Framework for Business Process and Rule Integration: A Case of BPMN and SBVR

Ran Cheng[1], Shazia Sadiq[1], and Marta Indulska[2]

[1] School of ITEE, The University of Queensland, St. Lucia QLD 4072 Australia
{kelvinrc,shazia}@itee.uq.edu.au
[2] UQ Business School, The University of Queensland, St. Lucia QLD 4072 Australia
m.indulska@business.uq.edu.au

Abstract. In the current heavily regulated business world, organisations struggle to establish a consistent view of their policies and operating procedures. These are generally captured through process modeling and business rule modeling, resulting in separate representations of rules and actual practice. The separation of the two models leads to increased risk of non-compliance, as well as reduced benefits from process improvement initiatives. So far, little guidance exists for organisations who struggle to consolidate their existing process models and business rules into one holistic model. In this paper we propose a preliminary framework that provides an overarching scope for the consolidation of existing business process models and business rule descriptions. The framework is supported by a collection of mapping methods (currently based on BPMN and SBVR) that assist business modelers in identifying inconsistencies between their existing business process models and business rule repositories, and thus helps to establish a coherent view of organizational policies and procedures.

Keywords: BPMN, SBVR, business process, business rule, integration.

1 Introduction

Business process modeling and business rule modeling languages are two approaches to modeling of organizational policies and procedures. Recent empirical research, however, indicates that neither approach in isolation is sufficient to represent all required details [1, 2]. Accordingly, [3] posited that the integration of the two approaches would be fruitful for organisations. Accordingly, the main aim of this paper is to take the first step towards developing a framework that facilitates a holistic view of essential techniques and methods that can be deployed to establish a consistent view of business operations using a combination of the outputs of business process and business rule modeling.

Integrating the outputs of the two modeling approaches is a challenging task. First, business process models tend to be visual in nature, with most of the relevant information represented graphically. Business rules, however, tend to be text-oriented. Thus, integration of the outputs of the two approaches requires an information exchange format with minimal information loss. Second, process models differ from

W. Abramowicz (Ed.): BIS 2011, LNBIP 87, pp. 13–24, 2011.

business rules fundamentally as they have different composing elements. Third, they are designed for different purposes - process models describe how things should happen, whereas business rules describe what should happen. Finally, the overlap and inconsistencies between business process models and business rules also presents a significant challenge. In particular, a set of criteria is required that helps a business analyst to resolve identified overlaps and inconsistencies in a satisfactory manner.

Accordingly, in this paper we propose an initial framework for process model and business rule integration. To explore the framework and explore its related methods, we select, for demonstration purposes, two popular standards for process and rule modeling - namely Business Process Modeling Notation 2.0 (BPMN) [4] and Semantics of Business Vocabulary and Business Rules (SBVR) [5]. We select BPMN because of its expressive representation ability as well as industry support. We select SBVR because it defines the vocabulary and rules for documenting business facts and business rules, and, because in combination with BPMN it provides the highest level of representational power [3].

In the remainder of this paper, we first present a brief exploration of related topics. We then present the proposed integration framework and its constituent methods. These methods have been deployed in a proof of concept tool suite, using BPMN and SBVR.

2 Related Work

The two predominant approaches for modeling organizational policies and procedures, namely business process modeling and business rule modeling [6], have been the focus of much research over the last few decades. This resulted in many proposals of modeling notations. For example, FlowMake [7], YAWL [8], and BPMN 2.0 [4] for business process modeling, to name a few; and E-C-A Based Business Rules [9], AgFlow [10], RuleSpeak [11] and SBVR [5] for business rule modeling, among others.

Since the introduction of rule modeling languages, researchers have considered the integration of process and rule models. [12] suggested business rule modeling should be merged with business process modeling to help with the temporal information control for information systems development. [9] proposed that ECA rules should be used to help integrate different process modeling and workflow languages. [13] proposed an initial procedure model for integrated process and rule modeling. All of these papers propose top-down approaches for integration and suggest that the integration should happen from the design stage. While we agree with this view, in reality organisations may already have existing separate process and rule models. Along this argument,[14] proposed an annotation approach to apply the compliance rules to business process modeling. [15] incorporated control objectives rules into process model via Formal Contract Language. [16] further supported the possibility of integration by proposing a methodology to translate rule-base business contract constraints into a business process.

