A Self-healing Approach to Designing and Deploying Complex, Distributed and Concurrent Software Systems*

Onn Shehory

IBM Haifa Research Lab, Mount Carmel
31905 Haifa, Israel
Onn@il.ibm.com

Abstract. Software systems have become too complex to manage and fix manually. An emerging approach to overcome this problem is software self-healing. Studies in related disciplines have offered several self-healing solutions, however these usually address a single class of problems, or they are not applicable in fielded systems. To address the industrial need for software self-healing, we have initiated the EU SHADOWS project. This project concentrates on self-healing of complex systems, extending the state-of-art in several ways. It introduces pioneering technologies to enable the systematic self-healing of classes of failures which are not solved by other approaches. It additionally introduces an approach to the integration of several self-healing technologies in a common solution framework. It also adopts a model-based approach, where models of desired software behavior direct the self-healing process. Combined, these allow for lifecycle support of software self-healing, applicable to both academic and industrial systems.

Keywords: Complex software systems, Self-healing, Autonomic computing.

1 Introduction

Our society has become dependent on the correct functioning of complex software systems. Classical software assurance methods are limited in their ability to address the increasing scale—in both size and complexity—of contemporary computer systems. This negatively affects system quality and total cost of system ownership. Recent research on self-management and autonomic computing introduces new possibilities for developing highly-reliable self-managed complex software systems, while reducing cost-of-ownership. Computational systems able to manage themselves are commonly called *self-** systems. Current research in this area is still in its early stages. We believe that independent solutions for self-* systems cannot be applied straightforwardly in industrially relevant systems, and that industrially applicable solutions must integrate several technologies in a common framework.

Self-healing refers to the capability of a software system to automatically diagnose and heal the root-cause of its failures and performance problems, and prevent them

* This research is funded in part by the European Commission via the SHADOWS project under contract No. 035157.

R.H. Bordini et al. (Eds.): ProMAS 2006, LNAI 4411, pp. 3–13, 2007.

from reappearing. This is performed through structural modifications of the software system and its operating parameters. We have initiated an EU project (SHADOWS [1]) that concentrates on the self-healing of complex systems, extending the state-of-art in several ways. Firstly, it introduces pioneering technologies to enable systematic self-healing of classes of failures which are not solved by other approaches. Additionally, it integrates several technologies which address different classes of problems and work at different abstraction levels, to provide a common solution framework. Further, it implements a model-based approach, where models of desired software behavior govern the self-healing process throughout the system design phase and the system deployment phase.

The SHADOWS project consists of nine partners. Some partners are research organizations, whereas others are industrial software developers. This combination allows the research partners not only work together to create a unified self-healing solution, but also to validate the solution in the field, in several different application domains, thus delivering a higher-quality solution.

At the time of writing this document, the project has been running for only a few months. Therefore, this article will introduce the background, the problem, the underlying technologies, and the approach and directions taken, however concrete results are not available yet. Intermediate results of the project will be posted at its web site [1].

2 Problem Statement

Classical software assurance introduces two main classes of techniques for increasing system reliability [2][3]. The first class includes rigorous development and testing methodologies; these increase the quality of single components and of their integration [4][5]. The second class includes fault tolerance mechanisms [6][7], which can guarantee the reliability of the system in the presence of faults. The classical methods have proven very useful in the past, and are still widely used. However, they do not scale well to address the increase in size and complexity of contemporary computer systems. This negatively affects the total cost of system ownership. To address the limited capability of classical techniques, self-management and autonomic computing [8][9] methods are being studied. These new research directions introduce new possibilities for developing highly-reliable self-managed complex software systems, while reducing cost-of-ownership.

Self-management of computational systems has already been envisioned in the past, notably by Turing and von-Neumann [10][11]. Only recently, as a result of market needs, attention to self-management has returned. Self-managed systems are commonly denoted as *self-** systems; the asterisk may indicate a variety of attributes, e.g., self-awareness, self-configuration, self-diagnosis etc. Current research in this area has explored multiple directions, approaches and problems, however it is still in its early stages and has focused mostly on academic settings. Among the approaches to self-management one finds reconfigurable architectures, consistency management, code relocation, control theory, agent-based systems, game theory, and others [12][13][14][15][16][17][18][19][20]. Below we briefly survey the primary research directions in the field of self-healing.