Transforming BPMN Diagrams into YAWL Nets

Gero Decker[1], Remco Dijkman[2], Marlon Dumas[3], and Luciano García-Bañuelos[4,⋆]

[1] Hasso Plattner Institute, Germany
gero.decker@hpi.uni-potsdam.de
[2] Eindhoven University of Technology, The Netherlands
r.m.dijkman@tue.nl
[3] University of Tartu, Estonia
marlon.dumas@ut.ee
[4] Universidad Autónoma de Tlaxcala, Mexico
lgbanuelos@gmail.com

Abstract. While the Business Process Modeling Notation (BPMN) is the de facto standard for modeling business processes on a conceptual level, YAWL allows the specification of executable workflow models. A transformation between these two languages enables the integration of different levels of abstraction in process modeling. This paper discusses the transformation of BPMN diagrams to YAWL nets and presents a tool that carries out this transformation.

1 Introduction

Process modeling occurs at different levels of abstraction. First, models serve to communicate as-is business processes, pinpoint improvement options, conduct resource and cost analysis and to capture to-be processes. The Business Process Modeling Notation (BPMN [1]) is the de facto standard for process modeling at this level. On the other hand we find languages that are targeted at technically realizing business processes, used as input for process execution engines. The Business Process Execution Language (BPEL) is a standard for implementing process-oriented compositions of web services. YAWL [2] is an alternative to BPEL, with a strictly defined execution semantics, a first-class concept of "task", and sophisticated support for data mappings and task-to-resource allocation.

While the mapping from BPMN to BPEL has been studied in detail and is implemented by several tools, the mapping from BPMN to YAWL has not yet received attention. At first glance, this mapping may seem straightforward. Indeed, the conceptual mismatch between BPMN and YAWL is not as significant as the one between BPMN and BPEL, especially with regards to control-flow structures. However, mapping BPMN to YAWL turns out to be tricky in the details, revealing subtle differences between the two languages.

The transformation from BPMN to YAWL can be used as an instrument to implement process-oriented applications. It also opens the possibility of reusing

⋆ Funded by CUDI (e-Grov Project) and by ANUIES (ORCHESTRA Project).

M. Dumas, M. Reichert, and M.-C. Shan (Eds.): BPM 2008, LNCS 5240, pp. 386–389, 2008.

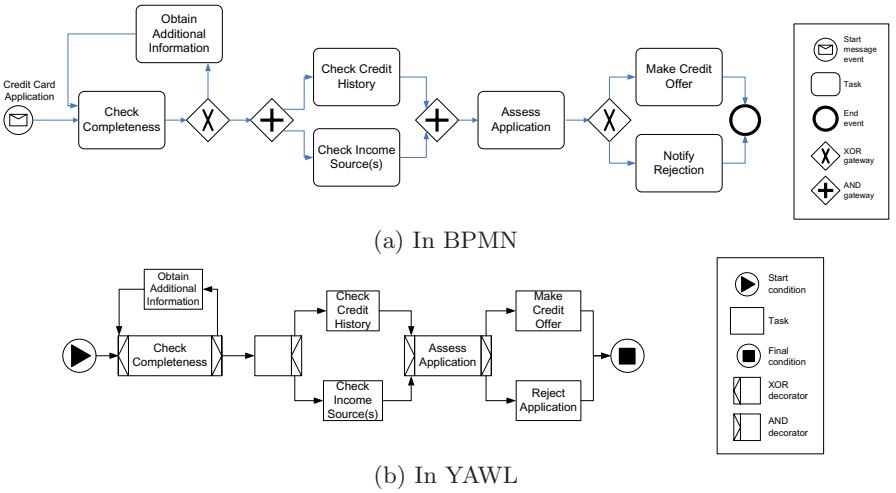
existing static analysis techniques available for YAWL. Like Petri nets, YAWL has a formally defined semantics which enables the analysis of YAWL nets to detect semantic errors such as deadlocks. At the same time, YAWL allows one to capture advanced process modeling constructs that can not always be captured in plain Petri nets, e.g. the OR-join or multi-instance activities.

The next section outlines the mapping from BPMN to YAWL. Section 3 then discusses the tool implementation and Section 4 concludes. The tool is available at: `http://is.tm.tue.nl/staff/rdijkman/bpmn.html`

2 Overview of BPMN to YAWL Transformation

At their core, BPMN and YAWL share several common concepts. In particular, the concept of task in BPMN matches the concept of task in YAWL. Also, the concept of gateway in BPMN matches the concept of decorator in YAWL, and the concept of flow in BPMN matches the same concept in YAWL. As an illustration, Figure 1 shows a simple business process model in BPMN, and the corresponding YAWL net produced by the BPMN2YAWL tool. It can be seen from this simple example that the the join decorator of task "Check Completeness" matches the XOR-merge gateway in BPMN, meaning that the task can be reached from either of its incoming flows. Similarly, the split decorator of this same task matches the XOR-split gateway in the BPMN diagram.

