Chapter 1
Making the Business Process Execution Language (BPEL) Flexible

Daniela Wolff
University of Applied Sciences Northwestern Switzerland, Switzerland

Nishant Singh
Indian Institute of Technology, India

ABSTRACT
The Business Process Execution Language (BPEL) is a process modeling language which uses standard control constructs to define a workflow. But, today’s enterprises need to be agile to cope with increasing change, uncertainty and unpredictability. Therefore, automating agile business processes is still a challenge as they are normally knowledge intensive and can be planned to a limited degree. The execution order depends heavily on the case, which has to be performed. So instead of modeling all possible cases and situations which might occur in a knowledge intensive process we introduced an approach which uses semantic technologies and rules. Business rules can be utilized to allow for case-specific adaptation of process steps. A component was developed which allow during run-time rules to automatically detect the state of the case and to determine the necessary process adaptations.

INTRODUCTION
Today’s enterprises need to be responsive to competitors, the market, organizational changes and changing customer requirements and need to response immediately. Agile enterprises are able to scope with increasing changes, uncertainty and unpredictability in the business environment. Although, the original concept of an agile manufacturing was popularized 1991 by the Iacocca Institute (Iacocca 1991), there is still a lack in supporting agility by information systems (Mutschler, Reichert et.al. 2006; Rymer and Moore 2006).

Several workflow management systems exists, which mostly provide process definition tools that presents the knowledge about the process using standard modeling elements like activities, roles or control flows elements (Mendling and Neumann 2005). This is suitable for production oriented processes, but it is ill-suited to deal with changes (van der Aalst and Jablonski 2000). Because, change is difficult, complex and risky according
Making the Business Process Execution Language (BPEL) Flexible

to unintended side effects. Every change of one part of the enterprise has an impact of another part, which leads to the choice, whether to make a change or abandon the competitive benefits of innovation because of the risk (Mitra and Gupta 2008). However, change may range from complete restructuring of the process definition to ad-hoc modification of a single process instance. Especially, knowledge-intensive tasks are performed in a fair degree of uncertainty in which they have to deal with exceptional situations, unforeseeable events and unpredictable situations. If possible at all, covering all possible situations and events, the process model would be highly complex and difficult to manage containing multiple decision points. So, knowledge-intensive processes can be planned to a limited degree (Faustmann 1998). The ability to have more freedom while performing a process instance is called flexibility (Sadiq, Sadiq, and Orlowska 2001).

Knowledge-intensive processes might be structured, but often they are semi-structured or unstructured. Possibly, only parts of a process can be classified as knowledge-intensive (Heravizadeh and Edmond 2008). So, due to performance and efficiency issues it makes sense to combine workflow management approaches with a component which executes the flexible process part.

Service-oriented Architecture (SOA) inherently enables flexibility and adaptivity through choreography of services where each service can select and invoke any other web service. Hence, combining technologies of SOA- and BPM will achieve more flexibility in orchestrating tasks.

The Business Process Execution Language for Web Services (BPEL) supports the specification for coordination and composition of web services (Alonso et.al. 2004). However, by providing predefined control constructs BPEL lacks the flexibility to deal with knowledge-intensive processes. Thus, BPEL is useful for the execution of the structured part of a business process, but it has to be adapted to fulfill the requirements of knowledge-intensive processes.

The execution of knowledge-intensive tasks depends heavily on the case, which has to be performed. For instance, running a credit check in an enterprise is often only done for new customers. Depending on their amount of order several checks might be necessary ranging from requesting only a credit institute to get information about the customer to demanding several department collection services. The condition of an object and how it has to be performed is called its state. The state of an object is given by the values of all of its properties (Mitra and Gupta 2006). To make a process more case sensitive, the case with its property has to be modeled. The state of the case is then during run time used to specify the process flow for the knowledge-intensive process part.

Business rules can be utilized to allow for context-specific adaptation of process steps. During run-time rules allow to automatically detect the state of the case and to determine the necessary process adaptations.

In this chapter, we present an approach where we use BPEL for the execution of the structured part with a component, which is able to perform the knowledge-intensive process part and can adapt the process flow depending on the case. Because the data of case can be variously interpreted by different people and applications, we use ontologies, which serve as communication medium between people and applications and between different applications ontologies are used to specify the case. The Semantic Web Rule Language (SWRL) combines OWL with RuleML, which allows the definition of terms and facts in an ontology. With the use of SWRL and OWL a case sensitive execution of a knowledge intensive task can be supported.

Hence, in the next section, Modeling Approach, we discuss the business rule paradigm and SOA paradigm (basically BPEL). Thereafter, we propose our approach describing modeling of variable processes and how to execute the process model. In the following section, Run Time Approach, we describe the business rule engine RHEA, which