Validation Rules for Exporting Business Process Diagram to Business Process Execution Language

Parnu Sooksanan
BPEL Solution Manager
No Magic Asia, Thailand

Siripong Roongruangsuwan
Enterprise Solution Manager
No Magic Asia, Thailand

Abstract

The Business Process Modeling Notation (or BPMN) is a set of readily understandable notations used for all business analysts/users. The BPD diagram is used for designing the business process with a set of BPMN notations. The Business Process Execution Language for Web Services, called “BPEL” or “BPEL4WS” is a language used for the definition and execution of business processes using Web Services. As we investigate the validation rules for exporting BPMN elements to BPEL files, we have found that there are some verification and validation rules needed in order to ensure that the converted BPEL files can be manipulated and executed respectively. This paper reviews the current validation rules for the BPD diagrams and shows the verification & validation rules for exporting BPMN to BPEL. Additionally, in this paper, No Magic Inc.’s MagicDraw Version 10.5 is used for the business process modeling tool and BEA WebLogic Integration (WLI) Version 8.1 is also used for business process execution respectively. As a result, this paper reveals that those rules for exporting feature are useful for all business analysts/users who want to convert the BPD diagram to BPEL.

1. Introduction

No Magic Inc.’s MagicDraw since Version 10.0 provides valuable BPD diagrams, which follow the BPMN 1.0 specification under “The Business Process Management Initiative (BPMI)” [3]. Differed from other UML-diagrams such as an Activity Diagram or a Use Case Diagram, BPD diagram allows technical staffs to easily understand and implement business diagrams. The Business Process Execution Language (BPEL) is an XML schema language for specifying the automated business processes bidirectionally between multiple enterprises using Web Service technology [10]. MagicDraw Version 10.5 will provide the functionality for users to export the BPD diagram to the BPEL4WS based on BPEL specification v.1.1 [1, 2]. However, after exporting BPMN notations from BPD diagram to the BPEL files, some additional process data and Web Services information are required to be manipulated in order to deploy the BPEL files on the vendor-specific servers. In this paper, we had an experiment of exporting BPMN notations to the BPEL files. The experiment reveals that there are some open issues for converting the BPMN notations to the BPEL files with a flavor of BEA. Conversion from BPD diagrams to the BPEL files is the significant part of the process. In order to successfully import the BPEL files to BEA Web Logic Workshop (WLW), the proper BPEL file must follow the known limitations of BEA’s BPEL Import Tool. Moreover, the BPD diagram itself has some limitations such as Ambiguous Issues, Open Issues, and Valid/Invalid BPD diagram of converting to
BPEL [6]. With these limitations, the verification and validation rules for exporting BPD diagrams to the BPEL files are established.

2. Literature Review

2.1 Related Works

The Business Process Management Initiative (BPMI) has its own limitations and open issues [6]. Several notations in BPMN can’t be mapped to BPEL tags. The list of unmappable notations in accordance with the BPMN version 1.0 specification [6] is as follows:

- **Start Event with a Trigger that is None** - A receive or a pick must be present at the start of process and locate within a sequence or a flow. Also, a createInstance must set to “yes”.
- **End Event with a Result that is None** - End Event with a Result that is None will not be mapped, but it can be used to define the end of a path in any branch of the process.
- **Cancel End Event** - Cancel End Event is still an open issue for mapping.
- **Intermediate Event with a Trigger that is None** - The purpose of Intermediate Event with a Trigger that is None is usually for showing the status of the process.
- **Cancel Intermediate Event** - Cancel Intermediate Event is still an open issue for mapping.
- **Sub-Process set to a Transaction** - A Sub-Process set to a Transaction is still an open issue for mapping.
- **Embedded Sub-Process Ad hoc** - An Embedded Sub-Process set to an Ad hoc can’t be mapped to BPEL tag because it is not executable.
- **Reference Sub-Process** - A Reference Sub-Process can’t be mapped to BPEL tag because BPEL tag doesn’t provide the referencing.
- **Reference Task** - A Reference Task can’t be mapped to BPEL tag because BPEL tag doesn’t provide the referencing.
- **Gateway Complex** - Gateway Complex is still an open issue for mapping.
- **Pool, Lane, Association** - Pool, Lane, and Association can’t be mapped to BPEL tag because they are not executable.
- **Artifacts** - Artifacts are only used for giving information to user, but they are not mapped to BPEL tag.
- **Message Flow** - Message Flow can’t be mapped to BPEL tag. It uses for sending message among Pools.

