Engineering of Next Generation Self-Aware Software Systems: A Research Roadmap

Samuel Kounev

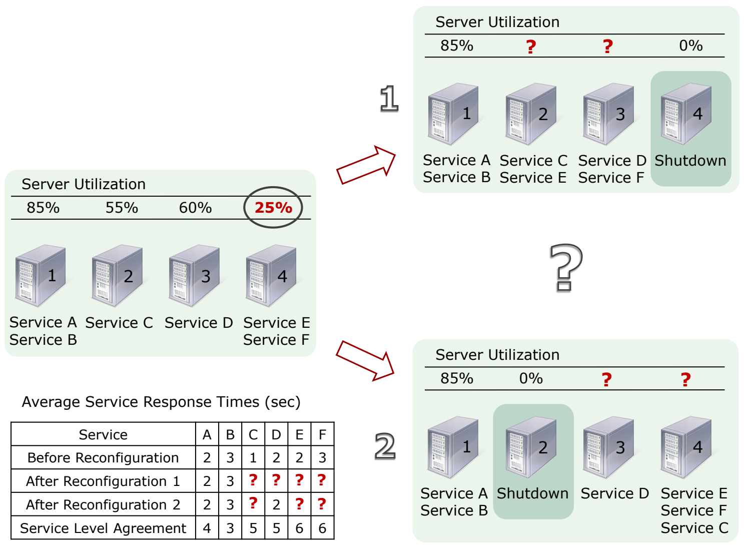
Institute for Program Structures and Data Organization (IPD)
Karlsruhe Institute of Technology (KIT)
76131 Karlsruhe, Germany
kounev@kit.edu

Abstract With the increasing adoption of virtualization and the transition towards cloud computing platforms, modern enterprise software systems are becoming increasingly complex and dynamic. The lack of direct control over the underlying physical hardware and the resulting gap between logical and physical resource allocations pose some major challenges in providing quality-of-service (QoS) guarantees. Due to the inability to automatically keep track of the complex interactions between the applications and workloads sharing the physical infrastructure, modern enterprise systems often exhibit poor QoS and resource efficiency, and have high operating costs. In this paper, we present a research roadmap and a long-term vision aiming to address these challenges. The presented research agenda is pursued by the Descartes Research Group at KIT which is funded by the German Research Foundation within the Emmy Noether Programme. Our long-term goal is to develop a novel methodology for engineering of next generation *self-aware* software systems. The latter will have built-in QoS models enhanced to capture dynamic aspects of the system environment and maintained automatically during operation. The models will be exploited at run-time to adapt the system to changes in the environment ensuring that resources are utilized efficiently and that QoS requirements are continuously satisfied.

1 Introduction

Modern enterprise systems based on the Service-Oriented Architecture (SOA) paradigm have highly distributed and dynamic architectures composed of loosely-coupled services that operate and evolve independently. Managing system resources in such environments to ensure acceptable end-to-end application quality-of-service (e.g., availability, performance and reliability) and efficient resource utilization is a challenge. The adoption of virtualization and cloud computing technologies, such as Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) and Infrastructure-as-a-Service (IaaS), comes at the cost of increased system complexity and dynamicity. The increased complexity is caused by the introduction of virtual resources and the resulting gap between logical and physical resource allocations. The increased dynamicity is caused by the complex interactions between the applications and workloads sharing the physical infrastructure. The inability to predict such interactions and adapt the system accordingly makes it hard to provide quality-of-service (QoS) guarantees in terms of availability and responsiveness, as well as resilience to attacks and operational failures. Moreover, the consolidation of workloads translates into higher utilization of physical resources which makes the system much more vulnerable to threats resulting from unforeseen load fluctuations, hardware failures and network attacks.

Service providers are often faced with questions such as: What QoS would a new service deployed on the virtualized infrastructure exhibit and how much resources should be allocated to it? What would be the effect of migrating a service from one virtual machine (VM) to another? How should the system configuration be adapted to avoid QoS issues or inefficient resource usage arising from changing customer workloads? Answering such questions requires the ability to predict at *run-time* how the QoS of running services and applications would be affected if the system configuration or the workload changes. We refer to this as *online QoS prediction*. Due to the inability to automatically keep track of dynamic changes in the system environment and predict their effect, SOA systems in use nowadays often exhibit poor QoS and resource efficiency, and have high operating costs. To accommodate load fluctuations, services are typically hosted on dedicated servers with over-provisioned capacity. Servers in data centers nowadays typically run at around 20% utilization [11] which corresponds to their lowest energy-efficiency region [2]. The growing number of under-utilized servers, often referred to as "server sprawl", translates into increasing data center operating costs including power consumption costs, cooling infrastructure costs and system management costs. To counter this development, novel methods for online QoS prediction and autonomic resource management are needed.



