Making BPEL Flexible – Adapting in the Context of Coordination Constraints Using WS-BPEL

Yunzhou Wu
LSDIS Lab, Dept. of Computer Science
University of Georgia, GA 30602
wuyzh@uga.edu

Prashant Doshi
LSDIS Lab, Dept. of Computer Science
University of Georgia, GA 30602
pdoshi@cs.uga.edu

ABSTRACT

Although WS-BPEL is emerging as the prominent language for modeling executable business processes, its support for designing flexible processes is limited. An important need of many adaptive processes is for concurrent activities in the process to respect *coordination constraints*. These require that concurrent activities coordinate their behaviors in response to exogenous events. We show how coordination inducing constraints may be represented in WS-BPEL, and use generalized adaptation and constraint enforcement models to transform the traditional BPEL process to an adaptive one. The final outcome is an executable WS-BPEL process without extensions, able to adapt to events while respecting coordination constraints between activities.

Categories and Subject Descriptors

H.3.5 [**Online Information Services**]: Web Services

General Terms

Algorithms, Theory

Keywords

BPEL, adaptation, coordination, Web services

1. INTRODUCTION

WS-BPEL is emerging as the prominent language for modeling executable business processes. This is because WS-BPEL provides an appropriate set of constructs to design a process in an intuitive way. One of these is the ability to synchronize concurrent flows within a process by specifying *control links* between activities. There could be additional types of constraints between concurrent activities. Consider a trip planner that executes the activities of booking an airline ticket and a hotel concurrently to improve its efficiency. Of course, the date of arrival in the airline ticket must coincide with the date of check-in at the hotel. In the event that the reserved airline ticket becomes unavailable, a new airline reservation with a different date must be *coordinated* with a new hotel booking for that date. In the example, the constraint that airline and hotel booking dates must coincide requires that actions in response to events be coordinated. We call such types of constraints between concurrent activities as *coordination constraints*. These constraints assume significance as processes become flexible and must adapt to events to preserve their optimality. Unfortunately, coordination constraints are not natively supported by WS-BPEL.

We show how coordination constraints may be generally represented in processes specified using WS-BPEL, by utilizing its extensibility constructs. Previous research [3] has shown how concurrent activities participating in coordination constraints may adapt to events while respecting the constraints. From these approaches, we deduce a generalized model of adaptation in conjunction with a constraint enforcement mechanism. WS-BPEL process designers may utilize these generic models for adapting the process while respecting the constraints. We also show how we may transform the process extended with coordination constraints and adaptation models into a core WS-BPEL process that does not utilize the extensibility constructs and is capable of executing on standard WS-BPEL implementations. Karastoyanova et al. [2] extend BPEL to allow for runtime selection of services using 'find and bind'. A manually provided policy is used and constraints on activities are not considered.

Motivating scenario Consider a Web services based process for organizing a trip that consists of concurrently booking an airline ticket, and hotel and rental car at the destination. We consider the event where a reservation of the airline subsequently needs modification because the airline becomes unavailable, perhaps due to overbooking. In response, the planner may choose to change the date of departure, change the destination airport to another one in the city, or simply wait hoping that a vacancy arises. Note that a change in departure date will require coordinated re-bookings of all three – airline, hotel and the rental car. A change in the destination airport will need modification in the booking of the airline and rental car agency concurrently.

2. COORDINATED ADAPTATION

If coordination inducing constraints exist between concurrent activities, we specify them within the <flow> activity in a WS-BPEL document. Analogous to the specification of synchronization constraints, we utilize the <links> construct to list the different coordination constraints. Within this construct, we utilize a new namespace-qualified element <cc_ns:ccLink> to declare each unique coordination constraint. Each constraint is defined as a rule using the syntax of the semantic Web rule language (SWRL).

The decision of how to react to external events becomes difficult in the presence of inviolable constraints between activities. The MDP-CoM approach [3] associates a *service manager* with each activity and formalizes each manager's behavior as an individual decision-making problem. It ensures coordination between the concurrent activities involved in a constraint using a simple coordination enforcement mechanism (CoM) such as a finite state machine.

Given the structural similarity between the decision models for different scenarios [3], it is possible to deduce a general-purpose decision making model for adapting to multiple events. In the model, occurrence of each external event (say $E1$) with a probability ($Pe1$) leads to a state signifying that the event occurred ($Se1$). Some states signify that multiple events have occurred. Coupling the generalized adaptation model with the CoM model of MDP-CoM enables the process to adapt to external events while respecting any coordination constraint, thereby making the process flexible.

