Processes, Workflows, Web Service Flows: A Reconstruction

Stefan Jablonski

Friedrich-Alexander-University of Erlangen-Nuremberg, Germany
stefan.jablonski@informatik.uni-erlangen.de

Abstract. The last decade was heavily focusing on process-oriented approaches for application integration. It started with the advent of workflow management technology at the beginning of the nineties. This development has been continued with the definition of flow concepts for Web services. In this article, we discuss the purpose and advantages of process-oriented concepts for application integration. Therefore, workflow technology and Web service flows are briefly introduced. Then we assess both technologies with respect to their role in application integration. Especially, we will reconstruct the fundamental differences between workflows and Web services.

1 Application Integration

Today's business is infinitely more complex than it was a long time ago. Each company has running a large number of major applications that implement the business functions. At separate points in time, various people used different technologies to write those applications for distinct purposes. This has resulted in application diversity which was initially not much of a problem, because applications were meant to automate self-sufficient, independent functions.

Meanwhile, applications are so dominant that most of the crucial business functions are implemented by them; their integration has now become a vital issue for an enterprise. Integration is especially of concern to large organizations which need to improve efficiency and collaboration across disparate organizational functions. There are many different approaches to integration [12]: technology integration, data integration, and application integration. In our considerations, we focus on the latter, the most comprehensive and challenging form of integration, namely (enterprise) application integration. Enterprise application integration (EAI) is a whole discipline working on this issue [13]. When integration has to be extended to inter-enterprise integration, the issue becomes even more challenging. In this context, topics like supply chain management, e-procurement, and marketplaces are discussed intensively.

T. Härder and W. Lehner (Eds.): Data Management (Wedekind Festschrift), LNCS 3551, pp. 201-213, 2005.

There is much written about EAI; many publications discuss requirements and high-level approaches in general, instead of offering concrete solutions. We do not want to extend this discussion, but want to pursue an approach in this article which seems to contribute considerably to the EAI issue: the message-oriented approach. However, this approach also has to be analyzed thoroughly, before it can be applied. Two aspects have to be considered:

- The message-oriented approach is absolutely compliant with modern architectural approaches, e.g., services-oriented architectures (SOA) or message-oriented middleware (MOM).
- The message-oriented approach is technology (system)-driven; in order to effectively enact it, application knowledge must be incorporated.

SOA is an architectural style [6] whose goal is to achieve loose coupling among interacting programs, here called software agents. A service is a unit of work performed by a service provider to achieve the desired results for a service consumer. Both provider and consumer are roles played by software agents on behalf of their owners. How is loose coupling among interactive software agents achieved? Two architectural constraints have to be employed:

- Simple and ubiquitous interfaces: The interfaces must be available for all providers and consumers.
- Descriptive messages: The messages must be constrained by an extensible schema which limits the vocabulary and the structure of messages.

SOA is based on messages to be exchanged between software agents. MOM is introduced to increase interoperability, portability, and flexibility for message exchange [4], because software agents can be distributed over multiple heterogeneous platforms. MOM reduces the complexity of the development of software agents that span multiple operating systems and network protocols by insulating the application developer from the details of the various operating systems and network interfaces. MOM provides application programming interfaces (APIs) that extend across diverse platforms and networks.

Both SOA and MOM are system-oriented concepts which focus on the provision of architectural constraints and infrastructural means. What is missing—but this is a natural consequence of the system-oriented perspectives of SOA and MOM—are guidelines when to use what messages in order to facilitate the development of an application system.

Since more than a decade, process modeling has been regarded as a most adequate means to model application systems [7]. The behavior of applications participating in a comprehensive process is typically described by their need to exchange messages among each other. Thus, process modeling (and enactment) ideally completes the message-oriented approach by identifying messages that are produced and consumed by applications. For example in [5], workflow management—which represents a special form of process modeling and enactment Sect. 9—is regarded as the top of a stack of messaging methods to support enterprise application integration. It refines techniques like basic asynchronous messaging, message brokering, and message transformation.