Overall, while some research has been integrated business process modeling and business rule modeling, there are no guidelines to integrate business processes and

generic business rules from implementation level (bottom-up) and handle the existing model repositories and rule bases.

3 Integration Framework

Figure 1 depicts the two approaches discussed above, *viz*. top-down and bottom-up. The importance of an approach that provides consistency across process model and rule description by design cannot be underestimated. In Figure 1, the left side indicates the top-down approach of [13]. However, most organisations already have large process model repositories as well as rule bases that have been developed historically and/or may have been inherited during acquisitions and mergers. As such, the main problem that organizations are faced with is how to integrate existing process models and rules to ensure that business requirements are consistent across the two and that the consistency can be sustained. Accordingly, the right side of Figure 1 depicts the bottom-up approach.

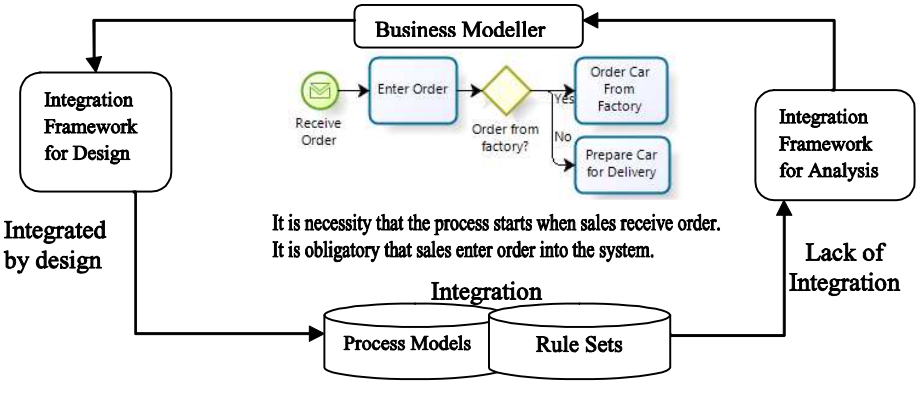

Fig. 1. Integration Framework for Design and Analysis

The bottom-up integration framework is built around a collection of mapping methods that provide distinct ways in which overlap and consistency (or lack of) between processes and rules can be studied. We observe two main aspects of the integration framework that can be found in most integration problems, namely, semantic and structural aspects:

Semantic Aspects: A prelude to studying the structural overlap/differences between process models and rule descriptions is to first identify and homogenize the semantics of the various elements used in the two models. For BPMN these include names of activities, roles, events, gateways, pools and lanes. For SBVR these include names, terms, verbs and keywords. It is clear that this is no trivial task and utilization of computational methods to overcome this challenge is incumbent upon the availability of a reference ontology that can provide the necessary relationships between common terms. In addition to establishing an agreement on the modeling labels, the semantic

aspects can also include issues relating to business goals, where there may be differences in the perceived goals of a process model and a rule set. In this paper, we do not consider the semantic aspects of the integration framework. Significant works that relate to term linkage and ontology development are already in existence [17-21].

Structural Aspects: It is clear that business and rule modeling notations have inherent structural differences. In order to study where these differences arise, we have identified and developed some key methods (see next section). These methods are not exhaustive but provide an initial set necessary to study the structural overlap and are demonstrated on BPMN and SBVR in this paper.

4 Approach

This research follows a Design Science [22] approach, where the artefact under development is the integration framework as well as the integration methods that are part of the framework. The notation agnostic framework was provided in previous section. In this section, we will present the integration method.

To begin the development of these methods an equivalence relationship had to be established between our two chosen notations, viz. BPMN and SBVR (see following section and Appendix 1) before any methods were developed. This process involved a construct-by-construct analysis of each of the two notations to determine where equivalences exist (details of which are omitted here due to page limitations). It was conducted by one researcher, discussed in a workshop-type setting, and then validated by another researcher so as to reduce the risk of bias. Once the set of construct equivalences were established, the researchers developed approaches for annotation and transformation of models. The methods then underwent a preliminary evaluation using a sample scenario.

