ViePEP – A BPMS for Elastic Processes

Philipp Hoenisch

Distributed Systems Group, Vienna University of Technology, Austria
p.hoenisch@infosys.tuwien.ac.at

Abstract. In today's IT industry resource-intensive tasks are playing an increasing role in business processes. By the emergence of Cloud computing it is nowadays possible to deploy such tasks onto computing resources leased in an on-demand fashion from Cloud providers. This enabled the realization of so-called Elastic Processes (EPs). These are able to dynamically adjust their used resources in order to meet varying workloads. Till now, traditional Business Process Management Systems (BPMSs) do not consider the needs of Elastic Processes such as monitoring the current system load, reasoning about optimally utilized resources, in order to ensure given Quality of Service constraints while executing required actions such as starting, stopping servers or moving services from one server to an other. This paper focuses on our current work on ViePEP, a research BPMS for the Cloud capable of handling the aforementioned requirements of EPs.

1 Introduction

Business Process Management is a multifaceted approach which covers the organizational, management and technical aspects of business processes. Further, it "includes methods, techniques, and tools to support the design, enactment, management, and analysis of operational business processes" [1]. In recent years, a specific subtopic of business process management gained more attention in many industries: the automatic processing of business processes also known as workflows (excluding the involvement of human services). In many cases, software services are composed to a workflow in order to realize a specific functionality. Therefore, by its nature, the individual (software) services in such a composition differ in terms of required computing resources (such as CPU, RAM, bandwidth, ...), priority and execution order. In order to realize and process such a workflow, different techniques, concepts, methodologies and frameworks from the field of computer science are required.

Such workflows are becoming more and more relevant in business processes in several different industries. Examples are coming from the finance industry, managing of smart grids or from the energy domain. In the latter one, data from a very large extend of sensors have to be gathered, processed and analyzed in almost real time. Further, this data has to be stored in order to be retrievable to a later moment for the generation of reports or statistical analysis.

It is a common service provider problem that acquired resources are hardly fully utilized, which is not very cost efficient. While this enables the provision of

N. Herzberg, M. Kunze (eds.): Proceedings of the 6th Central European Workshop on Services and their Composition (ZEUS 2014), Potsdam, Germany, 27-03-2014, published at http://ceur-ws.org

a high quality of service, it ends up in unwanted waste of resources. In contrast, if too many requests are forwarded to a particular Virtual Machine (VM) it may crash or the services being executed may produce faulty results. As this example is very specific for computer engineering, it is a common problem in economy. In computational processes in the context of Cloud computing, this can be described as the problem of *Elastic Processes* (EPs). EPs are "precisely defining the various facets of elasticity that capture process dynamics in cloud computing [...]. The main properties for modeling EPs' economic and physical dynamics are *resource elasticity*, *cost elasticity*, and *quality elasticity*" [7]. While EPs are a complex concept, the problem around it can be stated as: Finding the correct relation between Resources, Costs and Service Quality, or in other words: Acquire as little resources as required in order to ensure the best possible quality of service while only paying the least required amount.

Therefore, a technology is needed which is able react to a dynamic change of needed computing resources, while still ensuring the faultless business process execution. This means, this kind of technology has to be able to provide additional resources when needed, such that, the business process execution will not wait, stuck or even crash at a critical moment. For that reason, research scientists from the field of Business Process Management and software engineering have put a remarkable focus on the solution of such a problem in recent years.

In this paper, we present the ongoing research on Elastic Processes in the Cloud. More precisely, we present our extensive work on *ViePEP* – the *Vienna Platform for Elastic Processes*. ViePEP is a research-driven BPMS for the Cloud, capable of cost-effective workflow processing while monitoring their underlying service executions in order to provide a certain level of Quality-of-Service (QoS) and ensure no Service Level Agreement (SLA) violations.

The remainder of this paper is organized as follows: After a brief introduction of ViePEP's architecture including its functionality (Sect. 2) we will give some information about our current research on workflow scheduling (Sect. 3.1) and resource optimization (Sect. 3.2). Sect. 4 will give an overview of the related work and Sect. 5 will conclude this paper and give a short outlook on our future work.