Service	A	B	C	D	E	F
Before Reconfiguration	2	3	1	2	2	3
After Reconfiguration 1	2	3	?	?	?	?
After Reconfiguration 2	2	3	?	2	?	?
Service Level Agreement	4	3	5	5	6	6

Figure 1. Online QoS Prediction Scenario

To illustrate how online QoS prediction can help to improve the system resource efficiency, we consider a simple example depicted in Figure 1. A SOA system made of four servers hosting six different services is shown including information on the average service response times, the response time service level agreements (SLAs) and the server utilization. Now assume that due to a change in the demand for services E and F, the average utilization of the fourth server has dropped down to 25% over an extended period of time. To improve the resource efficiency, it is considered to shut down one of the servers migrating its services to other servers. Two possible ways to reconfigure the system (shutting down the second and the fourth server respectively) are shown. To ensure that reconfiguring the system would not break the SLAs, the system needs a mechanism to predict the effect of the reconfiguration on the service response times. Given that this must be done at run-time, online QoS prediction capabilities are required.

In the rest of this paper, we present the research agenda and long-term vision of a research project carried out by the Descartes Research Group at KIT aiming to address the challenges described above.

2 Research Agenda and Long-Term Vision

The Descartes Research Group [1] at KIT was started in July 2009 and is funded by the German Research Foundation (DFG) within the Emmy Noether Programme. The group is working on novel approaches to software and systems engineering that ensure that non-functional QoS requirements are continuously satisfied during operation while at the same time infrastructure resources are utilized efficiently lowering the system TCO (Total-Cost-of-Ownership). The research areas and technology domains we are focusing on are depicted in Figure 2.

Our research is divided into three main areas: i) system design, measurement and analysis targeted at understanding the system behavior and the way it is influenced by the environment it is running in, ii) system modeling for QoS prediction both at system design-time and during operation, and iii) autonomic and self-adaptive system QoS management. The three areas are closely interrelated. On the one hand, building representative models requires deep understanding of the system behavior as well as measurement data to calibrate the models. On the other hand, methods for self-adaptive QoS management rely on predictive models that help to predict the effect of adaptation decisions at run-time.

Our long-term research agenda aims at developing a novel methodology for engineering of so-called *self-aware* software systems [7]. The latter will have built-in online QoS prediction and self-adaptation capabilities addressing the challenges described in Section 1. This vision is the major topic of our research group which is named after the French philosopher and mathematician René Descartes. Self-awareness in this context is meant in the sense that systems should be aware of changes that occur in their environment and should be able to predict the effect of such changes on their QoS (*"thought is what happens in me such that I am immediately conscious of it" – René Descartes*). Furthermore,

