techniques and the scripts prepared using such techniques. This theory, the BWW representation model, is one of three theoretical models defined by Wand and Weber (1995). Some minor alterations have been carried out over the years by Wand and Weber (1993; 1995) and Weber (1997), but the key constructs of the BWW representation model can be grouped into the following clusters: things including properties and types of things; states assumed by things; events and transformations occurring on things; and systems structured around things (see (Rosemann et al. 2006) for a complete list).

Weber (1997) suggests that the BWW representation model can be used to analyze a particular modeling technique in order to make predictions about the modeling strengths and weaknesses of the technique, in particular its capabilities to provide *complete* and *clear* representations of the domain being modeled. He clarifies two principal evaluation criteria that may be studied using the BWW representation model: *ontological completeness*, i.e., the analysis of the extent to which a process modeling technique covers completely the set of constructs proposed in the BWW representation model, and *ontological clarity*, i.e., the analysis of the extent to which process modeling technique constructs are deemed redundant, overloaded, or excess as per the BWW representation model.

We justify our selection of the BWW representation model as the theoretical benchmark in this study in several ways. Most notably, it has been used in over thirty research projects for the evaluation of different modeling techniques (see (Green and Rosemann 2004) for an overview), including data models, object-oriented models and reference models. In particular, it has a strong track record in the area of process modeling, with contributions coming from various researchers. Specifically, it has been used for the evaluation of such leading process modeling techniques as Petri nets (Rosemann et al. 2006), Event-driven Process Chains (EPC) (Green and Rosemann 2000), Flowcharts (Keen and Lakos 1996), Data Flow Diagrams (DFD) (Keen and Lakos 1996), Integrated Definition Method 3 Process Description Capture Method (IDEF3) (Keen and Lakos 1996), Electronic Business using eXtensible Markup Language Business Process Specification Schema (eXML BPSS) v1.01 (Green et al. 2005), Business Process Modeling Language (BPML) v.1 (Green et al. forthcoming), and Web Service Choreography Interface v1.0 (WSCI) (Green et al. forthcoming). Thus, there is an extensive background related to the application of the BWW representation model for the evaluation of process modeling techniques. Also, as Bunge's established ontology (Bunge 1977), on which the BWW model is based, has been rigorously defined using set theory notation, its adaptation to the IS-specific BWW representation model also inherits the specification rigor, and thus, can be considered theoretically sound.

Research methodology

While the application of a representational theory for the purposes of representational capability analysis of a modeling technique is a well used approach, it is not without its criticisms (Rosemann et al. 2004). Being mindful of these criticisms, which in particular relate to the *process* of analysis, we sought to strengthen the rigor of our research by employing a methodology that minimized any potential subjectivity in the analysis. In order to achieve this objective, we have chosen a multiple-coder approach with researchers who have had extensive experience with the BWW representation model, and who also were familiar with the BPMN and BPEL4WS specifications. The methodology follows a three step process (Rosemann et al. 2004) that was employed first for BPEL4WS, and then for BPMN.

1. Two researchers individually analyzed the specification, performing a mapping between the representation theory constructs and the constructs found in the technique's specification.
2. The two researchers then met to compare their analyses and discuss any discrepancies. The result of this step was an updated analysis, with a consensus between the involved researchers.
3. The analysis obtained in step 2 was then presented to two other researchers for confirmation and further discussion. This resulted in a consensus over the final accepted analysis of the technique.

Once the analyses of the representational capability of the two techniques were finalized, an *overlap analysis* (Wand and Weber 1995; Weber 1997) was performed in order to identify those representational aspects that diverge between BPMN and BPEL4WS. Such an overlap analysis allows us to produce three distinct sets of outcomes.

First, we identify representation model constructs that are not incorporated into either of the two techniques. In this context, we performed an analysis of maximal ontological completeness (MOC) (Weber 1997; Green et al. forthcoming) in order to determine the highest possible representational power when using BPMN and BPEL4WS in combination, without the explicit

Indulska et al.

Seamless Mapping of BPMN to BPEL4WS

intention to map the former to the latter (such an intention in this case, as our analysis will show, limits representational capability).

