Automated Context Formalization for Context-aware Specification Approach

Amel Benabbou, Litio Laboratory, University Oran1 Ahmed Ben Bella, Oran, Algeria
Safia Nait-Bahloul, Litio Laboratory, University Oran1 Ahmed Ben Bella, Oran, Algeria

ABSTRACT

Requirement specification is a key element in model-checking verification. The context-aware approach is an effective technique for automating the specification of requirement considering specific environmental conditions. In most of existing approaches, there is no support of this crucial task and are mainly based on the considerable efforts and expertise of engineers. A domain-specific language, called CDL, has been proposed to facilitate the specification of requirement by formalizing contexts. However, the feedback has shown that manually writing CDL is hard, error prone and difficult to grasp on complex systems. In this article, the authors propose an approach to automatically generate CDL models using (IODs) elaborated through transformation chains from textual use cases. They offer an intermediate formalism between informal use cases scenarios and CDL models allowing to engineers to manipulate with familiar artifacts. Thanks to such high-level formalism, the gap between informal and formal requirements is reduced; consequently, the requirement specification is facilitated.

KEYWORDS

context-aware, Model-based verification, Requirement specification, UML interaction overview diagram, Use case

INTRODUCTION

In Model-checking, the System Under Study (SUS) is abstracted as a whole model, presented in the form of concurrent automata. The objective is to verify the correctness of requirements described in a formal properties language. Furthermore, in context-aware model-checking technique; the SUS model is separated from the set of behaviours of its environment (the set of actors). The context-aware technique focuses on modelling the system environment as a set of particular use cases scenarios, called Contexts. This technique has been introduced to solve the problem of explosion of the state space (Pelanek, 2009) by decreasing the set of possible behaviours. Thus, the SUS is closed with a well-defined finite and acyclic environment. Based on this idea, an approach has been proposed in (Dhaussy et al., 2012) within a verification methodology. In such approach, the SUS is modelled using Fiacre language (Berthomieu et al., 2008), which enables the specification of the interacting behaviours and constraints through automata. The surrounding environment in form of contexts is specified using the CDL (Context Description Language) (Dhaussy & Roger, 2011). Properties are formalized (also within CDL) to be checked on the SUS model using the model-checker OBP (Observer Based Prover) (http://www.obpcdl.org). See Figure 1.

CDL is a domain-specific language based on the concept of contexts and used for the formal specification of the SUS environment. The CDL structure is inspired from the Use Case Chart proposal (Whittle, 2006) and is hierarchically constructed in three levels: Level-1 is a set of use cases...
diagrams which describes hierarchical activity diagrams where either alternative (alternative/merge) or parallelization (fork/join) between several executions is available. Level-2 is a set of scenario diagrams organized by alternatives. Each scenario is fully described at Level-3 by UML2 sequence diagrams. Level-1 and Level-2 are based on the semantic of the scenarios and expressed using seq (•), alt ([]) and par (||) operators (par only for Level-1). At Level-3, the semantic of a scenario is expressed by a set of traces (sequence of ordered events) as described in (Haugen et al., 2005) and according to the semantics of UML 2 sequence diagrams. See Figure 2.

Within a CDL specification, the behaviour of each actor is considered as a set of use cases scenarios. The users are then required to identify the behaviour of each actor to formalize it in a CDL scenario. This manual process is a difficult and error prone task for engineers without formal methods background, furthermore, it requires: a) significant effort, especially when the system is strongly coupled with its environment; b) good knowledge of the syntax and semantics of CDL. Moreover, there is a semantic gap between the textual description of the use cases and CDL models that technically capture the messages exchanged between the system and its environment. Also, CDL has been evaluated through several industrial case studies (Dhaussy et al., 2009). However, the feedback reports that although it has solved several state explosion cases, it is perceived as a low-level language and difficult to grasp on complex models. Then, one need to express environmental scenarios at a higher level of abstraction that maps better to engineers. This aims to facilitate contexts elaboration and build intermediate models between textual format of use cases and CDL specifications. From these high-level artefacts, CDL models are later generated automatically.

Our work fits within this problematic. The authors are concerned by the elaboration of contexts from use cases to describe the functional requirements. In our approach, the use cases are of textual format constrained using a number of writing rules. These use cases are transformed into activity diagrams that are as well afterwards transformed into Interaction Overview Diagrams (IODs) (OMG, 2011). Within IODs, the boundaries between the system and its environment are limited to determine the interface with each actor; hence gates are used to relate IODs with the system boundaries. A global IOD is used in second time to organize all interactions according to the CDL structure given above (Figure 2). The set of interaction diagrams together with the SUS model (producing such model is

Figure 1. Context-aware verification approach: OBP Observation engine overview
Remote E-Voting Using the Smart Card Web Server
www.igi-global.com/article/remote-e-voting-using-the-smart-card-web-server/109580?camid=4v1a

Individual Improvisation in Information Systems Development
www.igi-global.com/chapter/individual-improvisation-information-systems-development/21064?camid=4v1a