Realizing Elastic Processes with ViePEP

Stefan Schulte[1], Philipp Hoenisch[1], Srikumar Venugopal[2], and Schahram Dustdar[1]

[1] Distributed Systems Group, Vienna University of Technology, Austria
{s.schulte,dustdar}@infosys.tuwien.ac.at
[2] School of Computer Science and Engineering,
The University of New South Wales, Sydney, Australia

Abstract. Online business processes are faced with varying workloads that require agile deployment of computing resources. Elastic processes leverage the on-demand provisioning ability of Cloud Computing to allocate and de-allocate resources as required to deal with shifting demand. To realize elastic processes, it is necessary to track the current and future system landscape, monitor the process execution, reason about how to utilize resources in an optimal way, and carry out the necessary actions (e.g., start/stop servers, move services).

Traditional Business Process Management Systems (BPMS) do not consider such needs of elastic process. Within this demo, we present ViePEP, a research BPMS able to execute and monitor resource-, cost- and QoS-elastic, service-based workflows and optimize the overall system landscape based on a reasoning of the non-functional requirements of current and forthcoming elastic processes.

1 Significance to the Field

Resource-intensive tasks are nowadays not only common within scientific workflows, but are also getting more and more common in business processes.[1]. For example, compute- and data-intensive analytical processes are found in the finance industry and in managing smart grids in the energy industry. In the latter case, data from a very large number of sensors needs to be gathered, processed and stored in real-time in order to offer consumers consumption reports or even guarantee grid stability [5]. As the number of active sensors differs during a day, the amount of data also fluctuates to a very large extent. Furthermore, certain processes or process steps are permitted to be postponed to the future, while others need to be carried out immediately.

In such a scenario, the permanent provisioning of IT capacity able to handle peak system loads is obviously not the best solution, as the capacities will not be utilized most of the time. With the advent of Cloud Computing, organizations nowadays have got a much more cost-savvy alternative which allows them to make use of computing resources in an on-demand, utility-like fashion [1].

[1] In the following, we will also make use of the term "workflows" to name the automated parts of business processes.

To the best of our knowledge, so far, little effort has been put into the investigation of methods and tools to integrate automated process execution and Cloud Computing in order to realize so-called *elastic processes* [2].

Nevertheless, scalability and cost-effective allocation of single tasks and applications have been observed by many researchers, e.g., [4]. So far, applications are mostly regarded in isolation, i.e., a process perspective across utilized resources (software, hardware, humans) is typically not applied. While Cloud resources have been used for executing scientific workflows [3], Service Level Agreements (SLAs) are typically not as much a concern in this domain as they are for business processes. In our experience, there is a lack of a BPMS able to carry out many service-based workflows in parallel, estimate their current *and* future resource demand under given Quality of Service (QoS) constraints, and allocate resources dynamically to meet their individual SLAs. This needs analysing the process to discover which of its steps determine the performance of its execution and prioritising them, allocating resources to services to address demand and balancing load on the resources by moving services between them.

Within this demo paper, we will present our *Vienna Platform for Elastic Processes (ViePEP)* [6], which is a research-driven, prototypical BPMS capable to execute elastic processes, monitor the current utilization of invoked resources as well as reason about future resource demands, and carry out necessary actions.

2 System Overview

In this section, we present the overall system architecture of ViePEP as depicted in Figure 1. Within ViePEP, processes are modeled making use of a XML-based description format, which also defines non-functional constraints and preferences for each process step. Each process step is mapped to a particular service which is able to fulfill it. The execution of processes is constantly monitored and the results are fed to a centralised manager. Based on the monitoring data, and knowledge about current and future process instances and their individual non-functional requirements, it is possible to reason about the best allocation of services to Virtual Machines (VMs) as well as more general actions, e.g., starting and terminating servers.

Figure 1 shows an overview of ViePEP using an FMC Block Diagram. Both the BPMS and Backends are running inside dedicated VMs. For the sake of simplicity, only one Backend VM is depicted; however, an arbitrary number of Backend VMs may be invoked. The BPMS consists out of the following main components:

- The **Workflow Manager** controls the workflow executions. It invokes the services step by step. Before each invocation, it queries the Load Balancer for the best fitting service instance for the next step. In addition, the Workflow Manager measures service-specific QoS data like response time.
- The **Load Balancer** retrieves the actual VM states from the shared memory and is responsible to balance the service invocations, so that the system load of all Backend VMs does not exceed a predefined threshold.