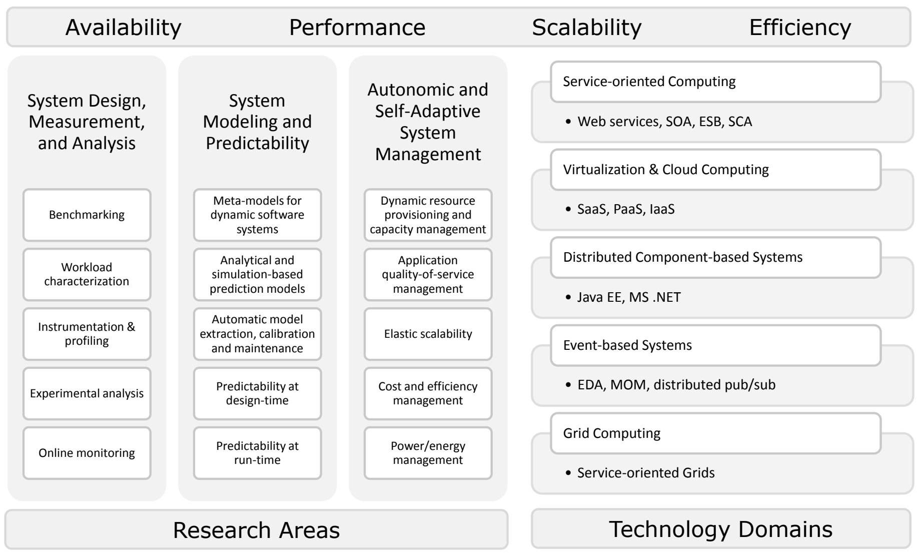

Availability
Performance
Scalability
Efficiency
System Design, Measurement, and Analysis
Benchmarking
Workload characterization
Instrumentation & profiling
Experimental analysis
Online monitoring
System Modeling and Predictability
Meta-models for dynamic software systems
Analytical and simulation-based prediction models
Automatic model extraction, calibration and maintenance
Predictability at design-time
Predictability at run-time
Autonomic and Self-Adaptive System Management
Dynamic resource provisioning and capacity management
Application quality-of-service management
Elastic scalability
Cost and efficiency management
Power/energy management
Service-oriented Computing
• Web services, SOA, ESB, SCA
Virtualization & Cloud Computing
• SaaS, PaaS, IaaS
Distributed Component-based Systems
• Java EE, MS .NET
Event-based Systems
• EDA, MOM, distributed pub/sub
Grid Computing
• Service-oriented Grids
Research Areas
Technology Domains

Figure 2. Research Areas and Technology Domains

systems should automatically adapt as the environment evolves in order to ensure that infrastructure resources are utilized efficiently and QoS requirements are continuously satisfied (*"for it is not enough to have a good mind: one must use it well" – René Descartes*). To realize this vision, we advocate the use of *dynamic QoS models* integrated into the system components and used at run-time for online QoS prediction. The models will serve as a "mind" to the system controlling its behavior, i.e., resource allocations and scheduling decisions. In analogy to Descartes' *dualism principle* (*"the mind controls the body, but the body can also influence the mind"*), the link between the QoS models and the system components they represent will be bidirectional.

The new dynamic QoS models will be designed to encapsulate all information, both static and dynamic, relevant to predicting a service's QoS on-the-fly. This includes information about the service's software architecture, its workload and its execution environment. Current architecture-level[1] performance models for component-based architectures, surveyed in [8], (e.g., PCM [3], CBML [12], CB-SPE [4]) will be used as a basis. The latter will be extended to capture the performance influences of the platforms used at each layer of the service execution environment focusing on the virtualization and middleware layers. Resource

[1] We distinguish between *descriptive* architecture-level QoS models and *predictive* QoS models. The former describe QoS-relevant aspects of software architectures and execution environments (e.g., UML models augmented with QoS-related annotations). The latter capture the temporal system behavior and can be used for QoS prediction by means of analytical or simulation techniques (e.g., Markov chains, layered queueing networks or stochastic Petri nets).

allocations at the different layers will be modeled explicitly and benchmark results will be exploited to quantify the relative performance of different execution platforms. This will make it possible to predict how the QoS of a running service will be affected if resource allocations are modified or if the service is migrated from one VM to another possibly running on a different platform.

Unlike conventional architecture-level QoS models, the developed models will be *dynamic* in the sense that they will be maintained and updated automatically to reflect the evolving system environment. To realize this, execution platforms should be enhanced with functionality to automatically extract and maintain the models during operation. Depending on the type of system considered and the availability of monitoring and instrumentation frameworks, the degree of automation of the initial model extraction will be different. For example, for a newly developed system, the model extraction could potentially be completely automated, whereas for legacy systems some manual steps might be required.

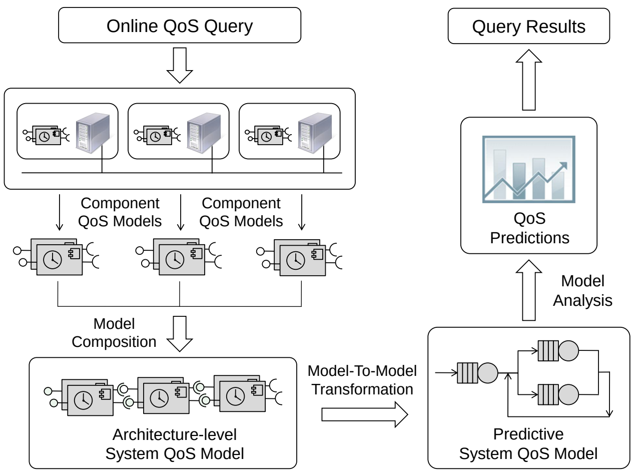


Figure 3. Online QoS Prediction Process

The dynamic QoS models will be used during operation to answer QoS-related queries such as: What would be the effect on the QoS of running applications if a new application is deployed in the virtualized environment or an existing application is migrated from one server to another? How much resources need to be allocated to a newly deployed application to ensure that service-level agreements (SLAs) are satisfied? What QoS would the system exhibit after a period of time if the workload continues to develop according to the current trends? How should the system configuration be adapted to avoid QoS problems or inefficient resource usage arising from changing customer workloads? We refer to such queries as *online QoS queries*.

