A MONITORING AND ADAPTATION ARCHITECTURE FOR THE FUTURE INTERNET OF SERVICES

Iris Braun, Matthias Wauer, Sandro Reichert, Josef Spillner, Anja Strunk, Alexander Schill
Technische Universität Dresden
Faculty of Computer Science
Institute of Systems Architecture
Chair of Computer Networks
Nöthnitzer Str. 46
01187 Dresden, Germany
{iris.braun, matthias.wauer, sandro.reichert, josef.spillner, anja.strunk, alexander.schill}@tu-dresden.de

ABSTRACT

The article presents the architecture of a distributed platform for the future Internet of Services, where services can be provided and sold over so-called service market places. We focus on the description of the monitoring and adaptation mechanisms in the service execution phase which are necessary to ensure the quality of service executions and the reliability of service level agreements.

KEYWORDS

SOA, monitoring, adaptation, architecture, web service, SLA

INTRODUCTION

Nowadays the Internet is an unstructured collection of vast amounts of static information and knowledge. Semantic technologies can fundamentally transform the existing Internet into a network of structured knowledge resources which can be efficiently established and complex knowledge can be reproduced as well. The German research program THESEUS[1] aims to develop a platform for the Internet of the next generation which will provide easy access to the structured global knowledge and to novel services, and crucially improve the quality of information of the relevant contents that are needed at a given moment.

The vision of the future Internet is to enhance the web of information to a web of services. The goal of the use case TEXO[2] within the THESEUS project is to develop so-called business webs where economically viable services are provided, distributed and combined to build value-added services for the customers. Services can be of technical or traditional business nature, or hybrids. Service marketplaces emerge as web platforms where providers can publish and sell services as well as customers can find suitable services and combine them, corresponding to their needs. The basis for successful business processes in the future Internet of Services (IoS) is a reliable and flexible execution of technical services. Customers will use services if and only if they get a contractually bounded Quality of Service (QoS), e.g. availability, response time, or throughput. Contracts between consumer and provider contain, besides functional aspects of the service, non-functional properties which describe the negotiated QoS in a Service Level Agreement (SLA). In order to enforce the SLAs, services have to be monitored continuously and adaptation mechanisms need to be triggered in case of an impending violation of an SLA. Any marketplace for the IoS will therefore need support from autonomous service execution nodes with integrated monitoring and adaptation capabilities.

[1] THESEUS programme website: http://www.theseus-programm.de

[2] Access to our TEXO contributions: http://texo.inf.tu-dresden.de

In [Braun08] we introduced methods and abstract architectural suggestions to ensure SLAs during the whole service usage lifecycle. In this article we discuss and summarize intermediate results of this research. The paper will concentrate on the service execution phase (runtime) and present a concrete architecture for distributed monitoring and adaptation of web services and compositions thereof. Section 2 gives an architecture overview, followed by detailed descriptions of the subarchitectures for monitoring in section 3 and adaptation in section 4. The article finishes with a conclusion and outlook to future work.

ARCHITECTURE OVERVIEW

Figure 1 illustrates the architecture[3] of our monitoring and adaptation (M&A) platform for web service execution and trading. We make a clear differentiation between a central marketplace server and several execution servers, so that service developers may host services at providers of their choice at distributed locations and offer them at the marketplace to a wide audience. This allows the distributed platform to scale, while the overall service quality must still be maintained. Our architecture principles have been derived from this goal.

