Chapter 3
Distributed and Adaptive Business Process Execution: A Scalable and Performant Solution Architecture

Michael Pantazoglou
National and Kapodistrian University of Athens, Greece

Aphrodite Tsalgatidou
National and Kapodistrian University of Athens, Greece

George Athanasopoulos
National and Kapodistrian University of Athens, Greece

Pigi Kouki
National and Kapodistrian University of Athens, Greece

ABSTRACT

Centralized business process execution engines are not adequate to guarantee smooth process execution in the presence of multiple, concurrent, long-running process instances exchanging voluminous data. In the centralized architecture of most BPEL engine solutions, the execution of BPEL processes is performed in a closed runtime environment where process instances are isolated from each other, as well as from any other potential sources of information. This prevents processes from finding relative data at runtime to adapt their behavior in a dynamic manner. The goal of this chapter is to present a solution for the performance improvement of BPEL engines by using a distributed architecture that enables the scalable execution of service-oriented processes, while also supporting their data-driven adaptation. The authors propose a decentralized BPEL engine architecture using a hypercube peer-to-peer topology with data-driven adaptation capabilities that incorporates Artificial Intelligence (AI) planning and context-aware computing techniques to support the discovery of process execution paths at deployment time and improve the overall throughput of the execution infrastructure. The proposed solution is part of the runtime infrastructure that was developed for the environmental science industry to support the efficient execution and monitoring of service-oriented environmental science models.

DOI: 10.4018/978-1-4666-6178-3.ch003

INTRODUCTION

Alongside high-level business process notation languages such as BPMN 2.0 (OMG, 2011), Web Services Business Process Execution Language (Alves et al., 2007), abbreviated to WS-BPEL or BPEL, is widely considered to be the de facto standard for the implementation of executable service-oriented business processes as compositions of Web services.

In many cases, which have become evident in various application domains, centralized BPEL engines are clearly not adequate to guarantee smooth process execution, and thereby ensure client satisfaction in the presence of multiple, concurrent, long-running process instances exchanging voluminous data. Indeed, as the numbers of clients grow, the underlying infrastructure needs to maintain and handle multiple process instances while waiting for the external Web services that are invoked to complete their execution.

In some cases, *clustering* techniques can be employed to address the scalability issue, by dispatching the execution of each incoming process request to the BPEL engine residing on the cluster member with the lowest workload. However, the deployment and maintenance of clusters consisting of two or more centralized BPEL engines, sets requirements on the underlying hardware resources that cannot be always fulfilled by the involved organizations. Furthermore, clustering could prove to be an inefficient approach under certain conditions, as it cannot overcome the emergence of bottlenecks that are caused by specific activities of a BPEL process. Moreover, as the execution of a process instance still takes place in a centralized manner, issues relating to large volumes of data are not effectively addressed. In such context, inevitably, the BPEL engine becomes bloated with pending requests coming from multiple concurrent clients. Hence, the overall throughput of the execution infrastructure is dramatically deteriorated, while the process execution times escalate to unacceptable levels.

Aside from the aforementioned scalability issues that derive from the *centralized architecture* of most BPEL engine solutions, the execution of BPEL processes is also performed in a closed runtime environment. More specifically, process instances are isolated from each other, as well as from any other potential sources of information. This prevents processes from finding and exploiting relative data at runtime, in order to improve their predefined behavior in a dynamic manner. By relative data we refer to semantically annotated, structured data that are semantically associated to a given process. Instead, it becomes the responsibility of the process designer to manually adapt the process specification so as to accommodate emerging data sources. For example, rendering a weather calculation process able to incorporate data stemming from a satellite that was not available during process design-time would deem process redesign.

In order to address all these challenges, we propose a *decentralized BPEL engine architecture* with data-driven adaptation capabilities. Our engine employs the *hypercube peer-to-peer (P2P)* topology along with a set of distributed algorithms in order to improve the average process execution times, and the enhancement of the overall throughput of the execution infrastructure in the presence of multiple, concurrent, and long-running process instances.

In addition to the decentralized architecture, the proposed engine accommodates the provisioning of adaptable BPEL processes by exploiting information available to the process environment along with existing services. Adaptation in the context of our approach is about the identification and use of possible alternatives for the achievement of the goals and sub-goals defined in a BPEL process;