Figure 3 illustrates the process that will be followed in order to provide an answer to a query. First, the QoS models of all involved system components will be retrieved and combined by means of model composition techniques into a single architecture-level QoS model encapsulating all information relevant to answering the QoS query. This model will then be transformed into a predictive QoS model by means of an automatic *model-to-model transformation*. Existing model-to-model transformations for static architecture-level performance models will be used as a basis, e.g., [3, 9]. The target predictive model type and level of abstraction as well as the solution technique will be determined on-the-fly based on the required accuracy and the time available for the analysis. Multiple model types (e.g., layered queueing networks, stochastic process algebras, queueing Petri nets and general-purpose simulation models) and model solution techniques (e.g., exact analytical techniques, numerical approximation techniques, simulation and bounding techniques) will be used in order to provide flexibility in trading-off between prediction accuracy and analysis overhead.

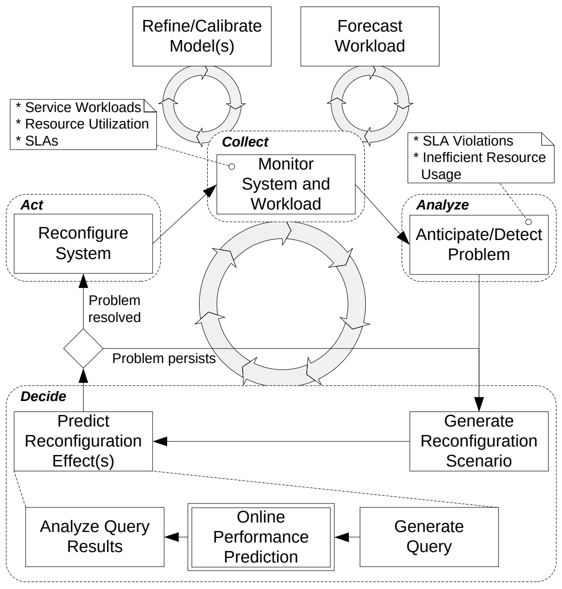

Refine/Calibrate Model(s)
Forecast Workload
* Service Workloads
* Resource Utilization
* SLAs
Collect
Monitor System and Workload
* SLA Violations
* Inefficient Resource Usage
Act
Reconfigure System
Analyze
Anticipate/Detect Problem
Problem resolved
Problem persists
Decide
Predict Reconfiguration Effect(s)
Generate Reconfiguration Scenario
Analyze Query Results
Online Performance Prediction
Generate Query

Figure 4. Online Reconfiguration Process

The ability to answer online QoS queries during operation will provide the basis for implementing techniques for self-aware QoS management. Such techniques will be triggered automatically during operation in response to observed or forecast changes in application workloads. The goal will be to proactively adapt the system to such changes in order to avoid anticipated QoS problems or inefficient resource usage. The adaptation will be performed in an autonomic fashion by considering a set of possible system reconfiguration scenarios (e.g, changing VM placement and/or resource allocations) and exploiting the online QoS query mechanism to predict the effect of such reconfigurations before making a decision. Figure 4 depicts the online reconfiguration process and the self-adaptation control loop. The latter is based on the generic model of a control loop from [6] which we have extended to integrate the use of the online QoS query mechanism. In addition to the main control loop, two additional loops are running in the background, one for continuously refining and calibrating online models and one for forecasting the workload evolution.

As an initial step towards the described vision, we conducted two preliminary case studies. In the first case study, we studied a complex Java EE application showing how detailed architecture-level performance models can be extracted and maintained automatically at run-time based on online monitoring data [5]. As a performance model we used the Palladio Component Model [3]. The extracted performance models provided performance predictions with less than 30% deviation from measurements on the real system. In the second case study, we studied a SOA application running in a service-oriented Grid computing environment [10] and showed how online performance models (based on hierarchical queueing Petri nets) can be used at run-time for autonomic QoS-aware resource allocation. The case studies demonstrate that the existing gap between architecture-level QoS models and run-time QoS management can be closed.