Because BPMN is a visual process modeling approach while SBVR is a text based rule modeling language, a canonical intermediary is needed to bridge the gap between two representations. We adopt the XML Process Definition Language (XPDL) [23] as the intermediate format. XPDL is a format standardized by the Workflow Management Coalition (WfMC) for interchange of business process definitions between different workflow products. We select XPDL as the bridge between the BPMN and SBVR for two reasons. First, it fulfils the transformation from the BPMN graphic presentation into text representation. Second, it is standardized and well supported by many business process modeling tools.

Below we present two methods for mapping the structural aspects of the two representations, namely annotation-based and transformation-based methods. The former allows identifying the overlaps between two modeling approaches, while the latter allows performing the transformation between two modeling approaches. Before the methods are developed, however, it is important to first study the structural property of the respective modeling constructs. As per the methodology discussed above, Appendix 1 summarises the construct equivalence between the two chosen languages and use this equivalence in the subsequent method presentation.

The constructs presented have been extracted from the respective specifications of BPMN and SBVR. SBVR consists of Name, Term, Verb, Fact Type and Keywords [5]. Since Fact Type can be composed by Terms and Verbs, we remove Fact Type

from the SBVR constructs list. The keywords are further divided into Quantification keywords, Logical keywords, Modal keywords and other keywords, so the Business Rule can be composed from Name, Term, Verb and four different types of keywords.

In terms of BPMN constructs, [24] identified the most often used BPMN constructs. [25] further divided these common used BPMN into four groups: BPMN Common Core, BPMN Extended Core, BPMN Specialist Set and BPMN Overhead We present the BPMN-SBVR mapping through a mapping table provided in Appendix 1. This table focuses on the most common BPMN constructs as identified in above research, namely Common and Extended Core.

In Appendix 1, the first column depicts the BPMN constructs. The second column is the XML tag used in XPDL for that corresponding BPMN construct. The last column depicts the instance of the corresponding SBVR SE construct and associated pattern used. This table is used to map BPMN constructs, XPDL elements and SBVR SE instances (or patterns) during annotation and transformation processes. During the annotation process, only three kinds of BPMN constructs can be annotated, which are Activities, Events and Gateways, thus annotation will only take place for the corresponding XML tags and the rest will be ignored. The annotatable constructs will be compared with SBVR statements to find a match, while non-annotatable constructs will bypass the matching process. During the transformation process, for each BPMN construct, either the corresponding SBVR SE instance or pattern is found. Then one or more connecting BPMN constructs can be automatically transformed into SBVR SE statement by organising the corresponding SBVR SE instance and pattern in order.

Connecting Objects are implicitly translated into SBVR SE constructs depending on the objects they are connected to. Artefacts Objects, Annotation and Group are used to visually organize the BPMN diagram and they are not really part of the process so they are not mapped to SBVR SE constructs. In contrast, SBVR SE modal keyword does not have straightforward corresponding BPMN construct. BPMN does not have moderation feature. BPMN is designed to express either the “happy path”, which means the way it should be to get a job finished; or the “backup path”, which means in case of problem, what should happen to continue the work. Either way, the pattern is used there is “If <condition> then <consequence>”. In other words, under the <condition>, it must be <consequence> and the modal keyword here is always “must”. At this stage, part from “must” or “It is obligatory”, there is no way to present other modal operation in BPMN. We consider this as a limitation of this work.

Pools and Lanes are two different graphic representations to describe the same concept – participants. During the transformation process, both Pools and Lanes are mapped to the same SBVR SE element <participant>. In BPMN, Pools and Lanes are both containers for entities. However, within XPDL, Pools and Lanes are used as containers only to present graphic information and they do not claim the ownership of inside entities, such as activities, events and gateways. Although virtually, events and gateways are organized by participant, the participant-entity relationship is carried by the coordination information not by the ownership information. Task is the only entity which can carry participant information. To associate a task with the corresponding Pool or Lane, a pair of elements (<participant> and <performer>) are used. For each <task>, there is a hidden element <performer>, which contains the ownership information. For instance, if <task> is performed within a <lane>, the <task> must have a <performer> attribute which matches to the <participant> of that <lane>.

