Assembly of components based on interface automata and UML component model

Samir Chouali *, Sebti Mouelhi *, Hassan Mountassir *

*Laboratoire d'Informatique de l'université de Franche-Comté, LIFC
{schouali, smouelhi, hmountassir }@lifc.univ-fcomte.fr

Abstract. We propose an approach which combines component UML model and interface automata in order to assemble components and to verify their interoperability. We specify component based system architecture with component UML model, and component interfaces with interface automata. Interface automata is a common Input Output (I/O) automata-based formalism intended to specify the signature and the protocol level of component interfaces. We improve interface automata approach by component UML model, in order to consider system architecture, in component composition and interoperability verification methods. Therefore, we handle in interface automata, the connection between components, and the hierarchical connections between composite components and their subcomponents.

1 Introduction

Component based systems are made up of collection of interacting entities, called components. The idea in component based software engineering (CBSE) is to develop software applications not from scratch but by assembling various library components, Szyperski (1999); Heineman et Councill (2001). This development approach allows, to extend component based systems via plug and play components, and to reuse components. Therefore one saves on development costs and time.

A component is a unit of composition with contractually specified interfaces and explicit dependencies, Szyperski (1999). An interface describes the services offered and required by a component without disclosing the component implementation. It is the only access to the information of a component. Interfaces may describe component infomation at signature (method names and their types), behaviour or protocol (scheduling of method calls), semantic (method semantics), and quality of services levels. The success of applying the component based approach depends on the interoperability (we say also component compatibility) of the connected components. The interoperability can be defined as the ability of two or more entities to communicate and cooperate despite differences in their implementation language, the execution environment, or the model abstraction, Konstantas (1995); Wegner (1996). The interoperability holds between components when their interfaces are compatible.

In this paper, we focus on assembling components described by interface automata. The interface automata based approach was proposed by L.Alfaro and T.Henzinger, Alfaro et Henzinger (2001, 2005); Alfaro et al. (2002). They have proposed to specify component interfaces with automata, which are labelled by input, output, and internal actions. These automata allow to describe component information at signature and protocol levels. An interesting verification approach was also proposed to detect incompatibilities at signature and protocol levels between two component interfaces. The verification is based on the composition of interfaces, which is achieved by synchronizing component actions.

The essential drawback of the interface automata approach is that, it is unable to accept as an input a set of interface automata, more than two, associated to all components composing a component based sys-