3 Concluding Remarks

Modern enterprise systems based on the SOA paradigm have highly distributed and dynamic architectures composed of loosely-coupled services often deployed in virtualized computing infrastructures. Managing system resources in such environments to ensure acceptable end-to-end application quality-of-service and efficient resource utilization is a challenge. In this paper, we presented a research roadmap and a long-term vision aiming to address this challenge. The presented research agenda is pursued by the Descartes Research Group at KIT which is working towards the development of a novel methodology for engineering of next generation *self-aware* software systems. Such systems will be aware of their QoS and the way it is affected by the environment they are running in. Moreover, they will automatically adapt as the environment evolves ensuring that system resources are utilized efficiently and QoS requirements are continuously satisfied. The described approach raises several big challenges that represent emerging hot topics in the software engineering community and will be subject of long-term fundamental research in the years to come. The resolution of these challenges promises to revolutionize the field of systems engineering and radically transform the way enterprise systems are built and managed. The introduced self-awareness and autonomous management will provide a number of benefits such as better quality-of-service, lower operating costs and improved energy efficiency.

4 Short Biographical Sketch

Dr.-Ing. Samuel Kounev is head of the Descartes Research Group at Karlsruhe Institute of Technology (KIT). He received a MSc degree in mathematics and computer science from the University of Sofia (1999) and a PhD degree in computer science from Technische Universität Darmstadt (2005). From February 2006 to May 2008, he was a research fellow at Cambridge University working in the systems research group of Prof. Jean Bacon at the Computer Laboratory. In April 2009, he received the Emmy-Noether Career award from the German Research Foundation (DFG) and started the Descartes Research Group at KIT. Dr. Kounev's research is focused on methods for autonomic management of system quality-of-service (e.g., availability, performance and reliability) and resource efficiency (e.g., energy consumption) throughout the system life-cycle. Dr. Kounev is founder of the SPEC International Performance Evaluation Workshop (SIPEW) and co-founder of the ACM/SPEC International Conference on Performance Engineering (ICPE). He served as a BEA Technical Director from 2004 to 2008 and as a release manager of SPEC's Java Subcommittee from 2003 to 2009. During this time he led the development and standardization of several widely adopted industry-standard benchmarks including SPECjAppServer2004, SPECjms2007 and SPECjbb2005. He is a member of the ACM, IEEE, and the GI e.V.

References

1. Descartes Project Website. http://www.descartes-research.net, 2010.
2. L. A. Barroso and U. Hölzle. The Case for Energy-Proportional Computing. *IEEE Computer*, 40(12):33–37, 2007.
3. S. Becker, H. Koziolek, and R. Reussner. The Palladio component model for model-driven performance prediction. *Journal of Syst. and Softw.*, 82:3–22, 2009.
4. A. Bertolino and R. Mirandola. Software Performance Engineering of Component-based Systems. In *Proc. of WOSP-2004*. ACM Press, 2004.
5. F. Brosig, S. Kounev, and K. Krogmann. Automated Extraction of Palladio Component Models from Running Enterprise Java Applications. In *Proc. of ROSSA-2009*. ACM, 2009.
6. B. H. C. Cheng, R. de Lemos, H. Giese, P. Inverardi, and J. Magee. Software Engineering for Self-Adaptive Systems: A Research Roadmap. In *Software Engineering for Self-Adaptive Systems, LNCS 5525*, pages 1–26, 2009.
7. S. Kounev, F. Brosig, N. Huber, and R. Reussner. Towards self-aware performance and resource management in modern service-oriented systems. In *Proceedings of the 7th IEEE International Conference on Services Computing (SCC 2010), July 5-10, Miami, Florida, USA*. IEEE Computer Society, 2010.
8. H. Koziolek. Performance evaluation of component-based software systems: A survey. *Perform. Eval.*, In Press, 2009.
9. H. Koziolek and R. Reussner. A Model Transformation from the Palladio Component Model to Layered Queueing Networks. In *Proc. of SIPEW-2008*. Springer LNCS 5119, 2008.
10. R. Nou, S. Kounev, F. Julia, and J. Torres. Autonomic QoS control in enterprise Grid environments using online simulation. *Journal of Systems and Software*, 82:486–502, 2009.
11. K. Parent. Server Consolidation Improves IT's Capacity Utilization. Vol. 2006: Court Square Data Group, 2005.
12. X. Wu and M. Woodside. Performance Modeling from Software Components. In *Proc. of WOSP-2004*, pages 290–301. ACM Press, January 2004.