We now present the procedures for the mapping methods. These procedures are further demonstrated in the next section through a scenario implemented in a proof of concept tool.

4.1 Annotation Based Mapping

In BPMN, Annotation can only be associated with Flow Objects, which including Events, Activities and Gateways [4]. The objective of this annotation method is to find out which BPMN constructs are indeed used for SBVR annotation and where the access points are in BPMN diagram to insert relative SBVR SE statement as annotation. The annotation process includes the following procedures:

Segmentation of SBVR SE statement: SBVR SE statements are composed of Name, Term, Verb and Keywords. Proper segmentation helps to find the right access point in BPMN. For example, the statement "It is a necessity that the process starts when sales receive order" is made up of the following segments: "It is a necessity that" (modal keyword), "process, sales" (term) and "starts, receive order" (verb).

After getting a list of segments from a statement, analyses need to be conducted to find the point of entry in BPMN. Using the example from last bullet point, all modal keywords are not mapped into BPMN and can be ignored. Regarding to other keyword, apart from "before/after" which can provide temporal information to help find BPMN point of entry, the rest of them can be ignored. "process" refers to the BPMN diagram not a particular BPMN element. So the segments left are "sales, starts and receive order". After filtering, the point of entry for this statement in BPMN diagram is "message start event" "receive order". An SBVR SE statement may have more than one point of entry in a BPMN diagram. It is acceptable to attach the statement to any of the BPMN elements, since the objective of this method is to find out the overlaps and differences not the best access point.

Note that "before" and "after" keywords can be used to indicate the temporal information. For simplicity, in this paper, all the SBVR statements only use keyword "after". For example, this format will be used "Its necessity that operation A happens after operation B." Using this convention, if the there are more than one activities are found in a SBVR statement, we will assure the one comes first will be what we are looking for and the others will be used to express temporal information. In other word, the statement will be attached to the activity that appears first.

```
Algorithm 1: BPMN annotation
Input: BPMN diagram (XPDL file) and SBVR statements (text format)
Output: Annotated BPMN diagram (XPDL format)
Define ActivitySet A, StatementSet S
For each statement in S
    Let SmallestActivityIndex = -1, ActivityName = ""
    For each activity in A
        Let ActivityIndex = statement.indexOf(activity)
        If( ActivityIndex > 0 and smallestActivityIndex = -1), then
            smallestActivityIndex = ActivityIndex; and ActivityName = activity
        If( ActivityIndex > 0 and ActivityIndex < smallestActivityIndex), then
            smallestActivityIndex = ActivityIndex; and ActivityName = activity
    If (smallestActivityIndex > -1) then
        attach statement to Activity where Activity.name = AcitivityName.
```

4.2 Transformation Based Mapping

Based on appendix 1, the following procedure is used to transform BPMN (in XPDL) format into SBVR SE statements.

```
Algorithm 2: BPMN to SBVR SE transformation
Input: BPMN diagram (XPDL file)
Output: SBVR statements (text format)
Foreach activity in XPDL
   If activity.type = StartEvent
      Then output statement: "process starts on the condition of  " +
   StartEvent.type
   If activity.type = EndEvent
      Then output statement: "process ends on the condition of " + EndEvent.type
   If activity.type = Gateway
      Then output statement: pre-condition + gateway.type + post-condition
   If activity.type = Task
      Then output statement
activity.perfomer + "perform task" + activity.name + "after" + activity.source.
activity.perfomer + "send message" + activity.message.name + "to" +
   activity.message.target
Foreach pool in XPDL
   If pool.message exist
      Then Output statement pool.name + "send message" + pool.message.name + "to" +
pool.message.target.
```

As discussed in previous section, only "must" (or "It's obligatory that") modal keyword can be used in BPMN, thus we make all the statements obligatory.

