Knowledge-Based Architectural Adaptation Management for Self-Adaptive Systems

John C. Georgas[1]
Institute for Software Research
University of California, Irvine
Irvine, CA 92697-3425 USA
+1 949 824 5160
jgeorgas@ics.uci.edu

ABSTRACT

Self-adaptive systems continually evaluate and modify their own behavior to meet changing demands. An important element in the construction of architecture-based self-adaptive software is the specification of adaptation policy: this extended abstract presents an overview of work towards basing such specification on architecture-centric knowledge-based policies. This approach leverages techniques from the artificial intelligence field to explicitly represent adaptation policy at the architectural level, providing for strong decoupling between policy specification and architectural compositions, and supports dynamic runtime policy evolution promoting reuse potential and runtime flexibility.

Categories and Subject Descriptors

D.2.11 [Software Engineering]: Software Architectures – *languages*

General Terms

Design, Languages, Management

Keywords

Architectural adaptation management, self-adaptive software

1. OVERVIEW

Self-adaptive software is aimed at addressing the challenges of constructing systems that autonomously respond to a variety of situations. *Architecture-based* self-adaptive software focuses on using software architecture models as the central abstraction for decision-making and change enactment; the focus on explicit architectural models addresses the challenges and complications imposed by the multiple artifacts and levels of abstraction through which autonomous behavior can be expressed. A variety of research efforts have adopted this architecture-centric approach including contributions based on dynamic software architecture description languages (ADLs) as well as work based on dynamic distributed systems managed using explicit architectural models.

A core element in architecture-based self-adaptive systems is the specification of *adaptation policy*: the mapping between a set of conditions indicating the need for an adaptation and a collection of modifications addressing this need, along with a description of properties these modifications result in. Adaptation policies encapsulate the timing and specifics of change, addressing when and what needs to be done in order for a system to successfully adapt. These policy specifications, however, are often too closely coupled with the system to which they relate and remain static during runtime; in a domain expressly geared towards creating agile and adaptable systems, these specific shortcomings result in the specification of adaptive behavior that is conceptually hard to reuse across multiple systems and difficult to dynamically modify.

Our main research goal is to address these difficulties by developing mechanisms and supporting infrastructure for the specification of adaptation policies that are strongly decoupled from systems and dynamically modifiable at any point in the system's lifecycle, especially runtime. To achieve this, we propose an *architecture-centric* and *knowledge-based* approach in which adaptive behavior is achieved through a body of *observations* establishing known information about a system and a collection of *rule-based adaptation policy* specifications. Explicitly represented and dynamically and independently managed at the architectural level, these artifacts provide for a clear separation of concerns between a system's elementary components and the specification of the policies which define self-adaptive behavior as well as enabling systems which can not only autonomously adapt, but also dynamically change the manner in which they do so.

The key elements of this approach are the treatment of both adaptation policy and related knowledge as first-class architectural elements explicitly represented as decoupled parts of a software architecture, the dynamic management of policies at runtime, and the novel integration of well-established artificial intelligence methods in dynamic software architecture research. Of particular importance in the application of these techniques are issues of expressiveness, complexity, scalability, and overhead for which we plan evaluation activities using both small-scale prototypes as well as large-scale adaptive telemetry processing ground systems. For more information on this work, please see [1].

2. REFERENCES

[1]Georgas, J.C. *Knowledge-Based Architectural Adaptation Management*. <http://www.isr.uci.edu/projects/kbaam/>, Institute for Software Research, University of California, Irvine.

[1] This work was supported in part by NSF Grant CCF-0430066.

ICSE'05, May 15-21, 2005, St. Louis, Missouri, USA.
ACM 1-58113-963-2/05/0005.