There are still many open issues and questions about implementation. Each vendor has made their own decision on specific implementation on the open issues and questions [5]. BEA is the chosen vendor of this paradigm. This paper will focus on the BEA WLW and BEA WLI; therefore the limitations and open issues from BEA are concerned. The BEA’s BPEL Import Utility Plug-in also has the limitations and issues as well. The limitations and issues provided by BEA are listed as follows: [4]

- **Compensation Handlers, Global eventHandler, until attribute, link and** will be ignored but the conversion of BPEL file will continue. Moreover, the activities related to the link will be ignored as well.
- **If there is a flow in the first sub-component in BPEL file, the particular BPEL can’t be converted.**
- **The only one reply activity is allowed in the BPEL file.**
- **A dummy Timer will be generated, if there is the scope level eventHandlers in the on Message in the BPEL file.**
- **If there are more than one partnerlinks, only one partnerlink**
will be converted to \textit{clientRequestWithReturn}. Others will be converted to an \textit{asynchronous} interface.

- The default portType will be assigned if the binding and service information for any \textit{portTypes} are missing.

**2.2 Problem Statement**

From the related works above, the major problems of exporting BPMN notations to the BPEL file based on BPMN 1.0 specification and BPEL4WS 1.1 specification are:

- BPEL4WS 1.1 specification has the portability problem on different vendors.
- In order to export BPD diagram to BPEL file, there is an issue to verify and validate the BPD diagram before conversion.

**3. Proposed Solution**

**3.1 Proposed Validation Rules**

To solve those problems above, the MagicDraw Version 10.5’s validation rules of exporting the BPD diagram into the BPEL file are established. Those rules will verify and validate the BPD diagram so the diagram will comply with both of BPMN limitations and BEA’s BPEL Import Tool. The BPEL files, which correspond with the proposed validation rules, will be imported successfully into BEA WLW and will be deployable and executable on BEA WLI. Along with those two main limitations, the compatibility and appropriate constraints given from No Magic Asia are inserted into the validation rules. Therefore, the proposed validation rules are the combination of BPMN semantics/limitations, known limitations of BEA’s BPEL Import Tool and some additional constraints established by No Magic Asia. The validation rules can be described as follow:

1. Note, Note HTML, Text Box, Text Box HTML, Anchor to Note, Dependency, Constraint, Separator, RectangularShape and Group will be ignored and not be mapped.
3. A process level must have one and only one Start Event notation.
4. At least one End Event notation must be presented in the diagram.
5. Multiple End Events are allowed in the diagram.
6. All notations, except Start Event must have the incoming Sequence Flow.
7. Start Event notation must be a source of Sequence Flow. It means Start Event notation must have an outgoing Sequence Flow.
8. Every branch of process must have the End Event notation.
   - In Expanded Sub-Process may use End Event None as the ending point.
9. All notations, except End Event, must have the outgoing Sequence Flow.
10. Intermediate Event notation must be between Start Event and End Event notations.
11. Task and Sub-Process can’t have any Event attached to them.
12. Only Message, Timer, Error, Compensation, and Rule Intermediate Event can be used in normal flow.
13. Intermediate Message, Timer, Error, Compensation, and Rule must have only one incoming Flow.
14. Intermediate Event must have only one outgoing Sequence Flow.
15. Intermediate Link will not be mapped because the methodology of mapping needs to track the flow to element. Therefore, if the Intermediate Link Event doesn’t have the flow to it, it will not be recognized.
16. If a Gateway has only one incoming and one outgoing Sequence Flow, this gateway is not needed.
17. Gateway can’t have both multiple incoming Sequence Flows and also multiple outgoing Sequence Flows. If it has multiple incoming Sequence Flows, it must have only one outgoing Sequence Flow. If it has multiple outgoing Sequence Flows, it must have only one incoming Sequence Flow.
18. If there is only one incoming Sequence Flow to DataBased XOR Gateway and more than one outgoing Sequence Flow, DataBased XOR Gateway will act as a convergence. DataBased XOR Gateway must have one outgoing Default Sequence Flow and at least one outgoing Condition Sequence Flow with condition expression. The normal Sequence Flow can’t be an outgoing Flow from DataBased XOR, which acts as a convergence.
19. If there are more than one incoming Sequence Flow to DataBased XOR Gateway and only one outgoing Sequence Flow, DataBased XOR Gateway will act as a divergence. The only outgoing Sequence Flow from DataBased XOR Gateway must be a normal Sequence Flow not Condition Sequence Flow or Default Sequence Flow.
20. For EventBased XOR Gateway, the outgoing Sequence Flow can target only Task, which TaskType set to Receive or an Intermediate Event with Trigger attribute set to Message, Timer or Rule.
21. If there is only one incoming Sequence Flow to EventBased XOR Gateway and more than one outgoing Sequence Flow, EventBased XOR Gateway will act as a convergence. The only outgoing Sequence Flows from EventBased XOR Gateway must be normal Sequence Flows not Condition Sequence Flow or Default Sequence Flow.
22. If there are more than one incoming Sequence Flow to EventBased XOR Gateway and only one outgoing Sequence Flow, EventBased XOR Gateway will act as a divergence. The only outgoing Sequence Flow from EventBased XOR Gateway must be a normal Sequence Flow not Condition Sequence Flow or Default Sequence Flow.
23. If the outgoing Sequence Flow targets to a Receive Task, then an Intermediate Event with a Trigger Message can’t be target of the other outgoing Sequence Flow of this EventBased XOR Gateway.
24. If there is only one incoming Sequence Flow to Inclusive OR Gateway and more than one outgoing Sequence Flow, Inclusive OR Gateway will act as a convergence. Inclusive OR Gateway must have one outgoing Default Sequence Flow and at least one outgoing Condition Sequence Flow with condition expression. The normal Sequence Flow can’t be an outgoing Flow from Inclusive OR, which acts as a convergence.
25. If there are more than one incoming Sequence Flow to Inclusive OR Gateway and only one outgoing Sequence Flow, Inclusive OR Gateway will act as a divergence. The only outgoing Sequence Flow from Inclusive OR Gateway must be a normal Sequence Flow not Condition Sequence Flow.
Sequence Flow or Default Sequence Flow.

