Computing (2013) 95:165–166
DOI 10.1007/s00607-012-0238-4

Editorial

Radu Calinescu · Robert France · Carlo Ghezzi

Published online: 18 November 2012

Modern software systems are often complex, inevitably distributed, and operate in heterogeneous and highly dynamic environments. Examples of such systems include those from the service-oriented, cloud computing, and pervasive computing domains. In these domains, continuous change is the norm. Changes may occur in the requirements or in the environment in which the software is embedded. They are difficult to predict and are often outside the control of the application. Furthermore, their occurrence may be disruptive, and therefore the software must also change accordingly. In many cases, the software is required to self-react by adapting its behavior dynamically, in order to ensure required levels of service quality in changing environments. In summary, these systems are required to be adaptable, flexible, reconfigurable and, increasingly, self-managing.

It has been recognized that in order to support dynamic adaptation and self-* (autonomic) behaviors, appropriate mechanisms should be available for runtime monitoring and on-line validation. Although significant research has been invested and matured in the area of model-driven software development, more recent research has focused on extending the applicability of model-driven techniques to the monitoring and analysis of run-time behavior. Research in this area aims to provide sound foundations that support dynamic adaptation and self-* behaviors.

R. Calinescu (✉)
University of York, York, North Yorkshire, UK
e-mail: radu.calinescu@york.ac.uk

R. France
Colorado State University, Fort Collins, USA
e-mail: france@cs.colostate.edu

C. Ghezzi
Politecnico di Milano, Milano, Italy
e-mail: carlo.ghezzi@polimi.it

This special issue of Springer's Computing contains three papers that reflect the state-of-the-art research on the use of models at runtime. In the paper titled "The Role of Models@run.time in Supporting On-the-Fly Interoperability", the authors, N. Bencomo, A. Bennaceur, P. Grace, G. Blair, and V. Issarny, discuss how models at runtime can be used to realize a new form of middleware, called emergent middleware, that resolves interoperability problems that arise in distributed systems. Runtime models are used to capture relevant information about the networked systems that need to interoperate. These models are then used to synthesize software mediators. The work in this paper is novel in that it describes how runtime models can be used to generate new software artefacts at runtime.

The paper titled "Detecting Component Changes at Run Time with Behavior Models", by Andrea Mocci and Mario Sangiorgio, describes an approach to inferring models of component behavior from runtime behaviors. These models are then used to detect changes in a component behavior. The approach provides a solution to the problem of detecting changes to component behaviors when there are no formal specifications against which runtime behaviors can be checked; such specifications are needed to support automatic change detection. The approach is applied to a real-world communication protocol.

The paper titled "Monitoring and Recovery for Web Service Applications", by Jocelyn Simmonds, Shoham Ben-David, and Marsha Chechik, describes a model-based approach to monitoring the runtime behavior of web services. The runtime models are finite state automata that specify forbidden and desired interactions between services. Execution traces expressed in BPEL are checked again the model at runtime to detect divergent behavior. When violations are detected the system synthesizes and presents proposed plans for resolving the violations. The generated plans are guaranteed to be correct.

These papers present novel, diverse, and interesting uses of models at runtime. We hope you enjoy reading these articles, and that they inspire you to further explore this very promising research area.