Chapter 3

EXTREME: EXecuTable Requirements Engineering, Management, and Evolution

Ella Roubtsova
Open University of The Netherlands, The Netherlands

ABSTRACT

Requirements engineering is a process of constantly changing worlds of intentions, goals, and system models. Conventional semantics for goal specifications is synchronous. Semantics of conventional system modeling techniques is asynchronous. This semantic mismatch complicates requirements engineering. In this chapter, we propose a new method EXTREME that exploits similarities in semantics of goal specification and executable protocol models. In contrast with other executable modelling techniques, the semantics of protocol modelling is based on a data extended form of synchronous CSP-parallel composition. This synchronous composition provides advantages for relating goals and system models, reasoning on models, requirements management, and evolution.

INTRODUCTION

It is “reasonably well known that requirements will never be totally complete, finished, and finalized as long as a system is in service and must evolve to meet the changing needs of its customers and users” (Firesmith, 2005). However, there is a temporary notion of adequate completeness at some moment in time when the stakeholders are agreed on requirements. Adequate completeness of requirements is needed to estimate the development costs and to avoid incorrect assumptions for implementation decisions.

One of the powerful instruments to get adequately complete requirements is executable system modeling. Psychology studies show that people’s thinking is context related (Tversky & Simonson, 1999). For requirements engineering this means that stakeholders can identify missing or tacit requirements at the moment they see the behaviour of the system model. Hence, the executable models offer to stakeholders the contextual basis for identification of incompleteness. The semantics of executable modelling should be consistent with the semantics of goals.

In practice there is a semantic mismatch. The semantics of goals is synchronous. The conventional executable system modeling techniques are asynchronous. Asynchronous execution of the
models gives birth to states that are not expressed by the goals. In such states, stakeholders do not understand the execution of the models and cannot properly evaluate the models and reason on them.

In this chapter we propose a new method EXTREME that exploits similarities in semantics of goal specification and executable protocol models in order to simplify executable requirements engineering, management and evolution. Protocol models use a data extended form of synchronous CSP-parallel composition. The combination of protocol models and goal-oriented approaches semantically coherent, all states can be goal interpreted and this eases reasoning on models in terms of goals, goal refinement and identification of missing requirements.

Before showing the EXTREME method, we first remind elements of goal modelling. Then we remind elements of protocol modeling and show how to create protocol models corresponding to goals. The process is illustrated with a case from the insurance domain. We discuss the semantic elements of Protocol modelling that make it suitable for combination with goal-oriented modeling.

GOAL MODELLING

Goal-Oriented Requirements Engineering (GORE) is a well-established group of approaches (Kavakli, 2002; van Lamsweerde, 2004; Darimont & Lemoine, 2006; Regev & Wegmann, 2012). The aim of a goal-oriented approach is to justify requirements by linking them to higher-level goals.

The notion of a goal is used as a partial description of a system state being a result of an execution of the system. The authors of the GORE methods emphasize the similarity between goals, requirements, and concerns and propose to combine them in one tree structure. Goals are refined by requirements and concerns. The goal models are used to keep the business motivation in mind of requirement engineers and to elaborate the strategic goals with requirements and concerns.

An example of a goal tree is shown in Figure 1. The top nodes of Figure 1 present business goals of a simplified system supporting insurance business. The goals are:

- A product is composed.
- A policy is bought by a registered customer.
- A claim of a client with a bought policy is handled.

Each parent goal (the one pointed to by the arrow) is refined with a list of sub-goals and requirements. The leaves of the tree present system requirements. Business and strategic goals are expressed using concepts of the stakeholders’ vocabulary. Lower-level goals are typically expressed using words from the stakeholders’ vocabulary as well as specific technical terms introduced in the model on purpose and where necessary (Respect-IT, 2007).

Identifying goals is not proceeding exclusively from either a top-down or a bottom-up approach. In most cases the two approaches are used at the same time. Refining goals in a goal model often follows a so-called “milestone approach” (although there are many other decomposition approaches). Milestone goals represent goals as intermediate states in a process aimed to achieve the top goals. For example, the goal “A product is composed” (Figure 1) is refined by goals “There is a list of medical procedures”, “Medical procedures are combined into groups”, “Each group corresponds to a NoLimit(Coverage) or MaxCoverage”.

GORE trees are also used to relate goals and structural elements of the: Entities, Agents, and Operations. Entities represent passive objects in contrast with Agents that represent active objects. Agents are either human beings or automated components that are responsible for achieving requirements. The goals of this level assigned to the humans are called expectations. Software agents are responsible for requirements. Agents, Entities and their Relations are captured in an