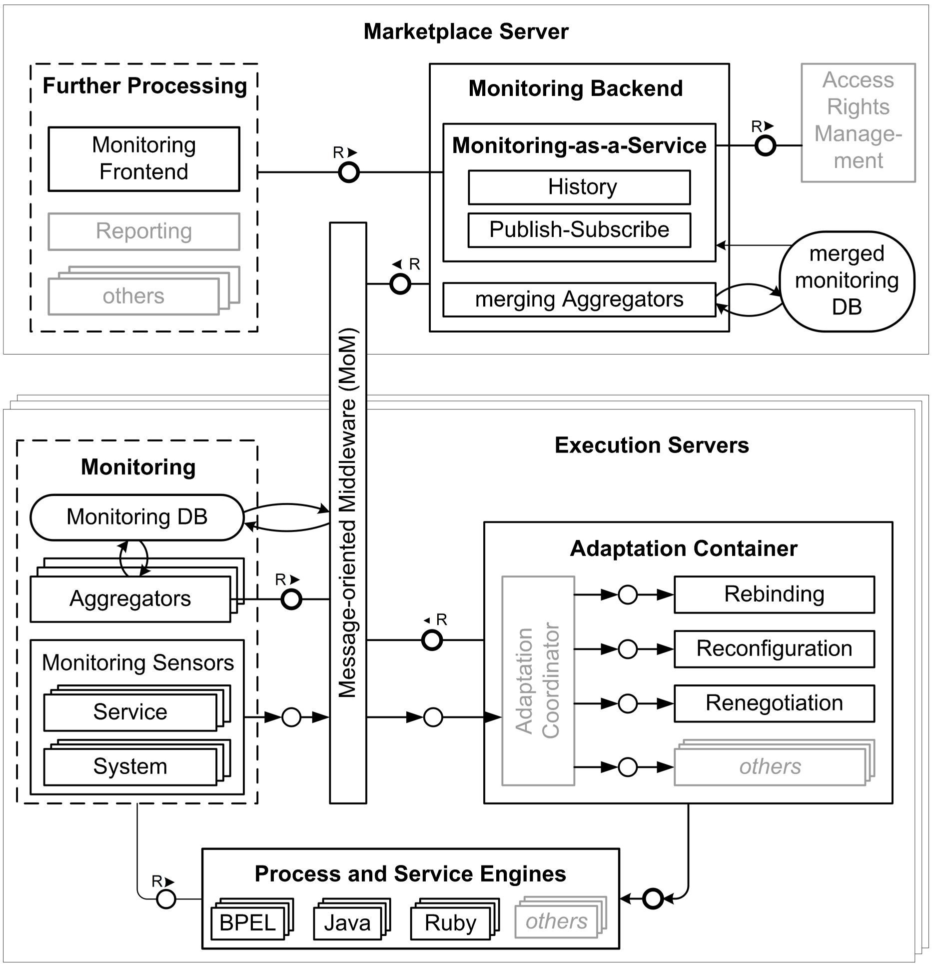

Figure 1: Architecture of monitoring and adaptation platform

The marketplace offers basic tools for e.g. service discovery, SLA negotiation, billing and access rights management to be used by both users and providers. Its design allows it to host heterogeneous tradable

[3] FMC-notation, see http://www.fmc-modeling.org

services and require valid SLAs from callers of the services. Our understanding of technical services assumes their availability as self-contained packages. The packages include the service implementation, declarative descriptions, and optionally templates for SLAs to be used in the negotiation process. In this article we focus on gathering monitoring data at the execution servers, aggregating and using it for adaptation purposes in case of imminent SLA violations and providing a central interface for further processing by consumers at the marketplace. The extension of our previous results on individual aspects of contracting, execution, monitoring and adaptation to a more process-oriented view can be understood when looking at Figure 2. As soon as a contract is established, the monitoring starts independently from service invocations. For every invocation, additional M&A tasks are performed. Altogether, the collected information leads to long-term improvements of future contracts.

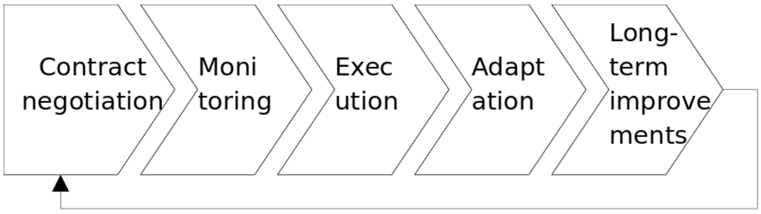

Figure 2: Runtime activities on execution nodes attached to service marketplaces

Our architecture combines knowledge from previous systems for M&A, like the importance of externalized adaptivity [Garlan01] and the monitoring of metrics according to QoS policies [Michaut02], with a direct suitability for Internet service marketplaces. Our approach is not as holistic as the S-Cube M&A roadmap [Kazhamiakin08], but is already implemented and functional enough for enhancing the IoS.

MONITORING SERVICE QUALITY

Contract-driven monitors on service execution platforms generate valuable data out of service offering, usage by consumers and internal execution [Spillner09a]. For our approach, we depend on precise structured information to draw decisions on whether and how to adapt the execution in order to keep guarantees. Sometimes, only future invocations can be improved based on long-term observations of service behaviour, while long-lasting calls like streaming services allow for a direct change in quality.