Second, we perform an inverse minimal ontological overlap (MOO) analysis (Weber 1997; Green et al. forthcoming), *i.e.*, we perform a *maximal* ontological overlap analysis. The key proposition of *minimal* ontological overlap is that users will employ a combination of techniques that overlap minimally ontologically in order to achieve greatest representational power (MOC), with the provided constructs, while maintaining minimal redundancy between the techniques. This proposition holds when a user is aiming to supplement a technique with an additional technique in order to increase representational capability without increasing complexity or ambiguity (Green et al. forthcoming). However, when we consider a situation in which a technique is meant to be automatically translatable to another technique, clearly what is required is the existence of *maximal* ontological overlap, in which case a bigger overlap of constructs sharing the same representational capacity between the two techniques would imply a larger set of constructs that can potentially be automatically mapped from one technique to the other. In our case, the maximal ontological overlap analysis is performed on a cluster by cluster basis (refer to BWW clusters shown in Table 1). Starting with the *things including properties and types of things* cluster, and taking the first construct – *thing* – we examine if both BPMN and BPEL4WS were found to have representational capability for this construct. We argue that if representational capacity exists in one technique but not the other, then problems will arise when a mapping between the two techniques is required. The problem arises because the translation of the modeled process to a less detailed technique will be at the cost of losing representational capacity power and, thus, the translation will be prone to loss of semantic information about the modeled domain. This process of analysis will then proceed for each BWW construct in the cluster, and then for all remaining clusters and their constructs.

Third, if both techniques were found to have a mapping to a representation model construct, we examine the identified technique constructs to confirm if a mapping is indeed possible between the constructs. In particular, we assess the respective construct specifications given in (Andrews et al. 2003; Green et al. forthcoming) and (BPMI.org and OMG 2006) in order to elicit whether the BPMN construct that was found to have a mapping to the respective BWW construct in fact complies with the specification of the BPEL4WS construct to which the representation model suggests it should map. Therefore, we are able to evaluate the mapping potential between BPMN and BPEL4WS on a construct level.

Representational capability of BPMN and BPEL4WS

In order to achieve the three sets of results outlined earlier, a representation analysis of BPMN and BPEL4WS was performed. A summary of this analysis is presented in Table 1, whereas the full details of the analysis have been published in (Recker et al. 2006) and (Green et al. forthcoming), respectively, with the former study also providing empirical validation of a set of propositions based on the BPMN analysis. BPMN and BPEL4WS are used at different abstraction levels, hence it is not surprising that the analyses of the two techniques do not yield the same results. However, if one is to map to the other, there still must be some commonality in terms of representation of their constructs. In particular, any representation present in one technique but not in the other would indicate a potential problem in terms of automatic mapping between the two.

Maximal ontological completeness

Maximal ontological completeness is indicated by the extent of coverage of the representation model constructs by the modeling language. If a representation theory construct does not have a mapping to a language construct, then the ability of the modeling language to represent some part of the real world relevant to information systems analysis and design is limited. Assuming that a user wants to employ BPMN and BPEL4WS in combination, as is stipulated by the specifications, we were motivated to identify whether the capability of these two languages used together to model real-world scenarios is limited by their actual level of maximal ontological completeness. In other words, representation model constructs that neither BPMN nor BPEL4WS were found to map to indicate the potential situations that the two languages are not able to model, even when used together for increased modeling capability. Here we identify such missing representation model constructs and posit about their potential ramifications. For a summary, please refer to the first row in Table 2.

Focusing on the *things including types and properties of things* cluster, visual inspection of Table 1 shows that both BPMN and BPEL4WS lack capacity for representing certain types of properties of things, namely *property in particular*, *hereditary*, *intrinsic*, and *non-binding mutual*, as indicated by a “-“ in Table 1 for these representation model constructs. Weber (1997, p. 34) points out that real-world things are perceived and described via their properties. In the area of process modeling, consequently, a lack of representation for properties results in a lack of means for modelers to comprehensively describe relevant entities or participants in their process models.

