Reviewing the Challenges of Security Features in Component Based Software Development Models

Hasan Kahtan, Nordin Abu Bakar, Rosmawati Nordin
Department of Computer Science
Faculty of Computer and Mathematical Sciences
Universiti Teknologi MARA
40450 Shah Alam, Selangor
Malaysia
hassan.kahtan@gmail.com, {nordin, roswati}@tmsk.uitm.edu.my

Abstract—Component-based software development (CBSD) is an emerging technology that focuses on building systems by integrating existing software components. CBSD offers a range of benefits, such as an improved ability to reuse existing codes, reduced development costs of high-quality systems and faster development time. Several earlier studies have reported different challenges involved with using CBSD in software development. This paper therefore presents a review of these challenges and on existing CBSD models. A thorough Systematic Literature Review (SLR) was conducted and the outcome of this review highlights the strengths and weaknesses of each model, which then serves as the basis for future research.

Keywords- Component-based software development CBSD, Reusability, Software Security, Dependability, Reliability, Safety.

I. INTRODUCTION

Component Based Software Development (CBSD) is a technique which focuses on using existing software codes to develop software applications thereby replacing the need to develop from scratch. The idea is to assemble software applications from reusable software codes therefore simplifying software development in terms of time and budget constraints. Reusable components can be found free of charge (Open Source Software Components) or purchased from third party vendors (Commercial Off-the-shelf COTS) or constructed in-house. The major difference between CBSD and conventional software development is that conventional methods ignore the challenges and risks of incorporating reusable components. Perfect components selected in the early phase of the development life cycle may lead to incompatibility problems at the deployment phase therefore requiring modification and adjustment as well as suitable processes.

CBSD offers a range of benefits such as: managing increasing complexity [1,2,3], improving the ability to reuse components [4,5,3,6,1], improving efficiency [7,8,9], decreasing the time and effort needed to develop software [10, 4, 5, 11], improving the quality of the system [4,5,12], reducing maintenance costs [14,5,6], increasing development productivity [15,12,16,13], ensuring a greater degree of consistency [17,18,19], providing a wider range of usability [17,20,21] and supporting the effective use of specialists [22,14,23].

This paper discusses the challenges of CBSD in the context of software security. Articles that report on software development using component-based and reusable software are reviewed. Towards the end, a review of the strengths and weaknesses of existing CBSD models are also presented.

II. CHALLENGES OF CBSD

Several studies have reported different challenges involved in using CBSD. According to Moradianin [24], the interdependencies of software components create problems at the integration phase. Therefore, security features of software components must be considered and evaluated earlier in the CBSD lifecycle. Some examples of these security features are [25]:

- Dependability - the degree to which the software is operable and capable of performing functionality.
- Reliability - the continuity of correct service.
- Integrity - the absence of improper system alterations.
- Safety - the absence of catastrophic consequences on the user(s) and the environment.
- Maintainability - the ability to undergo modifications and repairs.

Pervasive computing raises major concerns regarding the ability of current development models to develop dependable systems. Since CBSD is an approach to software engineering [14], an important question to consider is: can CBSD be used to develop software applications which are provably dependable? In order to answer this question, it is necessary to identify whether the CBSD has fulfilled its initial intended purpose, or not. The challenges focused in this paper can be categorized as follows:

A. Development Challenges

As components and applications have different types of requirements and separate development lifecycles, there are