5 Method Demonstration[1]

Consider a Car Sales process, there are six parties involved: Customer, Sales person, Preparation person, Finance person, Factory and Lender. These six parties need to interact with each other to finish the process, from car order to car delivery. A BPMN model for the scenario is shown in Appendix 2, while SBVR rules are presented below:

1. It is necessity that the process starts when sales receive order.
2. It is obligatory that sales enter order into the system.
3. It is obligatory that the finance people arrange finance for the customer after sales enter order.
4. It is necessity that the sales check if they need order car from factory after they enter order.

[1] We have developed a proof of concept prototype tool to implement the mapping methods presented in this paper. The tool is developed using Java. It takes BPMN diagram in XPDL format, and automatically annotates and/or transforms it with/into SBVR SE. JDom was used as the XML parser. Below we provide the results obtained from the tool of the implemented methods based on the above scenario.

5. It is possible that the sales send factory message to check ship date.
6. It is possible that factory sends message to provide ship date.
7. It is obligatory that the sales confirm the ship date with the customer.
8. It is possible that if car is unavailable, the sales terminate the process.
9. It is necessity that preparation people prepare car for delivery after the sales confirm the ship date with the customer.
10. It is necessity that preparation people prepare car for delivery if no need to order car from factory.
11. It is obligatory that the finance people communicate with lender if finance is required.
12. It is possible that the process is terminated if finance is unavailable.
13. It is obligatory that finance people need to wait for both "Arrange Finance" and "Prepare car for delivery" ready to close the order and deliver the car.
14. It is obligatory that the sales notify the customer when deliver the car with temporary registration.
15. It is obligatory that finance people check customer's credit before arrange finance.
16. It is necessity that finance people report to their manager after close and deliver the new car.
17. It is necessity that preparation people apply for temporary registration number after prepare car for delivery.

Applying the annotation process to the initial BPMN diagram, the annotated BPMN diagram is generated. By observing the Annotated BPMN diagram, we notice the following points. Firstly the SBVR SE statements for message flow are not annotated. Although in theory, these statements can be attached to the message flow in BPMN diagram, according to the BPMN specification, message flows cannot have annotation. Secondly, some of the SBVR SE statements used to describe this car sales scenario are not covered in the BPMN representation. However, our tool picked them up and annotated to the BPMN (the annotation shown in green colour). This shows that our method has the capability to find the missing spots in BPMN diagram.

With the transformation method presented in previous section, the following SBVR SE statements are automatically generated.

1. It is obligatory that Customer send message to start process.
2. It is obligatory that Customer send message Confirmation response to Sales.
3. It is obligatory that Factory send message Ship Date to Sales.
4. It is obligatory that Lender send message Loan response to Finance People.
5. It is obligatory that after close and deliver, finish the process, send message Deliver vehicle and temporary registration to Customer.
6. It is obligatory that Sales send message Confirmation request to Customer.
7. It is obligatory that Sales send message Factory order to Factory.
8. It is obligatory that Finance People send message Loan request to Lender.
9. It is obligatory that Sales perform task enter order after receive order.
10. It is obligatory that both if finance is unavailable? equals No and after task prepare car for delivery, execute task close and deliver.
11. It is obligatory that after order car from factory, execute car is unavailable?.
12. It is obligatory that if car is unavailable? equals Yes, terminate all process.

13. It is obligatory that if car is unavailable? equals No, execute task prepare car for delivery.
14. It is obligatory that after arrange finance, execute finance is unavailable?.
15. It is obligatory that if finance is unavailable? equals Yes, terminate all process .
16. It is obligatory that Finance People perform task arrange finance after enter order.
17. It is obligatory that after enter order, execute order from factory?.
18. It is obligatory that if order from factory? equals Yes, execute task order car from factory.
19. It is obligatory that if order from factory? equals No, execute task prepare car for delivery.