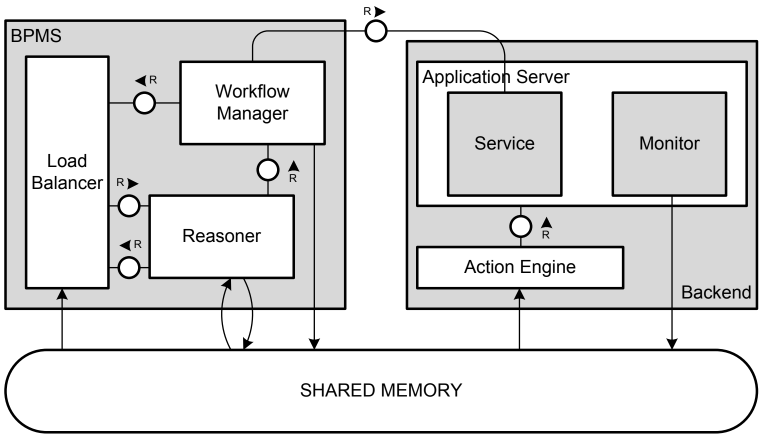

Fig. 1. Framework Architecture

– Reasoning is performed in order to estimate the future resource demand and optimize the overall process landscape, e.g., by choosing between different actions like duplicating, or moving Web services from one VM to another or starting and terminating servers [4]. For this, dynamic resource and cost information must be taken into account in order to decide how to utilize resources in a cost-efficient way while guaranteeing the non-functional constraints of each process step or overall process, respectively.
  In ViePEP, the **Reasoner** is responsible for reasoning and optimization of the process landscape. It requests information about future workflow instantiations and service invocations from the Workflow Manager. Furthermore, it retrieves the status of all Backend VMs from the shared memory and communicates with the Load Balancer in order to decide whether a particular Backend VM is sufficient to carry out a service, if another VM hosting that service needs to be started, or if a VM can be stopped due to low load.

The ViePEP-enabled Backend VMs contain the following components:

– An **Application Server** (here: Apache Tomcat) hosting a Web service.
– A **Monitor** is part of the Application Server. After an elastic process has been modeled and the corresponding workflow executed, it is necessary to monitor the current resource workload as well as the non-functional behavior of the single invocation services. Therefore, ViePEP allows to monitor the CPU and RAM utilization of all Cloud servers within a system. Monitoring data is stored in a shared memory based on Tuple Spaces, allowing the ViePEP BPMS to access the status of all running VMs.
– The **Action Engine** is responsible to execute commands to the Application Server. It gets according commands from the Reasoner through the shared memory data structure. It can copy the whole system state and start a new VM or shut itself down if necessary.

3 Demonstration

At this point, ViePEP is a fully functional research prototype. In the demo, we will show how the different functionalities (monitoring, reasoning, optimization) as presented in Section 2 can be used in order to control an elastic process landscape. We apply the perspective of a provider of elastic processes, i.e., a broker who gets workflow requests from clients and automatically provides the Cloud resources to execute the according workflow steps (services). For this, we show how a workflow is added to an existing system landscape and consequently executed. We apply a simplified example workflow from the energy domain. For reasons of simplicity, we are applying only one workflow model; however, the model is instantiated several times in parallel in order to show how changing loads on a VM leads to the start and termination of VMs.

We will show how the monitoring data is visualized and consequently exploited in order to assess current resource demands. Furthermore, we will compute future workflow instantiations and service invocations. Based on this, it is possible to derive the future resource demand and consequently carry out actions as explained above. The according screencast can be found at `http://www.infosys.tuwien.ac.at/prototypes/ViePEP/ViePEP_index.html`

Acknowledgements. This work is partially supported by the Commission of the European Union within the SIMPLI-CITY FP7-ICT project (Grant agreement no. 318201) and the Austrian Science Fund (FWF): P23313-N23.

Part of the implementation has been done during Philipp's stay at the University of New South Wales, which was supported by a scholarship from Vienna University of Technology's International Office. We'd like to thank Han Li for his help with the implementation.

References

1. Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I.: Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computing Systems 25(6), 599–616 (2009)
2. Dustdar, S., Guo, Y., Satzger, B., Truong, H.L.: Principles of Elastic Processes. IEEE Internet Computing 15(5), 66–71 (2011)
3. Juve, G., Deelman, E.: Scientific Workflows and Clouds. ACM Crossroads 16(3), 14–18 (2010)
4. Li, H., Venugopal, S.: Using reinforcement learning for controlling an elastic web application hosting platform. In: 8th International Conference on Autonomic Computing (ICAC 2011), pp. 205–208 (2011)
5. Rusitschka, S., Eger, K., Gerdes, C.: Smart Grid Data Cloud: A Model for Utilizing Cloud Computing in the Smart Grid Domain. In: 1st IEEE International Conference on Smart Grid Communications (SmartGridComm), pp. 483–488 (2010)
6. Schulte, S., Hoenisch, P., Venugopal, S., Dustdar, S.: Introducing the Vienna Platform for Elastic Processes. In: Zhu, H., Ghose, A., Yu, Q., Perrin, O., Wang, J., Wang, Y., Delis, A., Sheng, Q.Z. (eds.) ICSOC 2012. LNCS, vol. 7759, pp. 179–190. Springer, Heidelberg (2013)