Refinement of Software Architectures by Recursive Model Transformations*

Ricardo J. Machado[1], João M. Fernandes[2],
Paula Monteiro[1], and Helena Rodrigues[1]

[1] Dept. of Information Systems
[2] Dept. of Informatics
University of Minho, Portugal

Abstract. The main aim of this paper is to present how to refine software logical architectures by application of a recursive model-based transformation approach called 4SRS (four step rule set). It is essentially based on the mapping of UML use case diagrams into UML object diagrams. The technique is based on a sequence of steps that are inscribed in a tabular representation that is used to derive the software architecture for a focused part of the global system.

1 Introduction

The most complex activity during development of software systems is probably the transformation of a requirement specification into an architectural design [1]. The other phases have also their challenges, but they are better understood and a variety of methods, languages and tools are available to support the software engineer.

The process of designing software architectures is, by far, less formalised and often is greatly an intuitive ad-hoc activity, poorly based on engineering principles. Since the architecture of a software system constrains the space solution, the design decisions taken during architectural design must be made with great care, since they typically have a large impact on the quality of the resulting system.

An architectural transformation approach, called 4SRS (four step rule set), is presented that employs successive transformations of the software architecture, in order to satisfy the elicited user requirements. It is essentially based on the mapping of UML use case diagrams into UML object diagrams. The iterative nature of the approach and the usage of graphical models are important issues to guarantee that the final architecture reflects the user requirements.

Fig. 1 illustrates the recursive application of the 4SRS technique. This paper addresses the problem of deriving the logic architecture of a given platform service (called service object diagram), from a functional refinement of the platform architectural model (called platform object diagram), by adopting a recursive version of the 4SRS technique. The first 4SRS execution supports the platform requirements analysis by generating one platform object diagram that corresponds to the logic architecture of the system (this first 4SRS execution is described in detail in [2]). The second

* This work has been supported by projects STACOS (FCT/POSI/CHS/48875/2002) and USE-ME.GOV (IST-2002-002294).

J. Münch and M. Vierimaa (Eds.): PROFES 2006, LNCS 4034, pp. 422 – 428, 2006.

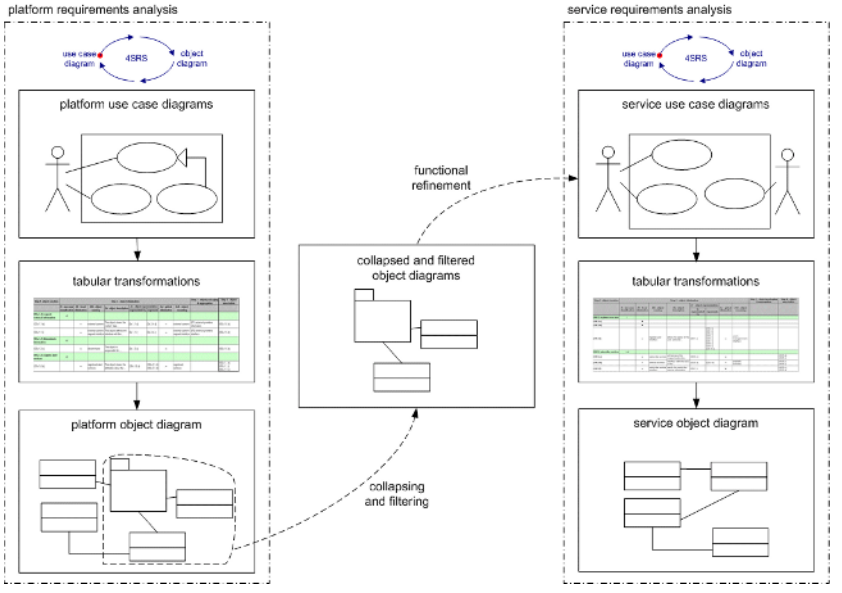

Fig. 1. Service specification with recursive 4SRS execution

4SRS execution supports the service requirements analysis by generating one service object diagram that corresponds to the logic architecture of the service to be specified (this second 4SRS execution is the aim of this paper).

The applicability of this technique is illustrated by presenting some results from a mobile application. For mobile applications, the definition of the underlying service oriented software architecture must consider as user requirements the services themselves, the mobile operators entry points and the final clients interfaces, and use them to characterize the platform. Within the presented demonstration case, the specification of one service of the mobile application was obtained by recursively applying the 4SRS technique.

2 Four Step Rule Set

4SRS is a technique proposed to transform users requirements into architectural models representing system requirements [3, 4]. It associates, to each object found during the analysis phase, a given category: interface, data, control. Each one of these categories is related to one of the three orthogonal dimensions, in which the analysis space can be divided (information, behaviour and presentation) [5].

For readability purposes, a brief description of the 4SRS technique is next presented. There is a complete description of its usage to obtain, in a non-recursive approach, the first logical architecture of the demonstration case used in this paper in [2]. In [6], an alternative version of the 4SRS technique is described for deriving the logical architecture for software product lines. This variant of the 4SRS technique