Monitoring activities usually produce a lot of data. Key questions include: Which data should be collected? Who gets access to the data, and what for? In our approach, we suggest a pyramidal process of gathering as much monitoring data as possible as needed for some adaptation tasks, but then gradually discarding it for the parts of the system which need less detailed information. The proposed reduction of information is achieved by using rule-based monitoring filters which let the number of execution platforms scale up to a certain minimum monitoring granularity. Monitoring sources include the protocol-level invocation entry points, operating system and service container process tracing, behavioural observation of network-related properties such as availability at a certain point of time and local system load development. We explicitly include prediction through aggregation of measured values and statistical distribution analysis since we aim to correct the service execution before SLAs get violated [Halima08].

Additional monitoring sensors can be added at runtime and activated according to negotiated SLAs [Spillner09b]. This design makes it possible to extend the nature of tradable services with custom sensors for special-purpose services. It also helps keeping the monitor itself scalable as the amount of required memory grows with the number of monitored objectives in the SLAs, and the amount of processing time grows with the number of SLAs. Aligned with our aim to support heterogeneous services, the monitor is capable of processing various SLA formats including custom variants of WS-Agreements and WSLA. This decision will keep the architecture useful even if the development of SLA languages moves on. The combination of monitor features such as SLA abstraction, dynamic sensor activation and distributed aggregation is unique to our architecture.

In order to guarantee a timely processing of monitoring events on the execution servers, we use a message-oriented middleware (MoM) to spread events to a number of local and remote receivers. Local receivers on the execution servers include the database for storing the events and an adaptation component for acting upon any alarming situation. The central marketplace's monitoring backend runs the remote receiver which periodically stores a subset of the overall produced data to make it centrally available to service users. A number of merging and filtering aggregators on all servers help to further reduce the information overflow and extract key performance indicators from the monitoring data. Hence, visualization and reporting applications can display both detailed information about each contract or service execution as well as higher-level information needed to compare the quality of services or the performance of execution servers. Other applications can likewise use the *Monitoring-as-a-Service* (MaaS) interface according to access policies to acquire and condition the available information. For instance, the service descriptions inside a service registry can be updated with increasingly realistic values over time. This back-channel is an important improvement over previous distributed pyramidal approaches [Prieto06].

Among the local receivers, the adaptation components operate under real-time constraints. The faster the adaptation can begin, the better the chances are for keeping the promised (and contractually fixed) guarantees with respect to service users.

ADAPTATION OF COMPOSITE SERVICES

To satisfy the user’s expectations, the highest goal while executing a web service is to ensure its guaranteed QoS level even in case of an error. Therefore, several adaptation strategies exist from migrating the service and switching to a redundant hardware structure to restructuring a composite service [Meyer07]. Due to the heterogeneous nature and varying degree of adaptability of web services, we have defined an extensible adaptation architecture called *adaptation container*. Like a plug-in, each adaptation mechanism is deployed as a separate module and performs its adaptation independently from the other parts. Currently we have implemented a *rebinding*, a *reconfiguration* and a *renegotiation* module.

The rebinding mechanism targeted at BPEL processes replaces already bound services by alternative ones in order to ensure the QoS level and executability of a service composition [Strunk09a]. As opposed to other approaches, we neither extend any BPEL engine nor WS-BPEL to support rebinding [Strunk09b]. Thus our rebinding component is fully compatible with standard WS-BPEL and can be flexible integrated in arbitrary SOA service execution environments.

Service reconfiguration affects the availment of system resources by services. It is targeted at services providing generic declarative reconfiguration interfaces which is hard to achieve with today’s implementation frameworks like Java servlets or OSGi bundles. We have thus developed a cross-domain property translator which takes knowledge of the effects of service properties on system properties into account to reconfigure services and optimise system performance within contractually defined limits [Spillner09c].

SLA renegotiation leaves the service execution intact and instead seeks approval from the user’s contract agent to alter the SLA towards more relaxed constraints [Parkin08]. This mechanism works independently from the underlying service model, but is constrained by legal limits on dynamics of SLAs. Other adaptation mechanisms such as resource elasticity or network protocol switches can be integrated fairly well at any time.