Most of the BPMN constructs (appendix 1) have corresponding SBVR constructs or combination of SBVR constructs. The transformation can be performed based on these overlaps without loss of information. However, some differences remain. Firstly, Connecting Objects (Sequence Flow, Message Flow and Association) only exist in BPMN but not in SBVR. Secondly, Artefacts Objects Annotation and Group only exist in BPMN but not in SBVR. Thirdly, although in this paper we do not consider the executable modeling which is new in BPMN 2.0, to our best knowledge, SBVR is not capable to adopt this new feature. Fourthly, very limited moderation operation can be used in BPMN.

6 Conclusion

We have proposed a bottom-up approach to process and rule integration to complement design approaches for integrated modeling of processes and rules. To this end we have developed two mapping methods within an overarching integration framework to support business analysts and modelers in studying the overlaps and differences between processes and rule repositories. The focus of this paper was not to analyse the respective expressive and notational capabilities of BPMN and SBVR. It can be observed that due to the inherent differences, there are a number of constructs (within the structural aspect) that cannot be mapped easily using either of the methods (annotation and transformation) presented above. However, we argue that the proposed methods provide 1) proof of feasibility of integration between process modeling and rule modeling; 2) implementable model which has been partially implemented (semantic aspects are not implemented yet).In the future, we plan to further explore the limitations we discussed in this paper. We will try to refine the method we proposed for annotation and transformation; develop the method to measure the difference between the BPMN and SBVR; add the semantic aspect and explore further the extent to which the automation of integration can be achieved.

References

1. Recker, J., Indulska, M., Rosemann, M., Green, P.: How good is BPMN really? Insights from Theory and Practice (2006)
2. Indulska, M., Recker, J., Rosemann, M., Green, P.: Business process modeling: Current issues and future challenges. Springer, Heidelberg (2009)

3. zur Muehlen, M., Indulska, M.: Modeling languages for business processes and business rules: A representational analysis. Information Systems 35(4), 379–390 (2010)
4. OMG. Business Process Model and Notation 2.0 Beta 1 Specification (2009), http://www.omg.org/cgi-bin/doc?dtc/09-08-14 (cited June 1, 2010)
5. OMG. Semantics of Business Vocabulary and Business Rules (SBVR), v1.0 (2008), http://www.omg.org/spec/SBVR/1.0/PDF (cited June 2, 2010)
6. Lu, R., Sadiq, S.: A survey of comparative business process modeling approaches. Springer, Heidelberg (2007)
7. Sadiq, W., Orlowska, M.: On capturing process requirements of workflow based business information systems. In: 3rd International Conference on Business Information Systems, Poznan, Poland (1999)
8. Van Der Aalst, W., Ter Hofstede, A.: YAWL: yet another workflow language. Information Systems 30(4), 245–275 (2005)
9. Knolmayer, G., Endl, R., Pfahrer, M.: Modeling processes and workflows by business rules. In: Business Process Management, pp. 201–245 (2000)
10. Zeng, L., Ngu, A., Benatallah, B., O'Dell, M.: An agent-based approach for supporting cross-enterprise workflows. IEEE Computer Society, Los Alamitos (2001)
11. Ross, R., Lam, G.: RuleSpeak Sentence Templates: Developing Rules Statements Using Sentence Patterns. Business Rule Solutions (2001), http://www.BRCommunity.com
12. Krogstie, J., McBrien, P., Owens, R., Seltveit, A.: Information systems development using a combination of process and rule based approaches. Springer, Heidelberg (1991)
13. zur Muehlen, M., Indulska, M., Kittel, K.: Towards integrated modeling of business processes and business rules (2008)
14. Governatori, G., Hoffmann, J., Sadiq, S., Weber, I.: Detecting regulatory compliance for business process models through semantic annotations. Springer, Heidelberg (2009)
15. Sadiq, W., Governatori, G., Namiri, K.: Modeling control objectives for business process compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 149–164. Springer, Heidelberg (2007)
16. Milosevic, Z., Sadiq, S., Orlowska, M.: Translating business contract into compliant business processes (2006)
17. Nirenburg, S., Raskin, V.: Ontological semantics. MIT Press, Boston (2004)
18. Maedche, A., Staab, S.: Measuring similarity between ontologies. In: Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web, pp. 15–21 (2002)
19. Giunchiglia, F., Shvaiko, P.: Semantic matching. The Knowledge Engineering Review 18(03), 265–280 (2004)
20. Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., Fienberg, S.: Adaptive name matching in information integration. IEEE Intelligent Systems 18(5), 16–23 (2003)
21. Ullman, J.: Information integration using logical views. In: Afrati, F.N., Kolaitis, P.G. (eds.) ICDT 1997. LNCS, vol. 1186, pp. 19–40. Springer, Heidelberg (1996)
22. Hevner, A., March, S., Park, J., Ram, S.: Design science in information systems research. Mis Quarterly 28(1), 75–105 (2004)
23. WfMC. XML Process Definition Language Specification Version 2.1 (2008), http://www.wfmc.org/xpdl.html (cited April 3, 2010)
24. Muehlen, M., Recker, J.: How much language is enough? Theoretical and practical use of the business process modeling notation. Springer, Heidelberg (2008)
25. OPAALS, Automatic Code Structure and Workflow Generation from Models, OPAALS Project, European Community (2008)