At this level, one key difference between BPMN and YAWL is that YAWL does not provide the ability to directly chain several connectors together, such as an AND-split connector with a branch that leads directly to an XOR-split connector. The BPMN2YAWL tool deals with this mismatch by introducing empty YAWL tasks (i.e. YAWL tasks without decomposition) which serve purely for control routing. In the working example (Figure 1), the AND-split in the



(a) In BPMN

(b) In YAWL

Fig. 1. Simple process model in BPMN and in YAWL

BPMN diagram is mapped to an empty task with an AND decorator. The BPMN2YAWL tool ensures that such empty tasks are only created when it is necessary, so as to minimise the number of empty tasks.

Subprocess tasks in BPMN are mapped to composite tasks in YAWL. In the case where the subprocess in BPMN has an attached event (e.g. an error event), the mapping is more complicated. Figure 2 shows an exception in BPMN and the mapping onto YAWL. In the BPMN diagram, an error 'invalid policy' can occur within the 'insurance check' subprocess. This error is passed to the parent process, which then continues to 'notify customer'. In YAWL a BPMN error is mapped onto a task that sets a subprocess variable (capturing whether or not the error has occurred) to 'true'. The parent process reads this variable upon completion of the subprocess and proceeds according to the value of the variable.

The transformation covers data and resource aspects in addition to control-flow. Properties and assignments in BPMN are mapped to variables, input/output parameters and input/output transformations in YAWL. Lanes in BPMN are mapped to roles in YAWL. Pools are treated as separate business processes (and each one is mapped separately), while message flows are not covered by the mapping since their implementation depends on the communication infrastructure.

The transformation does not cover transactions and compensation handlers because these constructs do not have a direct correspondence in YAWL. Also, these constructs are underspecified in the current BPMN specification. Finally, the transformation does not cover complex gateways.

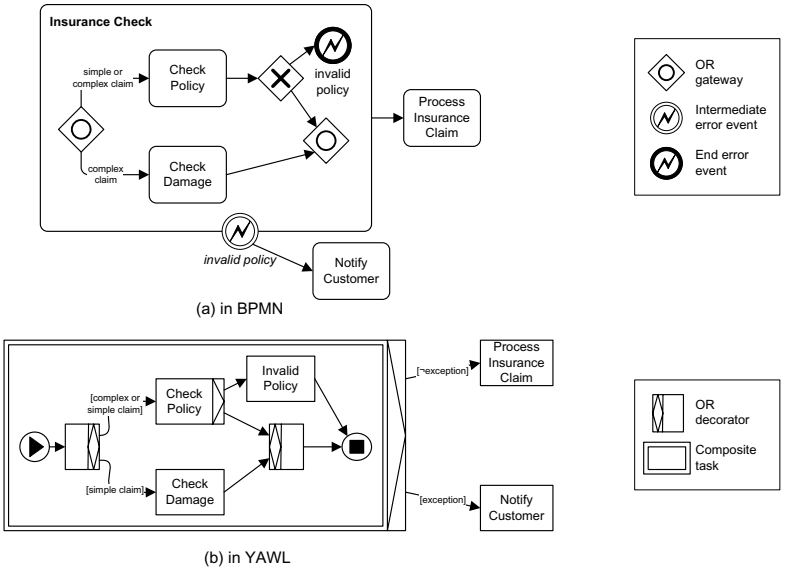


Fig. 2. Mapping attached error events from BPMN to YAWL

3 Tool Implementation

The BPMN2YAWL tool is implemented as an Eclipse plugin. The tool takes as input BPMN diagrams produced by the STP BPMN editor. The models produced by the STP BPMN editor are split in two files: one contains the XMI representation of the model, while the other contains layout information. Once installed, the BPMN2YAWL plugin provides a menu item that allows to transform the XMI file (.bpmn file). It then produces a YAWL engine file that does not contain layout information. This file can be imported into the YAWL editor which applies an automated layout algorithm.

The STP BPMN editor does not support certain features of BPMN. Specifically, it does not support the markers and properties for multi-instance activities and ad hoc activities. To overcome this limitation, the BPMN2YAWL tool is able to detect special types of text annotations: one for multi-instance activities and one for ad hoc activities. The text annotations for multi-instance activities include parameters for specifying minimum and maximum amount of instances to be started, and number of instances that need to complete before proceeding.

4 Outlook

Ongoing work aims at extending the BPMN2YAWL plugin in order to make the transformation reversible. After generating a model, the plugin will be able to propagate changes in the YAWL net into the BPMN diagram (and vice-versa) in order to maintain the models synchronized. For most constructs (e.g. tasks and gateways) the definition of this reversible transformation is straightforward. But when explicit conditions are introduced in the YAWL net, mapping these back to BPMN may prove challenging, or in some cases, impossible. We are investigating under which syntactic restrictions is it possible to preserve the reversibility of the transformation. The aim is that designers are only allowed to alter the YAWL net produced by BPMN2YAWL if the changes can be propagated back to the BPMN diagram. In tandem with this, we plan to incorporate features to visually report differences between process models in BPMN and in YAWL, so that when changes are made to either the source or the target model, the corresponding changes in the other model can be presented to the designer.

References

1. Business Process Modeling Notation, V1.1. Technical report, Object Management Group (OMG) (January 2008), `http://www.omg.org/spec/BPMN/1.1/PDF/`
2. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: yet another workflow language. Inf. Syst. 30(4), 245–275 (2005)