Software Architecture Recovery Process Based on Object-Oriented Source Code and Documentation

Sylvain Chardigny[1] and Abdelhak Seriai[2]

[1] MGPS
Port-Saint-Louis, France
s.chardigny@mgps.info
[2] LIRMM, university of Montpellier II/CNRS
Montpellier, France
seriai@lirmm.fr

Abstract. Architecture recovery aims at providing a high level abstraction of a system using the architectural elements to represent functionalities and interactions. This architecture makes easier the program comprehension and then provides many advantages during all the phases of software life cycle. Nevertheless, most architecture recovery approaches fail to combine the human expertise on the system with a high automation level. In order to solve this issue, we propose to use the intentional architecture of a system, which represent the system as imagined by its designers, to improve the adequation between the resulting software architecture and the architect's expectations without requiring more human expertise. Thus, we present in this paper a semi-automatic process to recover intentional architecture from the available documentation and the expert recommendations. This process is an extension of ROMANTIC, an approach aiming at recovering a component-based architecture of an existing object-oriented system.

1 Introduction

Given the explosive growth of the computer systems size and complexity, software architectures are emerging as a valuable ally for both the design and maintenance of these systems. This abstract view of systems has become, during the last decade, a central field of software engineering [1]. Its main advantages is to make easier the program comprehension by allowing us to focus on architectural elements (components, connectors and configuration) rather than implementation details [2]. In addition to program comprehension, this distinction between functionalities (components) and interactions (connectors) is crucial to safely maintain the system [3]. However most existing systems do not have a reliable architecture representation. Indeed these systems could have been designed without an architecture design phase, as it is the case for most legacy systems. In other systems, the available representation can diverge from the system implementation due to the lack of synchronization between software documentation and implementation.

Taking into account the previous considerations, we have proposed an approach called ROMANTIC[1] which focuses on recovering a component-based architecture from object-oriented systems [4]. Starting from the source code, our process aims at selecting among all the architectures which can be abstracted from a system, the best one according to our quality model. Then we formulate this model as measurable constraints and modelize the recovery process as a search-based problem aiming at balancing these competing constraints. This choice is motivated by the recent works on the search-based engineering showing that these techniques are very effective to solve this kind of problems [5].

The main advantage of our approach is its automation level which decreases the need of human expertise which is expensive and not always available. However, the code source is insufficient to recover an architecture which fulfills all the expert expectations and allows a complete comprehension of the system. Consequently, we propose to integrate in our process information about the intentional architecture, *i.e.* architecture as imagined by the designers, in order to identify an architecture reflecting all the design decisions.

The remainder of this paper is structured as follows. Section 2 presents an overview of ROMANTIC whereas Section 3 studies the place of the intentional architecture in the related work. In section 4 we describe the impact of the intentional architecture on the search-based process and the intentional architecture recovery process. Conclusion and future works are presented in Section 5.

2 Principles of ROMANTIC

ROMANTIC aims at recovering a component-based software architecture from an object-oriented system using a search-based approach [6]: an exploration process of the search space, *i.e.* a representation of all possible architectures, in order to identify the best solution according to a given fitness function.

Definition of the search-space. ROMANTIC operates on a search-space consisting of all the instances of a mapping model between object concepts (*i.e.* classes, interfaces, packages, etc.) and architectural ones (*i.e.* components, connectors, interfaces, etc.). According to this model, an architecture is a partition of the system classes. Each element of this partition, named “shape”, is mapped to a component and contains classes which can belong to different object-oriented packages. All existing links between shapes are mapped to connectors. Finally, the architecture configuration is mapped to the set of shapes constituting a partition of the system classes.

Definition of the fitness function. Our fitness function is based on a quality model for software architecture [7,4]. This model is based on the ISO-9126 norm [8] and refines successively architecture quality characteristics to sub-characteristics, properties on components and then on the shapes, like coupling or cohesion. The

[1] ROMANTIC: Re-engineering of Object-oriented systeMs by Architecture extractioN and migraTIon to Component based ones.