Appendix 1. Proposed Mapping between BPMN, XPDL and SBVR SE

BPMN Construct	XPDL Element	SBVR SE Instance or Pattern
Pool and Lane	<Participant/>	<participant placeholder>[1]
Task	<Task/>	<task placeholder>[1]
Data-Based Exclusive Gateway	<Route/>	After <antecedent 1> or <antecedent 2> or…<antecedent N> then <consequent> with N arbitrary (as merge) After <antecedent> if <condition I> then <consequent I> with I generic index from 1 to arbitrary N(as split)
Event-Based Exclusive Gateway	<Route GatewayType="EventBasedXOR" />	After <antecedent 1> or <antecedent 2> or…<antecedent N> then <consequent> with N arbitrary. (as merge)[2] After <antecedent> if <condition i> then <consequent i> with I generic index from 1 to arbitrary n(as split)[2]
Parallel Gateway	<Route GatewayType="AND" />	After <antecedent 1> and <antecedent 2> and … <antecedent N> then <consequent> with N arbitrary (as merge) After <antecedent> then <consequent 1> and <consequent 2>… and <consequent n> (as split)
Inclusive Gateway	<Route GatewayType="OR" />	After <antecedent 1> then < consequent> and <antecedent 2> then <consequent> … <antecedent N> then <consequent> (as merge)[3] After <antecedent> if <condition 1> then <consequence 1>, if <condition 2> then <consequence 2>… if <condition n> then <consequence n> (as split)[3]
None Start Event	<StartEvent Trigger="None" />	Start <task placeholder>
Message Start Event	<StartEvent Trigger="Message" />	After <message placeholder> is received, start <task placeholder>
Timer Start Event	<StartEvent Trigger="Timer" />	After <time placeholder>, start <task placeholder>
Conditional Start Event	<StartEvent Trigger="Conditional" />	If <condition>, then start <task placeholder>
Signal Start Event	<StartEvent Trigger="Signal" />	After <signal placeholder> received, start <task placeholder>
None End Event	<EndEvent />	End the process
Message End Event	<EndEvent Result="Message" />	After end of the process, send <message placeholder> to <participant placeholder>
Signal End Event	<EndEvent Result="Signal" />	After end of the process, broadcast <signal placeholder>
Cancel End Event	<EndEvent Result="Cancel" />	Cancel the process
Error End Event	<EndEvent Result="Error" />	On <error placeholder>, end the process.
Terminate End Event	<EndEvent Result="Terminate" />	Terminate all the process.
Data Object	<Artifact ArtifactType="DataObject" />	<file placeholder>

Note: 1. In pattern <participant placeholder><verbPhrase><Task Placeholder>, Pool and Lane's names will replace the <participant placeholder> and Task's name will replace the <Task Placeholder>. 2. Compare to Inclusive Gateway, only one <antecedent> <consequent> pair will happen. 3. Compare to exclusive gateway, more than one <condition> <consequence> pairs can happen. The <antecedent> may vary, but the <consequence> always be the same.

Appendix 2. Annotated Car Sales BPMN Diagram