Table 1. Representation mapping of BPMN and BPEL4WS

BWW Construct	Cluster	BPMN	BPEL4WS
THING	Things including properties and types of things	Lane, Pool	-
PROPERTY in general		N/A	N/A
in particular		Attributes of Pools, Attributes of Lanes	Name
hereditary		-	-
emergent		-	Correlation Set
intrinsic		-	-
non-binding mutual		-	-
Attributes	-	Names of properties	
CLASS		Lane, Data Object	Partner
KIND		Lane	-
STATE	States assumed by things	-	Set of variables
CONCEIVABLE		-	-
STATE SPACE		-	-
LAWFUL STATE SPACE		-	-
STATE LAW		-	-
STABLE STATE		-	-
UNSTABLE STATE		-	-
HISTORY	-	-	
EVENT	Events and transformations occurring on things	Start Event, Intermediate Event, End Event, Message, Timer, Error, Cancel, Compensation, Terminate	Message, Reply, Create Instance (on Activity), Wait
CONCEIVABLE		-	-
EVENT SPACE		-	-
LAWFUL EVENT SPACE		-	-
EXTERNAL EVENT		Start Event, Intermediate Event, End Event, Message, Timer, Error, Cancel, Compensation	Message
INTERNAL EVENT		Start Event, Intermediate Event, End Event, Message, Error, Cancel, Compensation, Terminate	Reply, Create Instance (on Activity), Wait
WELL-DEFINED EVENT		End Event, Compensation	Create Instance (on Activity)
POORLY DEFINED EVENT		Start Event, Intermediate Event, Message, Timer, Error, Cancel, Terminate	Message, Reply
TRANSFORMATION		Activity, Task, Collapsed Sub-Process, Expanded Sub-Process, Nested Sub-Process, Transaction	Event Handler, Compensate, Compensation Handler, Fault Handler, Receive, Assign, Throw, Terminate, Alarm Event (onAlarm), Message Event (onMessage)
LAWFUL TRANSFORMATION		Default Flow, Uncontrolled Flow, Exception Flow	While, Pick, Switch
stability condition		Rule, Conditional Flow	Expression
corrective action		'Exception Task', Compensation Activity	-
ACTS ON		Message Flow	Role
COUPLING	Message Flow	Service Link	
SYSTEM	Systems structured around things	Pool, Lane	Business Process Instance
SYSTEM COMPOSITION		Pool, Lane	Partners
SYSTEM ENVIRONMENT		Pool, Lane	-
SYSTEM STRUCTURE		-	Partners
SUBSYSTEM		Pool, Lane	-
SYSTEM DECOMPOSITION		Pool, Lane	-
LEVEL STRUCTURE		Pool, Lane	-

Similarly, both BPMN and BPEL4WS do not have mappings for constructs in the *states assumed by things* cluster, with the exception of BPEL4WS's loose representation for *state*. In this regard, Weber (1997, p. 37) notes that states capture the values of the attributes of a thing. During its lifecycle, that thing/entity traverses multiple states. In the arena of process modeling, a lack of means for describing states results in a lack of support for business rule definitions (for empirical support for this proposition refer, for example, to (Davies et al. 2004; Recker et al. 2006)). This undesirable situation results in a need for complementary techniques with which to specify business rules, orthogonal to the use of BPMN and BPEL4WS, e.g., with the use of business rule engines or state diagrams.

Indulska et al.

Seamless Mapping of BPMN to BPEL4WS

In the cluster of *events and transformations occurring on things*, the combination of BPMN and BPEL4WS is able to model all representation theory constructs apart from capturing conceivable and lawful event spaces (see Table 1). Events are state changes in a thing (Weber 1997, p. 41). Accordingly, an event space denotes the set of all events that may occur upon a thing. Therefore, a lawful event space relates to all business rules over states that process entities may move to over time. In process modeling, a lack of means for describing conceivable and lawful event spaces results in modelers being unable to capture which events potentially could occur and affect a business process, and which events may occur but should not be allowed to occur. Such information, however, could be vital to data recovery in exception handling or compensation processes.

