Volume 6, Issue 4, April 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

# Research Roadmap for Next Generation Self Aware Software System

**Manas Kumar Yogi*, K. Chandrasekhar, G. Vijay Kumar**
CSE Department, PEC, Surampalem,
A.P., India

***Abstract*— For providing effective QOS the gap between logical & physical resources allocations should decrease else QOS will decrease. In this paper we present the vision of designing a self-aware-system in such a way that QOS won't be affected in a negative way. Self-aware systems are designed to have built in models of QOS to capture dynamic aspects of the system .Resource utilization without effecting the service level agreement is yet another inherent characteristic of the self-aware systems. As we move in future with cloud computing paradigms, this paper traces the research roadmap for systems which are self aware.***

***Keywords*— Self Aware, QoS, Distributed, Prediction, Virtualization**

## I. INTRODUCTION

Modern enterprise system 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 Infrastrcture-as-a-Service (IaaS),comes at the cost of increased system complexity, and dynamicity. 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 resources 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 which corresponds to their lowest energy-efficiency region. 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 resources management are needed.

To illustrate how online QoS prediction can help to improve the system resource efficiency, we consider a simple example depicted in Fig 1. A SOA system made of four services hosting six different services is shown including information on the average service response times, the response service level agreements(SLAs) and the server 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 shutdown one of the server 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 if 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.

 


Average Service Response Times (sec)

| 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 |

*! = undecidable factor

Fig. 1.Prediction Scenario for Online QoS

## II. RESEARCH AGENDA AND LONG TERM VISION

The Descartes Research Group at KIT was started in july 2009 and is funded by the German Research Foundation (DFG) within the Emmy No ether 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 Fig 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, methods for self-adaptive QoS management rely on predictive models that help to predict the effect of adaptation decision at run time.

Our longterm research agenda aims at developing a novel methodology for engineering of so-called self-aware software systems. The latter will have built-in online QoS prediction and self-adaptation capabilities addressing the challenges describeded in the section 1.This vision is the major topic of our research group which is named after the French philosopher and mathematician Rene Descartes. Self-awareness in this context is meant in the sense that system should be aware of changes that occur in their environment and should be predict the effect of such changes on their QoS ("thought is what happends in me such that I am immediately conscious of it"-Rene Descartes).

 

