DYNAMICO: A Reference Model for Governing Control Objectives and Context Relevance in Self-Adaptive Software Systems

Norha M. Villegas[1,4], Gabriel Tamura[2,3,4], Hausi A. Müller[1], Laurence Duchien[2], and Rubby Casallas[3]

[1] University of Victoria, Victoria, Canada
{nvillega,hausi}@cs.uvic.ca
[2] INRIA - LIFL - University of Lille 1, Lille, France
{gabriel.tamura,laurence.duchien}@inria.fr
[3] University of Los Andes, Bogotá, Colombia
rcasalla@uniandes.edu.co
[4] Icesi University, Cali, Colombia

Abstract. Despite the valuable contributions on self-adaptation, most implemented approaches assume adaptation goals and monitoring infrastructures as non-mutable, thus constraining their applicability to systems whose context awareness is restricted to static monitors. Therefore, separation of concerns, dynamic monitoring, and runtime requirements variability are critical for satisfying system goals under highly changing environments. In this chapter we present DYNAMICO, a reference model for engineering adaptive software that helps guaranteeing the coherence of (i) adaptation mechanisms with respect to changes in adaptation goals; and (ii) monitoring mechanisms with respect to changes in both adaptation goals and adaptation mechanisms. DYNAMICO improves the engineering of self-adaptive systems by addressing (i) the management of adaptation properties and goals as control objectives; (ii) the separation of concerns among feedback loops required to address control objectives over time; and (iii) the management of dynamic context as an independent control function to preserve context-awareness in the adaptation mechanism.

1 Introduction

The necessity of a change of perspective in the engineering of software systems has been widely discussed during the last decade by several researchers and practitioners in different software application domains [1,2,3]. In particular, Truex *et al.* posited that software engineering has been based in part on an incorrect set of goals, from the assumption that software systems should support rigid and stable business structures and requirements, have low maintenance, and fully fulfill these requirements from the initial system delivery [4]. In contrast to this static and "stable" vision, they proposed a new set of goals based on permanent analysis, dynamic requirements negotiation and incomplete requirements

R. de Lemos et al. (Eds.): Self-Adaptive Systems, LNCS 7475, pp. 265–293, 2013.

specification. Their proposal is aligned with the vision of self-adaptive systems, where dynamic adaptation is necessary to ensure the continuous satisfaction of their functional requirements while preserving the agreed conditions on Quality of Service (QoS) levels. These QoS levels are usually represented in the form of Service Level Agreements (SLAs), and their enforcement mechanisms are based on contracts and policies, among others [5,6]. To achieve the continuous satisfaction of changing requirements, the development of this kind of systems requires adaptation mechanisms able to perform short-term adaptations on them, and manage their long-term evolution [7]. As part of this adaptation and evolution, system analysis must be performed at runtime, and its requirements satisfaction must be monitored and regulated by continuously adjusting or enhancing its behavior [8,3].

Although the feedback loop model of *control theory* has been used as a reference in many self-adaptive systems in different application domains, the visibility of the feedback loop as the crucial architectural element to govern software adaptation remains often hidden. In many cases, the managed application is intertwined with the adaptation mechanism, rendering it as hard to analyze, reuse, and manipulate [9,8,10]. In other cases, such as those following the multi-layer architectures (e.g., ACRA [11], FORMS [12] and Kramer and Magee's [13]), their designs assume a completely closed and controlled context where monitoring requirements are not subject to change, even though several feedback loops can be evidenced in them. However, for many systems it is not affordable to discard unexpected context changes and dynamic changes in adaptation goals and user requirements, such as SLA re-negotiation at runtime. In these cases, statically deployed context monitoring elements are not enough to cope with these levels of dynamics, which are implied by context unpredictability.

Hence, as context information requirements evolve over time, due not only to changes in the execution environment, but also to the evolution of the adaptive system and its requirements, monitoring infrastructures are also required to be self-adaptive. Furthermore, in these cases the adaptation of the monitoring infrastructure implies to update the context analyzer of the target system's adaptation mechanism. Therefore, these changes must be coordinated by an independent feedback loop, that is, the one that manages changing control objectives and adaptation goals at runtime, thus preserving context-awareness in the system evolution.

In this chapter we present DYNAMICO (Dynamic Adaptive, Monitoring and Control Objectives model), a reference model for engineering context-based self-adaptive software composed of three types of feedback loops. Each of these feedback loops manages each of the three levels of dynamics that we characterize for self-adaptation: (i) the control objectives feedback loop, (ii) the target system adaptation feedback loop, and (iii) the dynamic monitoring feedback loop. As a reference model (i.e., a standard decomposition of a known kind of problems into distinguishable parts, with functionalities and control/data flow that are well defined [14]), DYNAMICO calls self-adaptive system designers to be aware whether the objectives, the system, or the monitoring infrastructure must be