Constructs mapped to the *systems structured around things* cluster were found to have full representation by the combination of BPMN and BPEL4WS (see Table 1). This implies that, according to representation theory, there are no limitations in representing the structure or the environment of processes, if the two are used in combination for representation of processes.

Mapping BPMN to BPEL4WS: indications from theory

Based on the observation that different representational capabilities of languages result in a difference in the coverage of real-world scenarios, we argue that we can use Representation Theory to indicate constructs that are not able to be unambiguously, and thus automatically, translated. Our argumentation rests on the observation that, unlike human recipients, machines simply cannot accommodate ambiguity and/or equivocality (Weber 1997, p. 76). Such ambiguity would, however, indeed stem from a lack of support for a representation of a real-world phenomenon that was captured in the original model but which cannot be represented in an equivalent form in the target.

In order to systematically identify such cases, we proceed by first identifying modeling capabilities that exist in BPMN but not in BPEL4WS (thus indicating that a mapping may not be possible). We then progress to identifying the capabilities of BPEL4WS that do not appear to exist in BPMN. These scenarios give indications as to how BPMN models can be further annotated or extended in order to make a mapping to BPEL4WS work. Finally, we apply the earlier introduced principle of *maximal ontological overlap* in order to identify the representation constructs that should be able to be mapped from one language to the other, and we analyze whether this is indeed the case. The need for this particular investigation stems from the observation that while two languages may both contain constructs for the same representation theory construct, these language constructs may diverge in the particular type of representation theory construct it is mapped to (Green and Rosemann 2000). We again approach these analyses on a cluster by cluster basis.

BPMN representations missing in BPEL4WS

Within the *things including types and properties of things* cluster, we observe from Table 1 that BPMN was found to have representations for the BWW model constructs of *thing* and *kind*, both of which are not featured in BPEL4WS. Consequently, this situation implies that while specific instances of an entity (*thing*), sets of entities (*class*), or subtypes of entities (*kind*), may be depicted in BPMN models, BPEL4WS specifications can only model sets of things but not specific instances. In this regard, Bodart et al. (2001) point out that the use of optionality that stems from a focus on classes, as opposed to instances or subtypes, may result in a superficial understanding of the specification. Moreover, object instances in a BPMN model (for example a specific organizational entity, a specific business partner or a specific application system) may need to be generalized to classes of instances, i.e., to a more aggregate level, in order to be represented in BPEL4WS.

In terms of the *events and transformations occurring on things* cluster, BPMN provides for the representation of the BWW construct *corrective action* that specifies how the values of a thing must be changed in order for it to reach a lawful state. While BPEL4WS provides representational capacity for expressing stability conditions that are required to evaluate if a set of activities should be executed, it is not able to specify the action itself, i.e., the state change required for such a lawful transformation. This potentially impacts the translation of BPMN models that contain exception handling and compensation activities (for example, escalation procedures or transaction roll-backs) that appear to be specified in BPEL4WS implicitly in workarounds or a composition of specification constructs rather than via direct representations.

Last, regarding the *systems structured around things* cluster, Table 1 indicates a surplus of representational capability in BPMN in comparison to BPEL4WS. In particular, BPEL4WS is missing the capacity to represent the environment of a system, its subsystems, the decomposition of a system, and its level structure. In consequence, demarcations of processes in inter-organizational collaboration scenarios, and different abstraction levels of processes and its entities may not be fully translatable to BPEL4WS specifications. One potential reason for this situation is the fact that BPEL4WS is an execution language that enacts one process from the perspective of one process stakeholder (or system) at a time. Thus, in execution, there is no need to structure a process or its entities into different levels. On the other hand, we foresee potential problems in explicitly defining sub-processes and their instantiation/invocation that may have been captured in a BPMN process model.

In summary, representations for *thing*, *kind*, *corrective action*, *system environment*, *subsystem*, *system decomposition* and *level structure*, which are supported by BPMN, do not appear to be clearly supported by BPEL4WS. Accordingly, we expect potential difficulties in mapping from these BPMN model representations to BPEL4WS (see row two of Table 2).

