Parameter Dependent Performance Specifications of Software Components*

Heiko Koziolek, Jens Happe, and Steffen Becker

Graduate School Trustsoft
University of Oldenburg Germany
Chair for Software Design and Quality
University of Karlsruhe, Germany
{koziolek, happe, sbecker}@ipd.uka.de

Abstract. Performance predictions based on design documents aim at improving the quality of software architectures. In component-based architectures, it is difficult to specify the performance of individual components, because it depends on the deployment context of a component, which may be unknown to its developers. The way components are used influences the perceived performance, but most performance prediction approaches neglect this influence. In this paper, we present a specification notation based on annotated UML diagrams to explicitly model the influence of parameters on the performance of a software component. The UML specifications are transformed into a stochastical model that allows the prediction of response times as distribution functions. Furthermore, we report on a case study performed on an online store. The results indicate that more accurate predictions could be obtained with the newly introduced specification and that the method was able to support a design decision on the architectural level in our scenario.

1 Introduction

Performance is an important quality attribute of a software architecture. It can by characterised by metrics such as response time, throughput, and resource utilisation. In many existing systems, the reason for bad performance is a poorly designed software architecture [15]. Performance predictions based on architectural descriptions of a software system can be performed before the implementation starts, thereby possibly reducing costs for subsequent refactorings to fix performance problems. It is the hope that such early analyses support the decision for design alternatives and reduce the risk of having to redesign the architecture after performance problems have been diagnosed in the implementation.

Component-based software architectures are well-suited for early performance predictions, if information needed for performance evaluation has been specified for each component by its developers. As component developers cannot know in which context their components will be deployed [8], these performance specifications should be parameterisable for different hardware resources, required services, and usage contexts to allow accurate predictions [3].

* This work is supported by the German Research Foundation (DFG), grant GRK 1076/1.

C. Hofmeister et al. (Eds.): QoSA 2006, LNCS 4214, pp. 163–179, 2006.

In many performance prediction approaches, the component specifications are parameterisable for different usage contexts only by allowing to specify probabilities for the possible requests to the component's provided services (e.g., [4,17]). It is often neglected that component services can be called with different parameters, and that these parameter can have a significant influence on the performance of the architecture.

The dependencies between parameters of a component service and its performance have to be made explicit by *component developers* in the specifications to allow accurate performance predictions. *System architects* can then adjust performance predictions to the expected usage profile. In some cases, the dependencies between parameters and performance might be intricate and hard to specify, for example if a service first performs complex computations on a parameter value an then changes its performance depending on the results. Furthermore, parameters might be complex objects or even other components, for which a reasonable specification is difficult. However, in this paper a first step to integrating parameters into performance specifications of software components shall be taken. Parameters considered here can be of a primitive or composite data type.

A notation based on extensions to the UML SPT profile [12] is provided to specify the dependencies between parameters and performance. This profile allows annotating UML diagrams with performance-related information. As many existing performance approaches already use this profile (e.g., [4,11]), they could also benefit from the extensions presented in this paper. Tools evaluating annotated UML diagrams could be changed with low effort to incorporate the extensions. Another advantage of the UML-based notation is the familiarity of the developers, who often already know the UML language. However, the concepts underlying the approach presented here are not bound to the SPT profile and might be carried over to other notations (e.g., future performance related profiles).

The contribution of this paper is a modelling notation for parameter dependent performance specifications for software components and an according analytical performance prediction model. Unlike most performance prediction models, we explicitly incorporate the influence of parameters on resource demand as well as on the usage of external services in our predictions. A case study, in which response times for a design alternative of a component-based software architecture are predicted, is provided to illustrate the benefits of this approach.

The paper is organised as follows: Section 2 describes the modelling in our performance prediction approach and focuses on parameter dependencies. Section 3 shows the necessary computations, and Section 4 explicitly lists the assumptions underlying the approach. The case study of a performance prediction for an component-based online store is provided in Section 5. Section 6 points out related work, while Section 7 draws conclusions and sketches future work.

2 Modelling Component Performance

Several models from different developer roles are used for the prediction of the performance in a component-based architecture in our approach. The architecture itself is modelled with a UML component diagram by the system architect. For each