The selection and scheduling of the different adaptation mechanisms is done by the *adaptation coordinator*. Each module registers itself at the coordinator by publishing its abilities and scope. As soon as the monitoring component detects an error, it triggers the adaptation coordinator, which decides, based on the published information, which adaptation mechanism will be performed. We are developing the adaptation coordinator, especially the definition of its decision rules, for even complex scenarios with conflicting SLA priorities and unshirkable degradation of quality in one set of services in order to keep guarantees in the complementary set.

CONCLUSION

In this article we propose our runtime platform as a base component for service marketplaces. The division into a central marketplace and various execution servers helps ensuring a contract-bound service execution even for large-scale service providers. A key concept for abidance of contracts is monitoring-driven coordinated adaptivity. By supporting many value-adding applications with monitoring data through Monitoring-as-a-Service (MaaS), further research can build upon our work and concentrate on user-relevant marketplace aspects. The described runtime platform and its monitoring and adaptation components were implemented in a prototype and will be evaluated in near future regarding stability, performance and scalability. Following a collaborative approach, we offer several of the components for public use and evaluation[4]. The overall architecture of the service marketplace will be evaluated in collaboration with several project partners in the TEXO use case of THESEUS.

ACKNOWLEDGEMENT

The presented work is part of the research project THESEUS and will be evaluated in the use case TEXO. The project was funded by means of the German Federal Ministry of Economy and Technology under the promotional reference ”01MQ07012”.

REFERENCES

[Braun08] Braun I. et al, 2008. Zusicherung nichtfunktionaler Eigenschaften und Dienstgüte im Future Internet of Services. *In PIK - Praxis der Informationsverarbeitung und Kommunikation*, Vol. 31(04), pp 225–231.

[Halima08] Halima R. B. et al, 2008. Providing Predictive Self-Healing for Web Services: A QoS Monitoring and Analysis-based Approach. *In Journal of Information Assurance and Security (JIAS)*, Vol. 3(3), pp 175–184.

[Garlan01] Garlan D. et al, 2001. Using Gauges for Architecture-Based Monitoring and Adaptation. In *Working Conference on Complex and Dynamic Systems Architecture*, Brisbane, Australia.

[Kazhamiakin08] Kazhamiakin R., 2008. Adaptation and Monitoring in S-Cube: Global Vision and Roadmap. In *Proceedings of the Workshop on Monitoring, Adaptation and Beyond (MONA+)*, Madrid, Spain.

[Meyer07] Meyer H. et al, 2007. ASG–Techniques of Adaptivity. *Proceedings of Autonomous and Adaptive Web Services*, Dagstuhl, Germany.

[Michaut02] Michaut F. and Lepage F., 2002. An architecture for application adaptation based on QoS monitoring. In *IEEE-International Conference on Systems, Man and Cybernetics (SMC)*, Hammamet, Tunisia.

[Parkin08] Parkin M. et al, 2008. An SLA Re-Negotiation Protocol. *Proceedings of the 2nd Workshop on Non Functional Properties and Service Level Agreements in Service Oriented Computing (NFPSLA-SOC)*, Dublin, Ireland.

[Prieto06] Prieto A. G. and Stadler R., 2006. Adaptive Distributed Monitoring with Accuracy Objectives. *Proceedings of the 2006 SIGCOMM Workshop on Internet Network Management*, Pisa, Italy.

[Spillner09a] Spillner J. et al, 2009. Distributed Contracting and Monitoring in the Internet of Services. *Proceedings of the 9th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS)*, pp. 129-142, Lisbon, Portugal.

[Spillner09b] Spillner J. and Hoyer J., 2009. SLA-driven Service Marketplace Monitoring with Grand SLAM. *Proceedings of the 4th International Conference on Software and Data Technologies (ICSOFT)*, Vol. 2, pp. 71-74, Sofia, Bulgaria.

[Spillner09c] Spillner J. et al, 2009. Service Adaptivity through Cross-Domain Reconfiguration of Non-Functional Properties. In *Electronic Communications of the EASST*, Vol. 18/2009, pp. 17-22.

[Strunk09a]: Strunk A. et al, 2009. Supporting Rebinding in BPEL Processes. In *Proceedings of the 7th IEEE International Conference on Web Services (ICWS)*, LosAngeles, California, USA

[Strunk09b]: Strunk A. et al., 2009. An Infrastructure for Supporting Rebinding in BPEL Processes. In *1st Workshop on Service-Oriented Business Networks and Ecosystems (SOBNE '09)*, Auckland, New Zealand.

[4]The TECЛA Service Platform: http://www.serviceplatform.org/