2 The Vienna Platform for Elastic Processes

In this section we want to introduce ViePEP. In general, ViePEP can be seen as a broker middleware which accepts workflow requests by a customer (Client in Fig. 1) and takes care of its execution. By the upcoming of Cloud computing and the new paradigm of Software-as-a-Service (SaaS) [4], many business processes are already SaaS-enabled, which means, they can be deployed independently and reusable in the Cloud. ViePEP takes care of the hosting and managing of the software services and maps the clients workflow requests in order to execute them. In addition, ViePEP considers the Service Level Agreements, which may be defined by clients. In order to accept hundreds of workflow requests simultaneously while still being able to ensure the given SLAs and being as

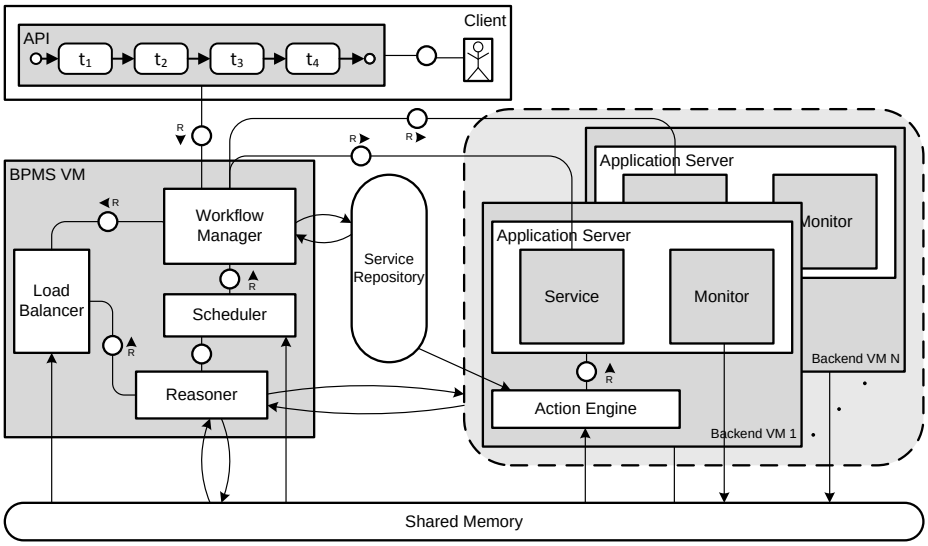

Fig. 1. ViePEP – Architecture

cost-efficient as possible, ViePEP was designed according to the MAPE-K cycle (*Monitor*, *Analyze*, *Plan* and *Execute*) which is used for autonomic computing[12].

As shown in Fig. 1, ViePEP has five top level entities: First, the *Client* models service-based workflows and can optionally define Service Level Agreements. Clients may request additional workflows consecutively or even many simultaneously. In addition, ViePEP is able to serve several different clients in parallel.

Second, the *BPMS VM* offers the core functionality of ViePEP. It is responsible of accepting new workflow requests (*Workflow Manager*) and stores them for a later or immediate execution. The exact execution time is computed by the *Scheduler*, which creates a schedule plan according the given deadlines defined in the given SLAs within the workflow requests. A first version of this scheduling plan is forwarded to the *Reasoner* which computes the amount of required resources. This can be done by reasoning on historical data from the Shared Memory. As this is the core functionality it will be further discussed in Sect. 3. Thus acquired resources are used equally, the single service invocations are balanced and distributed to the single service instances running on different VMs (Backend VM). Beside of the workflow executions, the Workflow Manager also measures the execution time of single service invocations, which is a prerequisite to detect possible deviations from the expected QoS attributes. By doing so, it is able to issue corresponding countermeasures if required.

Third, the *Backend VM* hosts an *Application Server* on which a particular service instance is deployed. In order to monitor the services' QoS, a *Monitoring* component is deployed. It measures the VM's CPU and RAM load and stores this information in the Shared Memory. The *Action Engine* is able to perform

actions issued by the Reasoner such as *deploy*, *undeploy* a particular service, or *move* a running service to another Backend VM.

Fourth, both, the *Shared Memory* and *Service Repository* are helper components and their functionalities are simple. The latter hosts all available services in form of deployable Web application ARchive (WAR) files. The Shared Memory is used to store the monitored data from each single Backend VM and share it with the BPMS VM. For a more detailed description about ViePEP please be refereed to [8,9].

