Chapter 24
DSCWeaver:
Synchronization–Constraint Aspect Extension to Procedural Process Specification Languages

Qinyi Wu
Georgia Institute of Technology, USA

Calton Pu
Georgia Institute of Technology, USA

Akhil Sahai
HP Labs, USA

Roger Barga
Microsoft Research, USA

ABSTRACT

Correct synchronization among activities is critical in a business process. Current process languages such as BPEL specify the control flow of processes procedurally, which can lead to inflexible and tangled code for managing a crosscutting aspect—synchronization constraints that define permissible sequences of execution for activities. In this article, we present DSCWeaver, a tool that enables a synchronization-aspect extension to procedural languages. It uses DSCL (directed-acyclic-graph synchronization constraint language) to achieve three desirable properties for synchronization modeling: fine granularity, declarative syntax, and validation support. DSCWeaver then automatically generates executable code for synchronization. We demonstrate the advantages of our approach in a service deployment process written in BPEL and evaluate its performance using two metrics: lines of code (LoC) and places to visit (PtV). Evaluation results show that our approach can effectively reduce the development effort of process programmers while providing performance competitive to unwoven BPEL code.
As more organizations adapt their execution models to become process oriented, it is increasingly important to support process modeling with expressive specification languages. This article focuses on the specification of process synchronization (Schmidt & Assmann, 1998).

The recent trends in synchronization modeling have adopted a procedural style in which sequencing constructs (e.g., and-split and and-join) are used to specify the control flow of a process. In languages such as BPEL (business process execution language; Specification: Business Process Execution Language for Web Services Version 1.1, 2003) and XPDL (extensible markup language [XML] process definition language; Workflow Management Coalition [WMC], 2002), recurring synchronization patterns of different processes are factored out as primitives for specifying the execution order of activities (van der Aalst, ter Hofstede, Kiepuszewski, & Barros, 2003). While this sequencing, construct-based approach is good at specifying well-structured processes from the perspective of a single endpoint, it has difficulties with synchronization needs outside of recurring patterns; for example, activities in distributed parallel subprocesses require synchronization.

This article describes an aspect-oriented programming (AOP) approach to synchronization modeling in which synchronization constraints are declaratively expressed in relationship statements between activities in a process. The goal of this approach is to minimize the dependencies between process-specific functionality and synchronization control. It has been observed that synchronization control described procedurally can lead to tangled and fragile code for both process maintenance and adaptation (Lopes & Lieberherr, 1994; Pesic & van der Aalst, 2006; Singh, 2003). The challenge here is to find appropriate high-level primitives to express various synchronization behaviors that alleviate the synchronization difficulties in procedural languages.

Another challenge is to automatically combine the synchronization constraints with other aspects of a process so that the result of applying this method is both compatible with the existing procedural programming environments and suitable for service-oriented environments.

Our method is based on DSCWeaver, a tool that offers a synchronization-aspect extension to procedural languages such as BPEL. The first contribution of DSCWeaver is the generation of an executable program from the DSCL synchronization constraints and other