Enabling the Selection of COTS Components

Sudipto Ghosh, John L. Kelly, and Roopashree P. Shankar

Department of Computer Science,
Colorado State University,
Fort Collins, Colorado 80523
{ghosh, jkelly, roopaps}@cs.colostate.edu

Abstract. Ensuring proper selection of COTS components is key to the success of component-based software development approaches. Although several approaches and criteria have been proposed for component selection, we lack techniques that can be used to systematically evaluate components against selection criteria for functionality, security, fault tolerance, and quality attributes. We propose a comprehensive approach for enabling the selection of COTS components by employing component understanding and fault injection testing techniques that aid in building an integrated comprehension model of the components. This model accumulates information regarding how each candidate component fared with respect to each criterion. This model can be used not only to aid in the final decision making process, but also serve as a guide during the component comprehension and evaluation stages.

Keywords: COTS, components, comprehension model, fault injection testing, fault tolerance, selection, security.

1 Introduction

The success of component-based software development (CBSD) approaches depends heavily on the use of systematic techniques for selecting COTS components during application integration. "Evaluation and selection of the most appropriate COTS product has a tremendous impact on the subsequent processes and products of software development and evolution [1]." The selected components affect not only the correctness, quality and dependability of the application, but also the cost and quality of the process and future maintenance activities. COTS components also increase the liability of the application developer.

Component selection involves the identification of appropriate selection criteria, assigning scores to alternative components for each criterion, and then making a decision on which component best meets the criteria [2]. However, component selection is usually difficult, mostly owing to the black-box nature of COTS components, and the often incomplete or imprecise documentation accompanying them. Not much is known about the internals of the COTS components and how they fit with original requirements and the remaining software system. Application developers who use COTS components are faced with a large

X. Franch and D. Port (Eds.): ICCBSS 2005, LNCS 3412, pp. 122–131, 2005.

number of requirements and constraints regarding the functionality, quality, security and fault tolerance requirements of the product. Since COTS component specifications do not necessarily match the exact requirements of the system, developers need to wrap the components to limit the functionality and range of inputs and outputs. Writing wrapper code and component integration requires significant effort. Moreover, components tend to evolve along with applications, thereby breaking existing versions of the assembled applications and increasing maintenance costs. Therefore, care must be taken during the selection of components to meet both short term and long term business objectives.

Although several approaches have been proposed for component selection [1], we lack techniques that can be used to systematically evaluate components against selection criteria for functionality, security, fault tolerance, and quality attributes. In this paper we present an approach for understanding the fault tolerance properties of Java components using the fault injection technique. We propose novel techniques for fault injection in Java components using: (1) aspect technologies, and (2) byte-code level manipulation.

We also propose the use of an integrated component comprehension model that accumulates all the information regarding how each component under consideration fared with respect to each selection criterion. This model can be used not only to aid in the final decision making process, but also to serve as a guide during the component comprehension and evaluation stages. We illustrate the use of the component comprehension model to understand the functionality of calendar components implemented in Java and downloaded from the world wide web. We explain the role of the component comprehension model in the overall component selection process.

2 Background

We see several important areas of existing work: (1) component selection and decision support approaches and selection criteria, and (2) testing approaches in component-based development. The first examines the overall selection process and general criteria used in selecting COTS components. The second examines strategies used in testing software components to understand their functionality and their effect on the remainder of the system.

2.1 Component Selection Approaches and Criteria

A number of approaches have been proposed to address selection of COTS components [1]. Examples are OTSO, PORE, CEP, CAP, CRE, QESTA, Storyboard, STACE, PECA, and combined selection of COTS components. Our goal is not to define a completely new approach. Instead, we describe an approach for evaluating components based on specific selection criteria related to functionality, component quality attributes, security and fault tolerance. The information obtained from the evaluation process can be used in any of the above approaches to perform an overall evaluation.