3 Scheduling, Reasoning and Optimization

As ViePEP is a fully functional BPMS for the Cloud it takes care of workflow scheduling (Sect. 3.1) and its actual workflow execution. However, in contrast to common BPMSs, ViePEP is considering the future workflow executions and reasons in order to achieve a cost-effective optimized system (Sect. 3.2). For that, we will discuss in this section our ongoing work on the core functionality of ViePEP: the reasoning about current and future workflow execution including the computation of the resource demand and how ViePEP achieves a resource optimized system landscape.

3.1 Scheduling

The core functionality of a common BPMS is to process workflows. In order to know when a particular workflow execution should be started, several different procedures have been established. In many cases the incoming workflows are first ordered according their priorities before being processed. This allows the processing of workflow requests with a higher priority before workflows with a lower priority. These different techniques have already been discussed by many researchers and are not focus of this work [17]. ViePEP is making use of a priority-based scheduling approach, i. e. workflows with a higher priority are processed before workflows with a lower priority. Priorities are calculated based on the deadline defined in the given SLAs of the workflow requests. If two or more clients have defined the same deadline for different workflows, ViePEP will serve them according a first-come first-serve manner. However, as ViePEP is able to process several workflows simultaneously while considering each given SLA, the clients will not notice any delay.

Clients can issue a workflow request and define a specific deadline, i. e. a point of time defining when the execution has to be ended. This can be defined either for the whole workflow or for a particular single step in a workflow. ViePEP's task is to process the workflow while ensuring this deadline. Since ViePEP is a BPMS serving several hundred or even thousand clients in parallel, the workflow scheduling is a complex task. Hoenisch et al., [8] describes the latest implemented scheduling algorithm. It splits up a workflow into its single steps and assigns them to a particular time slot. Each time slot is exactly as long as the single service invocation lasts. Service invocations of the same type, i. e. the same kind

of software service has to be invoked, can be combined within the same time slot in order to make fully use of the acquired resources.

This scheduling is a straight forward task for sequential workflows (which are the only one supported in ViePEP at the moment). However, it gets a much more complicated challenge if the workflow is more realistic, e. g. if it involves branches such as XORs, ANDs or loops. While ANDs are quite easy to implement, i. e. both branches have to be considered, XORs are much more complicated. In the latter one, a BPMS, considering XORs has to deal with probabilities. This means, it has to calculate how high is the chance that a workflow follows either the one path or the other one (but not both). This can either be done static, e. g. the probability that a workflow follows the one path is always a fix value and the other direction is always 1 minus that value, or dynamically. Of course, in a real world scenario, those values are not static, but may change dynamically, e. g. they depend on the output or input of previous steps. Therefore, a "smart" BPMS has to be able to learn from historical executions and predict the probabilities. While the scheduling itself might not be the biggest challenge, as already a lot of research has happened in this field, the combination with computing the demand of resources is much more complicated.

3.2 Reasoning & Optimization

As ViePEP is a smart BPMS it tries to consider all of the three properties of EPs equally. This means, the acquired resources are fully utilized in order to be as cost-efficient as possible. However, in the current version, the quality of a service is hardly defined by the services' output, rather than ensuring the given SLAs, i. e. ensuring that a workflow is processed in time.

Resource Prediction
As mentioned before, ViePEP tries to utilize the acquired resources as efficiently as possible. This means, the Reasoner computes the required amount of resources from historical data. In the current version, ViePEP makes use of Ordinary Least Squares (OLS) Linear Regression. At the moment, the provided services are only CPU intensive. Therefore, a high CPU load would influence a hosted service the most. Therefore, OLS is a perfect choice for the current scenarios as it is limited to two variables (2-dimensional optimization). While this is only the case in our selected services, in the case of image processing, the limiting factor may be the internal memory or RAM. For that reason, we are working on a multi-dimensional resource prediction mechanism considering several QoS aspects of a service. As in real-world scenarios, service invocations do not produce a linear resource consumption, and may last several minutes or even hours, a linear regression is not applicable anymore. Therefore, we propose to approach this problem from the other way around and make use of online reasoning approaches (e. g. *Kalman Filters*) in order to compute how many service invocations are possible on a particular resource and to predict the future demand of resources. In general, a Kalman Filter aims at providing the means of a mathematical equation to estimate a state of a process or a stream of updated data. In addition, a Kalman

filter makes use of historical and life data and is able to predict a future state even if the precise nature of the system is not known. Therefore, by "feeding" a Kalman Filter with monitored data, e. g. such as how many invocations happened in parallel, producing a certain load in CPU and used a particular amount of RAM, it is possible to compute how many invocations the monitored VM is able to handle in the future.

