From Software Architecture Descriptions to Object-Oriented Designs

Guillermo Rodríguez Álvaro Soria Marcelo Campo

ISISTAN Research Institute (CONICET-UNICEN) Campus Universitario, Paraje Arroyo Seco, B7001BBO, Tandil, Bs. As.

{guillermo.rodriguez, alvaro.soria, marcelo.campo}@isistan.unicen.edu.ar

Service-Oriented Architecture (SOA) is becoming a dominant approach for developing distributed enterprise-wide applications. One of the reasons for SOA's extended use is its capacity to rapidly build applications by assembling already-implemented and Internet-accessible services, which allows software organizations to hasten development of distributed applications and their consequent time-to-market. However, this approach to develop SOA applications results unsuitable for particular organizations with high-priority and critical demands of internal control, security, flexibility, confidentiality and data integrity of their services, since the development of core functionality may be jeopardized by either uncertainty or changing environment. Although many efforts have focused mainly on facilitating discovering of services [2], and the outsourcing and reuse of them in SOA-based applications, little attention has been paid to aiding designers in developing services associated with business goals and quality-attribute properties. Moreover, quality-attribute properties of the service assemblies have been disregarded, which often leads to mismatches between the quality-attribute behavior prescribed by the SOA and the one resulting after its implementation.

In this context, we claim that it is necessary to conduct further research on developing services driven by quality-attribute properties within a software organization. Thus, the development of a service normally starts with some form of architectural description (e.g., public interface, main features to be provided, operating environment) and a set of quality-attribute properties, which are taken by the designer to produce a more concrete design model of the service. This refinement process is called SOA materialization [1]. For example, the implementation of a connection between services can be achieved by applying an Asynchronous Query pattern or a Bridge pattern, depending on the quality attributes defined by the SOA application. This decision on the suitable candidate design is an error-prone and time-consuming task. Therefore, designers need to be assisted in choosing from two or more SOA design alternatives that may be just as effective from a functional standpoint, but which might still diverge from the intended SOA architecture in terms of quality-attribute properties.

The main objective of our work is to assist designers in deriving object-oriented designs from SOA descriptions, preserving quality attributes. The hypothesis is that by leveraging on previous successful experiences and domain knowledge it is possible to materialize a SOA onto object-oriented designs that implement Web Services. The proposed approach makes use of Case-Based Reasoning (CBR) and allows designers to explore object-oriented design alternatives for a given SOA. We focus on architectural connectors as the main entities to codify a problem [4]. The designer's knowledge about SOA connectors and their materializations can be captured, whereas the process of retrieving past solutions and applying them to new situations can be systematized. The main contribution of our work is the development of a case base that puts knowledge about architectural connectors, quality-attribute properties and SOA design patterns to practical use. Furthermore, we enrich the case base with mappings that explicitly link SOA and object-oriented design patterns to quality-attribute goals. This quality-attribute perspective of SOA implementations by semi-automated generative tools is a relatively unexplored research field.

We have made considerable progress so far in the development of the approach. We built a core engine in an Eclipse environment and analyzed the number of alternatives generated by our approach, leveraging on the architectural descriptions. We have performed preliminary experiments of the techniques in several case-studies, obtaining promising results. To evaluate our CBR-based approach, we exercised the tool support and explored the candidate experiences to materialize the input SOAs. The candidate alternatives were compared with real alternatives available on the case-studies. Additionally, we checked the preservation of quality attributes in the different solutions by utilizing available design documentation of the case-studies. The metrics that we used to evaluate the retrieved solutions came from the Information Retrieval field. Nonetheless, we plan to test the techniques with more case-studies to further corroborate our findings. Some of the next steps in our research are: (i) to study the adaption phase of cases so as to new cases can be as similar as possible to the solutions expected by designers; (ii) to model the behavioral aspects of SOA designs to complement the object-oriented solutions with sequence diagrams and provide further assistance to developers when implementing SOA systems; and (iii) to analyze conformance between the architectural behavior and its corresponding source code once the SOA materializations are obtained. As for this issue, we developed a tool approach to reduce conformance checking efforts in the architecture-centric development [3].

References

1. Marcelo Campo, Andrés Díaz Pace, and Mario Zito. Developing object-oriented enterprise quality frameworks using proto-frameworks. *Software: Practice and Experience*, 32(8):837–843, 2002.
2. Marco Crasso, Alejandro Zunino, and Marcelo Campo. A survey of approaches to web service discovery in service-oriented architectures. *Journal of Database Management (JDM)*, 22(1):102–132, 2011.
3. JA Díaz-Pace, Álvaro Soria, Guillermo Rodríguez, and Marcelo R Campo. Assisting conformance checks between architectural scenarios and implementation. *Information and Software Technology*, 54(5):448–466, 2012.
4. Nenad Medvidovic and Richard N Taylor. Software architecture: foundations, theory, and practice. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2*, pages 471–472. ACM, 2010.