Architecture Restructuring to Support Evolution towards Service-Oriented Architectures

Niels Streekmann
OFFIS - Institute for Information Technology
Technology Cluster Enterprise Application Integration
niels.streekmann@offis.de

Abstract

The evolution of software systems and system landscapes towards service-oriented architectures is a widely discussed topic. Flexibility is a frequently named goal in this context. To reach this goal, complex information systems that are transfered to service-orientation have to be restructured. In practice the restructuring requires a high manual effort for understanding the existing system and the defintion of a mapping of its elements to a target architecture. Our model-based restructuring approach supports reengineers by semi-automating this task. In this paper we describe how this approach can be applied in the context of the evolution towards a service-oriented architecture.

1 Introduction

According to [2] the main source for flexibility in service-oriented architectures is not reuse, but the reduction of dependencies between services. This leads to a reduction of maintenance effort since the systems that are affected by changes are smaller and have clearly defined interfaces. To enable the reduction of dependencies in existing systems, it is required to restructure their architecture in the long run instead of only wrapping systems with web services. When an existing system provides the functionality of several services, wrapping makes this functionality more accessible, but it does not lead to a reduction of dependencies since the functionality implementation does not change. Systems that have grown over years exhibit strong dependencies in the implementation of distinct functionalities. In these cases, service implementations have to be divided from each other in order to make all advantages of service orientation available.

In order to restructure an existing system the first step is to gain an understanding of its current architecture. The system can then be incrementally transformed to a target architecture that consists of components with clearly defined interfaces that encapsulate the services' functionality. Tools like Sotograph and Bauhaus support the process of architecture reconstruction and understanding. The reflexion method [1] is one of the scientific methods used.

Our approach (as described in [4]) has the goal of reducing the necesssary effort to understand the current architecture and of directly supporting the restructuring process. Where the reflexion method and its current adoptions strongly rely on the hypothesized current architecture of the system and its package structure, our approach focusses on the target architecture of the system and leaves out current coarse grained structures like packages in its analysis. Hence it is assumed that our approach shows its advantages when the current architecture and the target architecture are very different. E.g., when the target architecture is service-oriented and the current architecture follows technical principles.

2 Architecture Restructuring

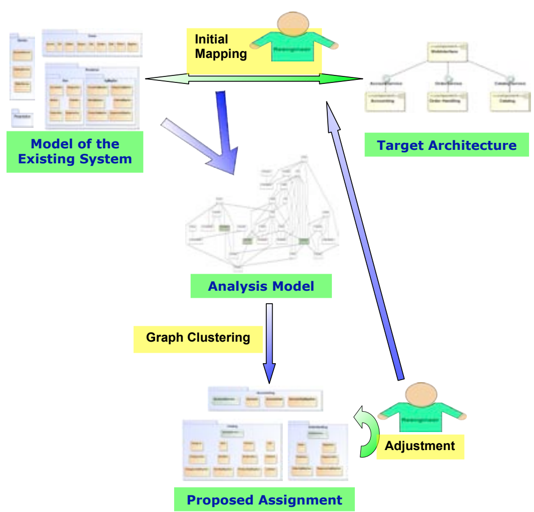

Figure 1: Overview of the restructuring approach

Figure 1 shows an overview of our approach. Details can be found in [4]. The goal of the approach is the definition of a complete mapping of elements of the existing system to components of the target architecture as a basis for the actual reengineering process. A model of the existing system and a model of the target architecture are preconditions to apply the approach. The target architecture model is assumed to be a UML model containing components and their interfaces with operations. In the case of a service-oriented target architecture interfaces represent services and components applications that implement these services. Finally, the information objects managed by the components should be contained in the target architecture.

Since the majority of reverse engineering tools use proprietary data models to serialise information about analysed systems, we chose to use specialised wrappers that transform this information directly to our analysis metamodel. The data needed to apply the approach comprises information about structural elements of the existing system, e.g. classes, interfaces and methods in object-oriented systems, and their dependencies. More fine grained structures like loops or other building blocks of methods are not considered. More coarse grained structures like packages are also not considered since they often represent the implemented architecture which is to be restructured.

Between the two models an initial mapping has to be created which maps information objects or operations of service interfaces to elements of the existing system. This partial mapping is then completed automatically using a graph clustering algorithm. The clusters found by the algorithm comprise all source elements that belong to one component of the target architecture. Using this information a UML diagram can be created that contains the components of the target architecture and defines a package for each component that contains classes, interfaces and operations that are represented by the cluster nodes. Thus, the code that has to be extracted from a system in order to make-up components or single services (as described in [3]) or that has to be cut-up in order to restructure the complete system can be defined.

3 Support for SOA Evolution

In the following section we give details on the application of our approach in the evolution towards service-oriented architectures. Possible scenarios in this context are: the restructuring of complete systems following its provided services or the extraction of single components, that provide services, from a complex application. We will describe the application of our approach to the second scenario.

The extraction of single services is a common task in a stepwise evolution towards service-oriented architectures. It exhibits different requirements to the target architecture. The target architecture should define the services to be extracted and their managed information objects. It also has to define the services of the remaining system that shall be used directly by the extracted services. The internals of the remaining system do not have to be modelled. This results in a target architecture that consists only of two components in the minimal case. One component provides the services to be extracted, and one component represents the rest of the system and provides the interfaces used by the extracted services.

In the context of a stepwise evolution, the target architecture may also contain temporary interfaces between the services to be extracted and the rest of the system, to be removed in later evolution phases. This case may result from the need for functionality that is only used by parts of the existing system which will be removed in the future and that is not part of the future service.

The proposed assignment maps all source elements that are needed for the implementation of the extracted functionality to the matching target architecture components. All remaining source elements are mapped to the component that represents the rest of the system. After this mapping, the services' implementations can be extracted from the implementation of the existing system in a later reengineering phase. The proposed mapping also shows all dependencies between the extracted components and the rest of the existing system. Dependencies that do not map to the interfaces described in the target architecture will have to be removed in the implementation.

4 Conclusions

We described the meaning of architecture restructuring in the context of the evolution towards service-oriented architectures. We also sketched our approach to architecture restructuring and its application in this context. Future work will be the evaluation of the approach and a comparison with related approaches.

References

[1] Andreas Christl, Rainer Koschke, and Margaret-Anne Storey. Automated Clustering to Support the Reflexion Method. *Journal Information and Software Technology*, 49(3):255–274, March 2007.

[2] Nicolai Josuttis. *SOA in der Praxis.* dpunkt.verlag, 2008.

[3] Harry M. Sneed. Migration in eine Service-orientierte Architektur. *Softwaretechnik-Trends*, 27(2):15–18, May 2007.

[4] Niels Streekmann and Wilhelm Hasselbring. Model-Based Architecture Restructuring Using Graph Clustering. In *Workshop Proceedings of the 13th European Conference on Software Maintenance and Reengineering*, pages 75–82, 2009.