BPEL4WS representations missing in BPMN

With respect to the *things including types and properties of things* cluster, Table 1 shows that BPEL4WS is capable of representing *emergent* properties as well as *attributes*, *i.e.*, it has means for naming properties we associate with things and also has specific representations for those properties that belong to composite things. Looking at a potential translation from BPMN to BPEL4WS, this indicates that general properties that may have been defined in BPMN models can more specifically be attributed in BPEL4WS. On the other hand, in order to do so, BPMN models would need to be further annotated with additional information. These annotations might in turn increase the complexity of the model and limit its intuitiveness, as well as resulting in a situation in which the model cannot be automatically translated due to human intervention required to interpret the (usually informally described) annotations.

Both BPMN and BPEL4WS are significantly deficient in representing states assumed by things, with BPMN being unable to do so. The BPEL4WS specification, however, indicates that a *state* can be represented by means of a set of variables. Yet, Green *et al.* (forthcoming) admit that there is no dedicated representation construct for representing a state in BPEL4WS. Not taking this into consideration, it appears that translation problems will not occur in this cluster, mainly due to a shared incapability for modeling states assumed by things.

In terms of the *systems structured around things* cluster, Table 1 shows that BPEL4WS has capability to represent *system structure* while BPMN lacks capacity to do so. While this does not appear to be a problem when translating a BPMN model to BPEL4WS, it would indeed be a problem if a user were to capture a BPEL4WS process in BPMN, as they would find that they would not be able to explicitly specify the set of relationships that exist between entities modeled within a given process, as well as those modeled that are external to the process. The only way to allude to the existence of such a collection of relationships would be via modeling flow or messages to/from these individual entities – however, such a representation may not give the user the same ease of understanding of how all the entities are related.

In summary, based on the BWW analysis, BPMN does not appear to have constructs for *emergent property*, *attributes*, *state*, and *system structure* (see row two of Table 2), which are supported by BPEL4WS. Hence, there exist potential problems when generating BPMN models from BPEL4WS specifications that uses these representations.

Maximal ontological overlap

Last, in our overlap analysis we assess scenarios in which it was found that BPMN and BPEL4WS both exhibit the capability to represent a particular BWW representation model construct. In these cases we consult the specification of the language constructs in order to ascertain whether or not they share the same context within being able to represent a specific aspect of the real world. For the purposes of this paper, we limit the discussion to only selected examples for each Representation Theory cluster. Our motivation here is to exemplify some further areas of concern where an automatic mapping is meant to be possible in light of the theory employed. A complete discussion of all technique constructs would exceed the length limitations of this paper and is furthermore deemed unnecessary for making our argument.

Starting with the *things including properties and types of things* cluster, more in-depth analysis of the BPMN and BPEL4WS constructs shows that attributes of pools and lanes in BPMN are able to be mapped to names in BPEL4WS because they are defined as common attributes of swimlanes in BPMN. This situation is despite the fact that BPEL4WS does not have a specific representation for a *thing*, but rather classes of things. Classes of things/entities that have been depicted via the BPMN lane construct can semantically be mapped from BPMN to BPEL4WS in the case of depicting roles of those entities only, as BPMN's Lane construct matches the BPEL4WS partner construct as per specification. BPMN's Data Objects were not found to have a mapping from BPMN to BPEL in our analysis, and this fact is also indicated in the BPMN specification (BPMI.org and OMG 2006, p. 178).