26. If there is only one incoming Sequence Flow to Parallel AND Gateway and more than one outgoing Sequence Flow Parallel AND Gateway will act as a convergence (Parallel AND Fork). The only outgoing Sequence Flows from Parallel AND Gateway must be normal Sequence Flows not Condition Sequence Flow or Default Sequence Flow.

27. If there are more than one incoming Sequence Flow to Parallel AND Gateway and only one outgoing Sequence Flow, Parallel AND Gateway will act as a divergence (Parallel AND Join). The only outgoing Sequence Flow from Parallel AND Gateway must be a normal Sequence Flow not Condition Sequence Flow or Default Sequence Flow.

28. Task notations must have only one incoming Sequence Flow. In case that there are multiple incoming Sequence Flow for Task, the Gateway – convergence is recommended to use instead.

29. A diagram (process) inside the Expanded Sub-Process must follow the same rules as the outer diagram (process).
   - Exception to this is the End Event None might be used in the Expanded Sub-Process
   - All notations inside the Expanded Sub-Process can’t have incoming Sequence Flow and outgoing Sequence Flow linked directly to other notations outside the Expanded Sub-Process.

30. Collapsed Sub-Process isn’t allowed to have any Event notations attached to its boundary.

31. Isolate notation is prohibited in the diagram. It means any notations without any incoming and outgoing flow are not allowed in the diagram.

32. All notations, except Gateway, must not have more than one incoming Sequence Flow.

33. Default Sequence Flow and Conditional Sequence Flow are allowed to use with only Gateway notation.

3.2 Example of BPD Diagram

In Figure 1: Invalid BPD diagram, there are four errors occurred as follows:
   - The Start None Notation is used in this diagram.
   - There are multiple incoming Sequence Flows for “Select the best Price” Task Notation, the Gateway – convergence is recommended to be used instead.
   - The Default Sequence Flow is allowed to use with only Gateway notations.
   - The End None Notation is used in this diagram.

Figure 1: Invalid BPD diagram

Figure 2: Valid BPD diagram
In Figure 2: Valid BPD diagram, the diagram is the correction of diagram in Figure 1: Invalid BPD diagram
- Start Message Notation replaces the Start None Notation.
- The DataBased XOR – convergence is used to merge the Sequence Flows before “Select the best Price” Task Notation.
- The Default Sequence Flow is removed.
- End Message Notation replaces the End None Notation.

4. Conclusion

The BPD diagram is an effective tool that helps the business analysts/users to express their business processes into the way that is readily understandable and functional for the technical staffs. To optimize the valuation of BPD, converting BPD to the execution form called “BPEL“, which can use the benefit of Web Service, is quite a challenge. However, BPEL specification version 1.1 is still not a widely accepted standard. Therefore, it is a very rare possibility to accomplish portability of the complex BPEL on different vendors. The BPEL file, which passes these validation rules, is likely to be able to deploy and execute on BEA WLI. However, these limitations may delimit user to generate a desirable diagram. The root of matter is because BPEL 1.1 has many open issues. The robust portable specification for BPEL called Web Services - Business Process Execution Language 2.0 (WS-BPEL 2.0) hopefully will eliminate as many open issues as possible and make BPEL actually portable.

References