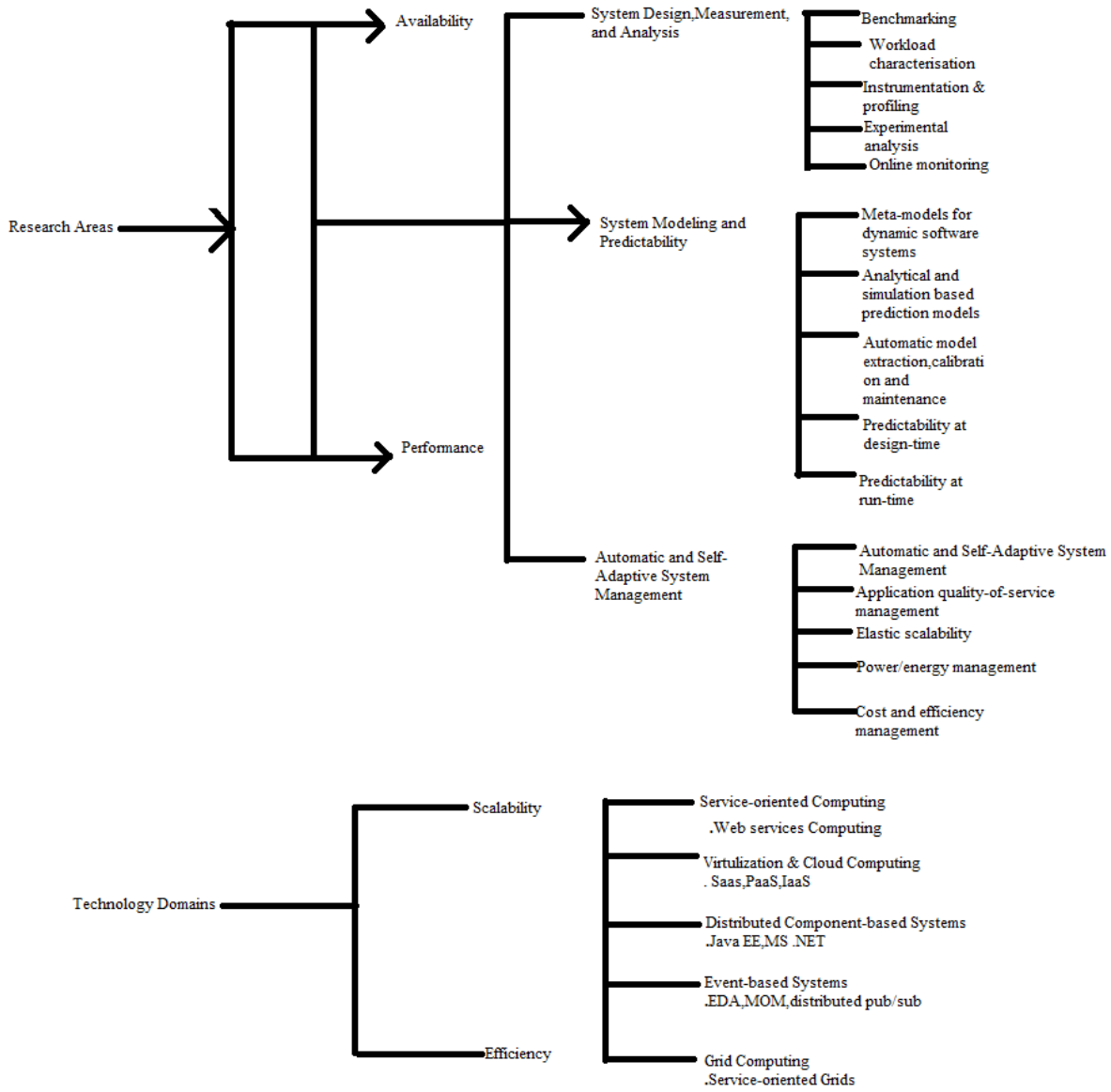


Fig. 2. Research Areas and Technology Domains

Furthermore,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" – Rene Descartes).To realize this vision,we advocate the use of dynamic QoS models integrated into the system components and used at runtime 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 software architecture,its workload and its execution environment.Current architecture-level performance models for component-based architectures, surveyed in (e.g., PCM, CBML, CB-SPE) 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 allocations at the different layers will be modeled explicitly and benchmark results will be exploited to qualify the relative performance of different execution platforms.This will be exploited to qualify 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.

 

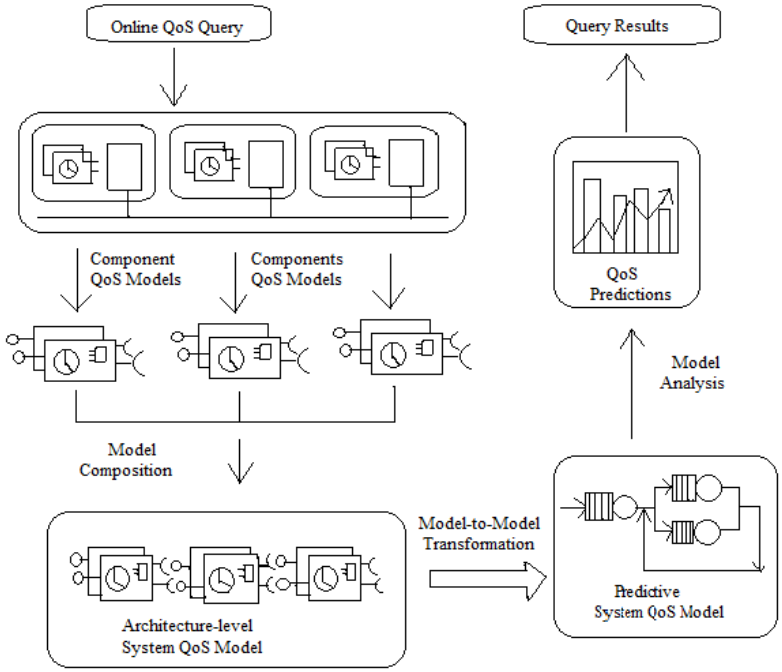


Fig 3. Online QoS Prediction Process

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 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.

Fig.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 archtecture-level performance models will be used as a basis,e.g.,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, queuing 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.

Algorithm for answering online QoS Queries:

Step 1 : Construction of single architecture level QoS model from all relevant QoS models of components.
Step 2 : This model is transformed into a predictive QoS model.
Step 3 : Collection of query from input component.
Step 4 : Monitoring system workload and depending on SLA's reconfiguring the process
Step 5 : Considering the effect of reconfiguration without effecting the QoS and resolving the query.
Step 6 : If problem is resolved, reconfigure the system into original state else go to step 4.

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 h 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 .

As an intial 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 architechture-level performance models can be extracted and maintained automatically at run-time based on online monitoring data.As a performance model we used the Palladio Component Model.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 and showed how online performance models(based on hierarchical queuing Petri nets) can be used at run-time for automatic 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.

 

## III. CONCLUSION

This paper presents a research roadmap to address the challenge for managing resources in dynamic architectures. These architectures have loosely coupled systems deployed in virtualized computing infra structure. We present a novel approach to resolve the challenges posed by QOS parameters as they are highly constrained. Self-aware systems are indispensible part of modern enterprise systems as they provide better QOS, operating costs are lowered & energy efficiency is improved.

## REFERENCES

[1] Descartes Project Website. http://www.descartes-research.net,2010.
[2] L.A. Barroso and U. Holzle.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.

 