While there are no common representations in the *states assumed by things* cluster, the *events and transformations occurring on things* cluster exhibits only two representation theory constructs that are not able to be captured by BPMN and BPEL4WS. All other remaining representation theory constructs in this cluster appear on the surface to have a mapping from BPMN to BPEL4WS. However, further examination of the BPMN and BPEL4WS specifications shows that mapping in these cases is still not clear, due to the varying levels of granularity at which these two languages opted to represent the constructs. For example, BPEL4WS has no consideration of the time dimension of events. Therefore, information about BPMN's time-associated events (*i.e.*, start event, intermediate event, end event) will be lost in translation to BPEL4WS. Furthermore, there is no general "event" construct in BPEL4WS. Therefore, if a BPMN user has chosen to model an event in BPMN by using the high level event construct (*i.e.*, without using the specialized constructs in the extended set of BPMN) it will not be clear what type of BPEL4WS event it should map to. However, heading in the other direction, a specific BPEL4WS event can be mapped to the higher level BPMN event construct but with some loss of detail. Along similar lines, activities in BPMN are modeled rather in general, whereas they are specialized into different types of activities in BPEL4WS. Therefore, as in the case of translating BPMN events to BPEL4WS, there will be confusion over the type of BPEL4WS activity to which the BPMN model activity maps. This situation will require the user to bring to bear extra model knowledge in order to correctly translate the model and is not likely to be automated. Within the representation theory construct of transformation there also exists a problem if users want to translate BPMN's sub-processes, as the BPEL4WS specification does not appear to have an equivalent representation for these types of transformations according to the BWW representation model. This fact is indeed confirmed by the BPMN specification, which states that "the mapping of a Sub-Process set to a Transaction is an Open Issue"

Indulska et al.

Seamless Mapping of BPMN to BPEL4WS

(BPMP.org and OMG 2006, p. 167). Several other instances of potential translation problems exist in this cluster. For example, while BPMN's message construct can be mapped to BPEL4WS's message construct, there is confusion over when it would map to the BPEL4WS reply construct instead (BPMP.org and OMG 2006, p. 189).

Last, in the *systems structured around things* cluster, both BPMN's Pool and Lane constructs, from a system composition perspective, representation-wise should be able to map to BPEL4WS' partners construct. However, according to the BPMN specification, the constructs of Pool and Lane do not yet have a mapping to BPEL4WS at all. Our theory, however, suggests that the demarcation of roles and responsibilities in a process scenario (captured via the BPMN Lane construct) is in BPEL4WS best captured with the Partners construct. This finding may serve as a suggestion for future mapping specification that specifically takes into account aspects of organizational structures related to process design and enactment.

In summary, according to the BWW representation analysis, confusion and potential problems will appear when attempting to translate into BPEL4WS BPMN models that use the Lane, Data Object, Sub-Process, Transaction and Pool constructs, as well as the event time dimension and general (rather than specific types of) activities. For a summary please refer to the last row of Table 2.

Conclusions, limitations, and future research

In this paper we presented a BWW representation model-based analysis of the level of modeling capability that should be expected by users who employ the combination of BPMN and BPEL4WS for their process modeling needs. Furthermore, we used the analysis results in order to identify areas of concern when mapping BPMN models to BPEL4WS executable processes. The main motivation for this work was the fact that existing transformation strategies tend to collapse when defining mappings for a general case, which in turn indicates a need to discover what type of representations are not present in either of the two techniques, as well as the types of representations that cannot be unambiguously mapped from BPMN to BPEL4WS.

Our study revealed a number of representational inconsistencies between BPMN and BPEL4WS that potentially impact a translation of process models from the former to the latter. Table 2 summarizes the main findings from our research.

Table 2: Summary of overlap discussion

Aspect of analysis	BPMN	BPEL4WS
<i>Maximum Ontological Completeness:</i> Combined missing representations	<ul style="list-style-type: none"> - Property in particular, hereditary, intrinsic, non-binding mutual - Conceivable state space, lawful state space, state law, stable state, unstable state, history - Conceivable event space, lawful event space 	
<i>Overlap inconsistencies:</i> Missing BPMN representations		<ul style="list-style-type: none"> - Thing, Kind - Corrective action - System environment, Subsystem, System decomposition, Level structure
Missing BPEL representations	<ul style="list-style-type: none"> - Emergent property, attributes - State - System structure 	
<i>Construct Analysis:</i> Potential overlap problems	<ul style="list-style-type: none"> - BPMN Lane construct versus BPEL4WS Partner construct - BPMN Data Object construct misses corresponding BPEL4WS construct - BPMN event time dimensions are not captured in BPEL4WS event constructs - BPMN general activities versus differentiated BPEL4WS activity types - BPMN Sub-Process and Transaction constructs miss corresponding BPEL4WS constructs - BPMN Pool and Lane constructs do not have mapping to BPEL4WS 	

