UML-Based Design and Validation of Intelligent Agents-Based Reconfigurable Embedded Control Systems

Amen Ben Hadj Ali, Tunis El Manar University, Tunisia
Mohamed Khalgui, Xidian University, China
Samir Ben Ahmed, Tunis El Manar University, Tunisia

ABSTRACT
The paper examines UML-based design and validation of reconfigurable embedded control systems which can have multiple software architectural configurations such that each one is designed by a set of inter-connected software components. To handle dynamic reconfiguration scenarios, the authors define a software agent which interacts with users and applies several forms of reconfiguration at different granularity levels of the system’s architecture. The agent has the ability of monitoring the system’s environment and to apply appropriate and valid reconfiguration scenarios under well-defined constraints. Three architectural levels are defined in order to consider all possible reconfiguration forms of embedded systems. The authors define a set of UML-compliant metamodels to describe the knowledge about the reconfiguration agent, the system architecture, the reconfiguration scenarios, and the reconfiguration events. Validity of reconfigurations scenarios are checked using an UML-based environment which allows evaluating architectural and reconfiguration constraints. The proposed reconfiguration approach is applied to the FESTO production system.

Keywords: Agent-Based Architecture, Control Component, Design, Embedded Control Systems (ECS), Reconfiguration, Software Architecture, Unified Modelling Language (UML), Validation

INTRODUCTION
Embedded Control Systems (ECS) are special-purpose computer systems designed to perform one or a few dedicated functions, often with real-time computing constraints in order to control a physical process in the real world. Such systems are present in almost all modern life sectors such as automotive, avionics and industrial automation. In this context, one of the most important challenges is the tradeoff between performance and rapid response to market changes and customer needs. This tradeoff is better obtained when addressed early in the development process at design time. Indeed, modern embedded control systems incorporate
increasing amounts of software and even small changes in the system design or any failure at run-time require a cost- and time-intensive effort to adapt the system. One of the most promising directions to address these issues is the dynamic reconfiguration. This functionality refers to the process of modifying a system’s structure and behavior during its execution. Being reconfigurable is important for reacting fast to sudden and unpredictable requirement changes with minimum cost and risk.

In the last few years, researches have been conducted on architectures and software engineering in order to enable development of reconfigurable ECS. More specifically, to counter the effect of growing complexity, ECS are often designed in a component-based fashion using different technologies that have been proposed for this aim such as IEC61499 (IEC, 2005). From this vision, the RECS can be implemented by different software architectural configurations such that each one is designed by a set of inter-connected software components. Thus, the reconfiguration of ECS corresponds to the execution of reconfiguration scenarios on the software architecture of the system. A reconfiguration scenario is an ordered sequence of reconfiguration operations. Each operation is a transition from one configuration to another which is triggered under particular conditions as response to reconfiguration requests. A request represents a need to improve the system’s performance, or also to recover and prevent hardware/software errors, or also to adapt the system’s behavior to new requirements according to the environment’s evolution.

In addition, within the literature of RECS, two reconfiguration policies could be identified depending on the way of reconfiguration execution: static (offline) reconfiguration and dynamic (online) reconfiguration. In the last case, two sub-classes exist: manual reconfiguration executed by users and automatic (intelligent) reconfigurations assured by intelligent agents.

In our research work we are interested in the design of reconfigurable agents-based ECS in a platform independent way and with flexible reconfiguration spectrum covering manual, automatic and hybrid execution. For this purpose, we propose an UML-based approach for the design and validation of such systems. The Unified Modelling Language (OMG-UML, 2010) is the general-purpose standard for modeling intensive-software systems. The proposed approach is the first to our knowledge to deal with UML-based design and validation of reconfigurable agents-based ECS which is an attempt to answer three research questions: (i) how to model reconfiguration and system’s architecture, (ii) how to execute reconfiguration on the considered systems and (ii) how to ensure that reconfiguration agent brings the system into correct and safe behaviors. In order to answer these questions, there are two ingredients of the proposal, the specification and the validation of the solution.

The specification of the solution is covered by a set of UML-compliant metamodels enabling to specify the RECS architectures with their constraints, the intended reconfiguration scenarios and the architecture of the reconfiguration agent. The specification of RECS is based on the central concept of controller components described as UML components (OMG-UML, 2010) which are self-contained units, with ports as interfaces for external communications. Therefore, the concept of controller component used in this paper corresponds to a platform independent software unit that can be instantiated in several specific platform languages (such as Java or C++). The reconfiguration agent has the ability to monitor the system’s evolution and to apply appropriate reconfiguration scenarios under well-defined constraints at different granularity levels of the system’s architecture. In order to cope with all possible reconfiguration scenarios of UML-based ECS, we define three levels of reconfigurations: architectural reconfigurations allow add/remove controller components, structural reconfigurations updates composition of components and their internal implementation and data reconfigurations consist to data updating. Furthermore, we focus on typical requirements of real-time systems (such as timeliness and predictability) which were identified in Strasser, Muller, Sunder, Hummer,
COADA: Leveraging Dynamic Coalition Peer-to-Peer Network for Adaptive Content Download of Cellular Users


www.igi-global.com/chapter/coada-leveraging-dynamic-coalition-peer/68954?camid=4v1a