Resource Allocation
The result of the resource prediction (see Sect. 3.1) is a detailed plan of which service invocation is assigned to which VM and if additional VMs are required or if unneeded once can be released. The execution of this task is simple software engineering.

However, in many cases, companies manage an own private Cloud. Which means, neglecting the energy costs, these are free resources and ready to use. Therefore, the Reasoner should consider allocating private resources first until the demand is reached. However, it may be the case, that not enough resources are available in the private Cloud. Therefore, additional resources can be bought from an external Cloud provider (public Cloud). The result is a so called Hybrid Cloud. As the resource allocation on its own is not such a complicated task, the Reasoner has to consider the different pricing schemes of the public Cloud. Amazon's EC2 for example charges their customers on an hourly model. This means, it is not economic to acquire such a resource for just 20 minutes. Therefore a rescheduling might be necessary. In addition, several Cloud infrastructure providers offer different kind of VM types having a different amount of computing resources such as a multi-core CPU or more RAM and cost differently.

4 Related Work

To the best of our knowledge, so far, surprisingly little effort has been investigated into the field of elastic processes in the sense of dynamic resource allocation and elastic process execution [7]. Nevertheless, there is some related work which remains to be mentioned from the fields of Grid computing and Cloud computing.

In both cases, scalability and cost-effective allocation of single tasks and services have been the only focus by many researchers. Most research efforts are focusing on minimizing the costs for the consumers (clients) while taking into account a maximum allowed execution time or other QoS attributes [5,14]. However, in later research, new approaches also consider SLA enforcement including the consideration of penalty costs. This lead a completely different approach of resource management in a Clout environment [3,6].

In contrast to that, more recent research efforts are focusing on the infrastructure perspective, i. e. a higher resource utilization [13,16] or maximizing the Cloud provider's profit [15]. While most of the time, only rule-based thresholds are applied to identify whether a new resources are required or unneeded can be freed, Li at al. [16] makes use of automated machine learning to scale applications up or down. However, all these approaches lack of the consideration of the

process perspective but focus on an ad-hoc allocation of Cloud-based resources for single services. Only a few research already considered a process perspective in regard for Scientific Workflows [10,11]. Although Business Processes and Scientific Workflows share several similarities, they also differ vastly in the sense of timeliness. The latter one can be run sometimes during the night, ensuring only the availability of the result in the morning, in Business Processes, the requests has to be processed in almost real time.

Similar to our work, a workflow model, i. e. workflows are composed from single software services which can be deployed in the Cloud, has been considered by Wei and Blake [18] and Bessai et al. [2]. Nevertheless, only one workflow is considered simultaneously which is one of the main focuses of ViePEP.

5 Conclusion

In this paper we have presented our current state and work on ViePEP – the Vienna Platform for Elastic Processes. ViePEP was already evaluated by simplified use cases in [9,8]. However, although we have shown that ViePEP is able to handle the presented use cases, optimize the acquired resources by rescheduling incoming workflows in order to be cost efficient as possible, ViePEP is still not yet fully supporting *Elastic Processes* [7], which is heavily focused by us in our future work. Therefore, we are extending ViePEP in order to support more realistic workflows including branches and loops. Further, it is planned to evaluate ViePEP on an hybrid Cloud environment involving Amazon's EC2, Windows Azure and others. In addition to that, an interested reader may have noticed that ViePEP and the Shared Memory may result in a bottleneck as well. While we already considered the latter one, and replaced the Shared Memory with a lightweight JMS Queue, the scalability of the BPMS VM is still part of our future work.

Acknowledgements. This work is partially supported by the Commission of the European Union within the SIMPLI-CITY FP7-ICT project (Grant agreement no. 318201).