In particular, our research has found that there are a number of representation theory constructs, *viz.*, conceivable state space, lawful state space, state law, stable state, unstable state, history, conceivable event space, lawful event space, as well as some property types, that BPMN and BPEL4WS, even when used in combination, are not able to model. Each of these deficiencies has associated ramifications in terms of the level of model expressiveness. Furthermore, our research was able to identify modeling capability that is provided by BPMN but not by BPEL4WS, implying that users would encounter difficulties in the mapping of such BPMN models to BPEL4WS processes. For example, BPMN is able to model instances of an entity, while BPEL4WS is only able to model classes of entities.

Indulska et al.

Seamless Mapping of BPMN to BPEL4WS

Also, we were able to identify representations in BPEL4WS that cannot be expressed in BPMN. These lacking representations may cause BPMN modelers to employ additional specification means, such as textual annotations or complementary techniques, which in effect limits both the intuitiveness of the model and the chance of defining an automatic mapping. We were furthermore able to identify representation constructs that are supported by both BPMN and BPEL4WS. This would imply that a mapping would be less problematic than in the case of BPEL4WS missing representational capability provided by BPMN. While this is the case in many respects, our overlap analysis shows that there exist cases in which the two languages have support for a certain representation construct; however, they represent the concept at different levels of granularity. For example, while BPMN has the capacity to model an activity, an activity in BPEL is specialized into fifteen different types of activities, with no general "activity" construct. Hence, even though the two languages can both represent activities, mapping of an activity from BPMN to BPEL4WS will not be automatic and will require a user to bring to bear extra model knowledge in order to classify the BPMN activity into the respective BPEL4WS activity type.

One of the limitations of this research is our reliance on previous evaluations of BPMN and BPEL4WS by means of the BWW representation model. These studies were conducted by our team a year apart. Accordingly, limitations relating to subjectivity of analyses over time may be present (for example, learning effect). Additionally, while we use an enriched methodology that aims to minimize subjective interpretation, subjectivity may still have some degree of impact on the analysis (Rosemann et al. 2004).

In future research, we will extend this analysis to cover each BPMN and BPEL4WS construct in more detail. We will also aim to gain access to practitioners who have had experience with mapping BPMN models to BPEL4WS processes, in order to empirically validate the proposed mapping shortcomings. Future research will also need to consider rigorous transformation algorithms forthcoming from the findings from our analysis that will enable the automatic mapping of the overlapping BPMN and BPEL4WS modeling constructs.

