Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 189 (2007) 1–3
www.elsevier.com/locate/entcs

Preface

This special issue contains extended versions of a selection from the papers presented at WCAT'06, the Third International Workshop on Coordination and Adaptation Techniques for Software Entities, that was held in Nantes (France) on July 4, 2006. WCAT'06 was a satellite event of the 20th European Conference on Object-Oriented Programming, ECOOP 2006. The workshop provided a venue where researchers and practitioners on these topics could meet, exchange ideas and problems, identify key issues related to coordination and adaptation, and explore together and disseminate possible solutions. The website of the WCAT workshop series can be found at http://wcat.unex.es.

Motivation

Coordination and adaptation are two key issues when developing complex systems. *Coordination* focuses on the interaction among computational entities. *Adaptation* focuses on solving problems due to the interaction of mismatching entities. In the recent years, the need for more and more complex software, supporting new services and for wider application domains, together with the advances in middleware technologies, have promoted the development of distributed systems. The applications running on those systems are constituted of a collection of interacting entities (either considered as subsystems, modules, objects, components, or more recently web services) that collaborate to provide the required functionality.

One of the most complex tasks when designing and constructing such applications is not only to specify and analyze the coordinated interaction that occurs among the computational entities but also to be able to enforce them out of a set of already implemented (local) behaviours (i.e.: the computational entities). This fact has favoured the development of a specific field in Software Engineering devoted to the coordination of software. Such discipline, covering coordination models and languages, promotes the reusability both of the coordinated entities and of the coordination patterns that bind them together.

In fact, the ability to reuse existing software has always been a major concern of Software Engineering. In particular, the need for reusing and integrating heterogeneous software parts is at the root of the so-called Component-Based Software

1571-0661/$ – see front matter
doi:10.1016/j.entcs.2007.05.044

Development. The paradigm write once, run everywhere is currently claimed supported by several component-oriented platforms offered by the software industry.

However, a serious limitation of available component-oriented platforms (with regard to reusability) is that they do not provide suitable means to describe and to reason about the interacting behaviour of component-based systems. Indeed, while these platforms provide convenient ways to describe the typed signatures of software entities via interface description languages (IDLs), they offer a quite limited and low-level support to describe their concurrent behaviour. As a consequence, when a component is going to be reused, one can only be sure that it provides the required signature based interface, but little else can be inferred about the behaviour of the component with regard to the interaction protocol required by the environment.

Not solely the reuse of components is important, but also the adaptation of existing software for interaction with new systems is important for industrial projects. Especially the aforementioned web service technology is used regularly in this context.

To deal with those problems, a new discipline, *Software Adaptation*, is emerging. Software Adaptation focuses on the problems related to properly reusing existing software entities when constructing a new application by solving possible mismatches in their interaction. It is concerned with how the functional and non-functional properties of an existing software entity (class, object, component, etc.) can be adapted to be used in a software system and, in turn, how to predict properties of the composed system by only assuming a limited knowledge of the single components computational behaviour. The need for adaptation of software entities can appear at any stage of the software life-cycle, and adaptation techniques for all the stages must be provided. These techniques must be non-intrusive and based on formal executable specification languages such as Behavioural IDLs. Such languages and techniques should allow automatic and dynamic adaptation, that is, the adaptation of a component just at the time when it joins the system, and should be supported by automatic and transparent procedures. For that purpose, Software Adaptation promotes the use of software adaptors —specific computational entities for solving these problems. The main goal of software adaptors is to guarantee that software components will interact in the right way not only at the signature level but also at the protocol, Quality of Service and message-semantics levels.

The WCAT'06 workshop

Participants were asked to submit a short position paper prior to the workshop, addressing their points of view on the current problems, solutions and open issues in the fields of Software Coordination and Adaptation. Ten position papers were accepted for presentation in the workshop, which attracted a total amount of seventeen participants. During the workshop each participant presenting a paper gave a five-minute speech of her/his position, followed by a short discussion session, which served to identify a list of open issues in the fields under discussion. Then, several working groups were established, each on focussed on some relevant issue identified

during the discussion sessions. Finally, a plenary session was held, in which group conclusions were presented an discussed. A report on the WCAT'06 workshop has been published in the volume 4379 of the Springer LNCS series, pages 72–86.

After the workshop, participants were invited to submit extended and more technical versions of their work for this special issue. These submissions were reviewed by an international Program Committee set up for this purpose, including the six workshop organizers plus ten other relevant researchers related to the field. In the end, six extended papers were accepted for being published. We are very satisfied of the technical quality of the papers included in this special issue, and we are confident that we all together —authors of the papers, members of the program committee, and workshop organizers— have contributed to make an interesting special issue, representative of the current research efforts in the field of Software Coordination and Adaptation.

Program Committee

Steffen Becker
Carlos Canal
Nicolay Diakov
Juan Manuel Murillo
Pascal Poizat
Massimo Tivoli

Workshop Organizers