Chapter 1

Object-Aware Business Processes: Fundamental Requirements and their Support in Existing Approaches

Vera Künzle
Ulm University, Germany

Barbara Weber
University of Innsbruck, Austria

Manfred Reichert
Ulm University, Germany

ABSTRACT

Despite the increasing maturity of process management technology not all business processes are adequately supported by it. Support for unstructured and knowledge-intensive processes is missing, especially since they cannot be straight-jacketed into predefined activities. A common characteristic of these processes is the role of business objects and data as drivers for process modeling and enactment. This paper elicits fundamental requirements for effectively supporting such object-aware processes; i.e., their modeling, execution, and monitoring. Imperative, declarative, and data-driven process support approaches are evaluated and how well they support object-aware processes are investigated. A tight integration of process and data as major steps towards further maturation of process management technology is considered.

DOI: 10.4018/978-1-4666-4161-7.ch001
INTRODUCTION

Business Process Management provides generic methods, concepts and techniques for designing, enacting, monitoring, and diagnosing business processes (Van der Aalst, ter Hofstede, & Weske, 2003). When using existing process management systems (PrMS) a business process is typically defined as a set of activities representing business functions and having a specific ordering. What is done during activity execution is out of the control of the PrMS. Most PrMS consider activities as black-boxes in which application data is managed by invoked application components (except routing data and process variables). Whether an activity becomes activated during runtime depends on the state of other activities. Generally, a process requires a number of activities to be accomplished in order to terminate successfully. For end-users, PrMS provide process-oriented views (e.g., worklists.)

Existing PrMS have been primarily designed for highly structured, repetitive processes. By contrast, for unstructured and semi-structured processes existing PrMS do not provide sufficient support (Silver, 2009). In particular, these processes are driven by user decisions and are knowledge-intensive; i.e., they cannot be expressed as a set of activities with specified order and work cannot be straight-jacketed into activities (Van der Aalst, Weske, & Grünbauer, 2005). Another limitation of PrMS is their insufficient process coordination support; i.e., process instances cannot be synchronized at a higher-level of abstraction. Consequently, all behavior relevant in a given context must be defined within one process model (Van der Aalst et al., 2000; Müller, Reichert, & Herbst, 2007). This, in turn, leads to a “contradiction between the way processes can be modeled and preferred work practice” (Sadiq et al., 2005). Finally, since application data is managed within black-box activities, integrated access on business processes and data cannot be provided. Due to these limitations many business applications (e.g., ERP systems) do not rely on PrMS, but are hard-coding process logic instead. Resulting applications are both complex to design and costly to maintain, and even simple process changes require costly code adaptations and testing efforts.

To better understand which processes are handled well by existing PrMS and for which support is unsatisfactory, we conducted several case studies. Amongst others we analyzed business applications with hard-coded process logic; e.g., the processes as implemented in the human resource management system Persis and the reviewing system EasyChair (Künzle & Reichert, 2009a, 2009b). Processes similar to the ones we evaluated can be found in many other fields like order handling, healthcare and release management (Müller, Reichert, & Herbst, 2007). A major finding of all case studies was that data objects act as major driver for process specification and enactment. Consequently, process support requires object-awareness; i.e., business processes and business objects cannot be treated independently from each other. This has implications on the whole process lifecycle since PrMS should consider both object types and their inter-relations. Regarding its execution, on the one hand an object-aware process must be closely linked to relevant object instances; i.e., object attributes must process specific values to invoke certain activities or terminate process execution. On the other hand, an object-aware process does not only require certain data for executing a particular activity; i.e., it should be also able to dynamically react on data changes and newly emerging data. Consequently, process progress needs to be aligned with available object instances and their attribute values at runtime.

Regarding end-user functions provided by hard-coded business applications, in addition to a process-oriented view, there often exists a data-oriented view for managing and accessing data at any point in time. This includes overview tables (e.g., processed object instances) as well as activities that can be optionally executed. The latter are realized based on forms which can be