References

- Andrews, T., Curbera, F., Dholakia, H., Golland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., and Weerawarana, S. Business Process Execution Language for Web Services. Version 1.1, 2003, available at <http://xml.coverpages.org/BPELv11-May052003Final.pdf>.
- Bodart, F., Patel, A., Sim, M., and Weber, R. "Should Optional Properties Be Used in Conceptual Modelling? A Theory and Three Empirical Tests," *Information Systems Research* (12:4), 2001, pp. 384-405.
- BPMI.org, and OMG Business Process Modeling Notation Specification. Final Adopted Specification, 2006, available at <http://www.bpmn.org>.
- Bunge, M. A. *Treatise on Basic Philosophy Volume 3: Ontology I - The Furniture of the World*, Dordrecht, The Netherlands: Kluwer Academic Publishers, 1977.
- Davies, I., Rosemann, M., and Green, P. "Exploring Proposed Ontological Issues of ARIS with Different Categories of Modellers," in 15th Australasian Conference on Information Systems, Hobart, Australia, 2004.
- Dehnert, J., and van der Aalst, W. M. P. "Bridging The Gap Between Business Models And Workflow Specifications," *International Journal of Cooperative Information Systems* (13:3), 2004, pp. 289-332.
- Green, P., and Rosemann, M. "Integrated Process Modeling. An Ontological Evaluation," *Information Systems* (25:2), 2000, pp. 73-87.
- Green, P., and Rosemann, M. "Applying Ontologies to Business and Systems Modeling Techniques and Perspectives: Lessons Learned," *Journal of Database Management* (15:2), 2004, pp. 105-117.
- Green, P., Rosemann, M., and Indulska, M. "Ontological Evaluation of Enterprise Systems Interoperability Using ebXML," *IEEE Transactions on Knowledge and Data Engineering* (17:5), 2005, pp. 713-725.
- Green, P., Rosemann, M., Indulska, M., and Manning, C. "Candidate Interoperability Standards: An Ontological Overlap Analysis," *Data & Knowledge Engineering*, in press.
- Keen, C. D., and Lakos, C. "Analysis of the Design Constructs Required in Process Modelling," in Purvis, M. (Ed.) *International Conference on Software Engineering: Education and Practice*, Dunedin, Ireland: IEEE Computer Society, 1996, pp. 434-441.
- Leymann, F., and Roller, D. "Modeling Business Processes with BPEL4WS," *Information Systems and e-Business Management* (4:3), 2006, pp. 265-284.
- Mendling, J., Lassen, K. B., and Zdun, U. "Transformation Strategies between Block-Oriented and Graph-Oriented Process Modelling Languages," in Lehner, F., Nösekabel, H., and Kleinschmidt, P. (Eds.) *Multikonferenz Wirtschaftsinformatik 2006, Band 2*, Berlin, Germany: GITO-Verlag, 2006, pp. 297-312.
- Ouyang, C., Dumas, M., Breutel, S., and ter Hofstede, A. H. M. "Translating Standard Process Models to BPEL," in Dubois, E., and Pohl, K. (Eds.) *Advanced Information Systems Engineering - CAiSE 2006*, Luxembourg, Grand-Duchy of Luxembourg: Springer, 2006a, pp. 417-432.
- Ouyang, C., Dumas, M., ter Hofstede, A. H. M., and van der Aalst, W. M. P. "From BPMN Process Models to BPEL Web Services," in Leymann, F., and Zhang, L.-J. (Eds.) *4th International Conference on Web Services*, Chicago, Illinois: IEEE Computer Society, 2006b, pp. 285-292.

Indulska et al.

Seamless Mapping of BPMN to BPEL4WS

- Recker, J., Indulska, M., Rosemann, M., and Green, P. "How Good is BPMN Really? Insights from Theory and Practice," in Ljungberg, J., and Andersson, M. (Eds.) 14th European Conference on Information Systems, Goeteborg, Sweden, 2006.
- Rosemann, M., Green, P., and Indulska, M. "A Reference Methodology for Conducting Ontological Analyses," in Lu, H., Chu, W., Atzeni, P., Zhou, S., and Ling, T. W. (Eds.) Conceptual Modeling – ER 2004, Shanghai, China: Springer, 2004, pp. 110-121.
- Rosemann, M., Recker, J., Indulska, M., and Green, P. "A Study of the Evolution of the Representational Capabilities of Process Modeling Grammars," in Dubois, E., and Pohl, K. (Eds.) Advanced Information Systems Engineering - CAiSE 2006, Luxembourg, Grand-Duchy of Luxembourg: Springer, 2006, pp. 447-461.
- Wand, Y., and Weber, R. "An Ontological Model of an Information System," IEEE Transactions on Software Engineering (16:11), 1990, pp. 1282-1292.
- Wand, Y., and Weber, R. "On the Ontological Expressiveness of Information Systems Analysis and Design Grammars," Journal of Information Systems (3:4), 1993, pp. 217-237.
- Wand, Y., and Weber, R. "On the Deep Structure of Information Systems," Information Systems Journal (5:3), 1995, pp. 203-223.
- Weber, R. Ontological Foundations of Information Systems, Melbourne, Australia: Coopers & Lybrand and the Accounting Association of Australia and New Zealand, 1997.
- White, S. A. "An Example of Using BPMN to Model a BPEL Process," in Fischer, L. (Ed.) Workflow Handbook 2005, Lighthouse Point, Florida: Future Strategies Inc., 2005, pp. 213-232.