MEMS: A Method for Evaluating Middleware Architectures

Yan Liu[1], Ian Gorton[1], Len Bass[2], Cuong Hoang[3], and Suhail Abanmi[1]

[1] National ICT Australia (NICTA), Australia[1],
& School of Computer Science and Engineering, University of New South Wales, Australia
{jenny.liu, ian.gorton, suhail.abanmi}@nicta.com.au
[2] Software Engineering Institute, USA
ljb@sei.cmu.edu
[3] Engineering Faculty, University of Technology Sydney
vietcuong.hoang@student.uts.edu.au

Abstract. Middleware architectures play a crucial role in determining the overall quality of many distributed applications. Systematic evaluation methods for middleware architectures are therefore important to thoroughly assess the impact of design decisions on quality goals. This paper presents MEMS, a scenario-based evaluation approach. MEMS provides a principled way of evaluating middleware architectures by leveraging generic qualitative and quantitative evaluation techniques such as prototyping, testing, rating, and analysis. It measures middleware architectures by rating multiple quality attributes, and the outputs aid the determination of the suitability of alternative middleware architectures to meet an application's quality goals. MEMS also benefits middleware development by uncovering potential problems at early stage, making it cheaper and quicker to fix design problems. The paper describes a case study to evaluate the security architecture of grid middleware architectures for managing secure conversations and access control. The results demonstrate the practical utility of MEMS for evaluating middleware architectures for multiple quality attributes.

1 Introduction

Middleware refers to a broad class of software infrastructure technologies that use high-level abstractions to simplify construction of distributed systems. This infrastructure provides a distributed environment for deploying application-level components. These application components rely on the middleware to manage their life cycles and execution, and to provide off-the-shelf services such as transactions and security.

Consequently, the application component behavior and the middleware architecture are tightly coupled, and the middleware plays a critical role in achieving the

[1] National ICT Australia is funded through the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council.

C. Hofmeister et al. (Eds.): QoSA 2006, LNCS 4214, pp. 9 – 26, 2006.

quality attribute requirements of distributed applications. If the middleware architecture is poorly designed or implemented, contains subtle errors, or is inefficient or lacking in features, it may eventually lead to the failure of applications to meet their requirements.

It would therefore be valuable to have an evaluation method for designers of middleware-based applications to rigorously assess a technology and determine its fitness for purpose for an application. Such a method would also benefit the middleware developer community, who could use this approach to uncover potential design and implementation problems in their platforms.

Middleware creates new challenges and issues for software architecture evaluation methods. First, middleware technologies are *horizontal* in nature, providing mechanisms for a wide range of applications in many vertical application domains. The business goals for an application from the stakeholder's perspective are hence more likely to address the domain-specific application behavior rather than the specific requirements for the middleware itself. This indicates that evaluation methods for middleware should be driven by the concerns of individual quality attributes, instead of specific business goals.

Second, the ability of a middleware technology to support given quality attributes depends on the mechanisms and services provided by the infrastructure. Many competing middleware products exist, and they provide different mechanisms to address the same quality attribute. This indicates that evaluation methods require detailed technical inputs regarding the middleware infrastructure, including its programming model, APIs, configuration and deployment. This kind of knowledge helps to identify the effect of different middleware architectures on quality attributes.

Third, middleware technologies are becoming more and more complex. They typically have several thousand API calls and a collection of integrated services and tools of varying importance levels to different applications. This makes it difficult to evaluate the quality of a complete middleware technology. This requires evaluation methods to be flexible, and able to quickly provide feedback on alternative middleware architectures with respect to multiple quality attributes.

Finally, access to an adequate range of stakeholders for middleware systems is usually not possible. Stakeholders are used in many methods to prioritize requirements and to propose additional scenarios. When evaluating middleware, the developers of the middleware are usually not available. Also, the business goals for the middleware are not going to be represented by stakeholders associated with the application being constructed.

Software architecture evaluation methods exists in the literature and a comprehensive review can be found in [1] [8]. However, these methods are applied at the application level, where middleware is considered as features and constraints of the implementation of the applications. The strong relationship between a middleware technology and the quality of the overall system, and the issues of evaluating middleware architectures discussed above are not explicitly addressed. To the best of our knowledge, there has been little work on devising a systematic approach to evaluate the quality of middleware technologies.

In this paper, we present a structured approach to address the evaluation of middleware architectures, called MEMS. MEMS is a scenario-based evaluation approach with a clearly defined set of steps that helps evaluators determine the suitability of