3. EMBEDDING ADAPTATION MODELS

Observe that event handlers in WS-BPEL provide a simple way to adapt by performing predefined actions if certain events occur. We use WS-BPEL's extensibility in the form of a new namespace-qualified element, <cc_ns:alt_activity>, to specify a choice of actions that could be performed on receiving an event. Our approach is to allow the designer to reference the generalized adaptation model and the CoM. We also require that the designer indicate which of the concurrent activities and its operations are part of some coordination constraint. This is analogous to specifying if an activity is source or target of a synchronization constraint.

```
< eventHandlers >
  <onEvent  partnerLink="AirlineWSLnk"
    operation="Reserve"
    messageType="ticketUnavail">
    <scope>
      <invoke  name="invoke" partner="AirlineWS"
        portType="tpns:ReservationPT"
        operation="ChangeDate"
        inputVariable="ticketDate">
        <cc_ns:ccLinkInst name="date" />
        <cc_ns:adaptModel ref="uri:genericMdl1"/>
        <cc_ns:comModel ref="uri:CoM1"/>
      </invoke>
      <cc_ns: alt_activity>
        <invoke  name="invoke" partner="AirlineWS"
          portType="tpns:ReservationPT"
          operation="ChangeAirport"
          inputVariable="ticketAirport">
          <cc_ns:ccLinkInst name="airport" />
          <cc_ns:adaptModel ref="uri:genericMdl1"/>
          <cc_ns:comModel ref="uri:CoM1"/>
        </invoke>
      </cc_ns:alt_activity>
    </scope>
  </onEvent>
</eventHandlers>
```

Figure 1: Specifying alternative actions to perform in response to an event and the constraints on these actions.

As the generalized adaptation model is a template, we need to instantiate it for the process. In doing so, we do not assume any knowledge about the operation of the adaptation model and CoM from the process designer. Fortunately, this information could easily be provided in the <eventHandlers> section of the WS-BPEL process as partially shown in Fig. 1. Here, each <onEvent> element specifies an event, say $E1$, and the activities within the nested <scope> are the appropriate adaptive action(s) that could be performed from the corresponding state, $Se1$. Other parameters of the decision model such as the cost of performing an adaptive operation, are often specified in the agreements between the process and the partner Web services using WS-Agreement [1].

4. TRANSFORMING THE EXTENDED BPEL

We outline the steps to transform the extended WS-BPEL process into a fully standards-compatible adaptive process: **1.** We instantiate the generalized decision models and CoMs referenced by the BPEL process. Specifically, we utilize the information about the events and the corresponding action choices (Fig. 1) to establish the structure of the decision model. We utilize the costs of performing the different operations and probabilities of events obtained from the service agreements to parameterize the models. **2.** We combine each decision model specific to an activity with the CoM and formalize the combined model as a Markov decision process (MDP), as shown in [3]. We may then utilize standard solution techniques to solve the MDP. **3.** Solution of the MDP is a policy which maps each combined state (state of the decision model and the CoM) to an action that is optimal at that state. This action is one of the adaptive action choices available to perform. We transform the policy into WS-BPEL code for constructing the final WS-BPEL process. **4.** We generate the adaptive WS-BPEL process using a custom serializing library based on the open source ActiveBPEL library. We utilize the policies to guide the adaptive behavior of each concurrent activity in the process. Coordination is ensured by signaling an internal exception using the <throw> <catch> statements whenever a coordination inducing operation is performed.

5. PERFORMANCE EVALUATION

We compare the rewards gathered when executing a traditional non-adaptive WS-BPEL process (denoted as *originalBPEL*) that responds with a single action to external events, with our transformed adaptive WS-BPEL process (denoted as *adaptiveBPEL*). We varied the probability that the ticket is available after an external event occurs. When the probability of availability is low, *adaptiveBPEL* chooses to change the date in a coordinated manner. In comparison, the *originalBPEL* continues to wait incurring a high cost.

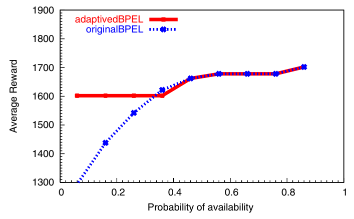

Figure 2: Performances of the traditional and adaptive WS-BPEL processes for the trip planner problem.

6. REFERENCES

[1] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu. *WS-Agreement Specification*, 2005.

[2] D. Karastoyanova, A. Houspanossian, M. Cilia, F. Leymann, and A. Buchmann. Extending bpel for run-time adaptability. In *EDOC*, pages 15–26, 2005.

[3] K. Verma, P. Doshi, K. Gomadam, J. Miller, and A. Sheth. Optimal adaptation in web processes with coordination constraints. In *ICWS*, pages 257–264, 2006.