References

1. van der Aalst, W.M.P., Hofstede, A.H.M.T., Weske, M.: Business Process Management: A Survey. In: International Conference on Business Process Management (BPM 2003). pp. 1–12. Springer, Berlin Heidelberg (2003)
2. Bessai, K., Youcef, S., Oulamara, A., Godart, C., Nurcan, S.: Resources allocation and scheduling approaches for business process applications in Cloud contexts. In: 4th IEEE International Conference on Cloud Computing Technology and Science Proceedings (CloudCom 2012). pp. 496–503. IEEE Computer Society, Washington, DC, USA (2012)
3. Buyya, R., Ranjan, R., Calheiros, R.N.: InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services. In: 10th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP 2010). Lecture Notes in Computer Science, vol. 6081, pp. 13–31. Springer, Berlin Heidelberg (2010)

4. 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)
5. Cao, Q., Wei, Z.B., Gong, W.M.: An Optimized Algorithm for Task Scheduling Based on Activity Based Costing in Cloud Computing. In: 3rd International Conference on Bioinformatics and Biomedical Engineering (ICBBE 2009). pp. 1–3. IEEE Computer Society, Washington, DC, USA (2009)
6. Cardellini, V., Casalicchio, E., Lo Presti, F., Silvestri, L.: SLA-aware Resource Management for Application Service Providers in the Cloud. In: First International Symposium on Network Cloud Computing and Applications (NCCA '11). pp. 20–27. IEEE Computer Society, Washington, DC, USA (2011)
7. Dustdar, S., Guo, Y., Satzger, B., Truong, H.L.: Principles of Elastic Processes. IEEE Internet Computing 15(5), 66–71 (2011)
8. Hoenisch, P., Schulte, S., Dustdar, S.: Workflow Scheduling and Resource Allocation for Cloud-based Execution of Elastic Processes. In: IEEE 6th International Conference on Service Oriented Computing and Applications (SOCA 2013). pp. 1–9. IEEE (2013)
9. Hoenisch, P., Schulte, S., Dustdar, S., Venugopal, S.: Self-Adaptive Resource Allocation for Elastic Process Execution. In: IEEE 6th International Conference on Cloud Computing (CLOUD 2013). pp. 220–227. IEEE (2013)
10. Hoffa, C., Mehta, G., Freeman, T., Deelman, E., Keahey, K., Berriman, B., Good, J.: On the Use of Cloud Computing for Scientific Workflows. In: IEEE Fourth International Conference on e-Science (eScience'08). pp. 640–645. IEEE Computer Society, Washington, DC, USA (2008)
11. Juve, G., Deelman, E.: Scientific Workflows and Clouds. ACM Crossroads 16(3), 14–18 (2010)
12. Kephart, J.O., Chess, D.M.: The Vision of Autonomic Computing. Computer 36(1), 41–50 (2003)
13. Kertesz, A., Kecskemeti, G., Brandic, I.: An Interoperable and Self-adaptive Approach for SLA-based Service Virtualization in Heterogeneous Cloud Environments (forthcoming). Future Generation Computer Systems NN(NN), NN–NN (2012)
14. Lampe, U., Mayer, T., Hiemer, J., Schuller, D., Steinmetz, R.: Enabling Cost-Efficient Software Service Distribution in Infrastructure Clouds at Run Time. In: 4th IEEE International Conference on Service-Oriented Computing and Applications (SOCA 2011). pp. 1–8. IEEE Computer Society, Washington, DC, USA (2011)
15. Lee, Y.C., Wang, C., Zomaya, A.Y., Zhou, B.B.: Profit-Driven Service Request Scheduling in Clouds. In: 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid 2010). pp. 15–24. IEEE Computer Society, Washington, DC, USA (2010)
16. 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. ACM, New York, NY, USA (2011)
17. Schulte, S., Schuller, D., Hoenisch, P., Lampe, U., Dustdar, S., Steinmetz, R.: Cost-Driven Optimization of Cloud Resource Allocation for Elastic Processes. International Journal of Cloud Computing 1(2), 1–14 (2013)
18. Wei, Y., Blake, M.B.: Adaptive Service Workflow Configuration and Agent-Based Virtual Resource Management in the Cloud. In: 2013 IEEE International Conference on Cloud Engineering (IC2E 2013). pp. 279–284. IEEE Computer Society, Washington, DC, USA (2013)