SODA: A Tool Support for the Detection of SOA Antipatterns

Mathieu Nayrolles[1,2], Francis Palma[2,3],
Naouel Moha[2], and Yann-Gaël Guéhéneuc[3]

[1] École Supérieur en Informatique Appliquée, France
mathieu.nayrolles@viacesi.fr
[2] Département d'Informatique, Université du Québec à Montréal, Canada
moha.naouel@uqam.ca
[3] Ptidej Team, DGIGL, École Polytechnique de Montréal, Canada
{francis.palma, yann-gael.gueheneuc}@polymtl.ca

Abstract. During their evolution, Service Based Systems (SBSs) need to fit new user requirements and execution contexts. The resulting changes from the evolution of SBSs may degrade their design and Quality of Service (QoS), and thus may cause the appearance of common poor solutions, called *Antipatterns*. Like other complex systems, antipatterns in SBSs may hinder the future maintenance and evolution. Therefore, the automatic detection of such antipatterns is an important task for assessing the design and QoS of SBSs, to facilitate their maintenance and evolution. However, despite of their importance, no tool support exists for the detection of antipatterns in SBSs. In this paper, we introduce a prototype tool, called SODA, for detecting SOA (Service Oriented Architecture) antipatterns in SBSs. SODA also supports specification of SOA antipatterns.

Keywords: Antipatterns, Service Based Systems, Detection, Specification.

1 Introduction

Service Based Systems (SBSs) evolve to fit new user requirements, e.g., additional functionalities or better Quality of Service (QoS). These technical and functional changes may degrade the design and QoS of SBSs and often introduce poor solutions, called *Antipatterns*, by opposition to *patterns* which are good solutions to recurring problems. *Multi Service* and *Tiny Service* are two common and recurring antipatterns in SBSs, and it is revealed, in particular, that *Tiny Service* is the root cause of many SOA failures [4]. *Multi Service* is an SOA antipatterns that corresponds to a service that implements a multitude of methods related to different business and technical abstractions. Such a service is not easily reusable because of the low cohesion of its methods and is often unavailable to end-users [1]. Conversely, *Tiny Service* is a small service with just a few methods, which only implements part of an abstraction. Such service often requires several coupled services to be used together, resulting in higher development complexity and reduced usability [1]. While degrading the design and QoS of SBSs, antipatterns may make it harder for engineers to perform maintenance and evolution tasks. SOA antipatterns are more dynamic in nature, thus more

challenging to detect. Therefore, the automatic detection of such SOA antipatterns is an important activity to assess the design and QoS of SBSs, and thus ease the maintenance and evolution tasks of the engineers. However, a number of works have been devoted for the development of detection tools within OO systems [2 ; 5 ; 6]. Yet, for the detection of SOA antipatterns in SBSs, there is no tool support.

Thus, in this paper, we present a SOA antipattern detection tool, SODA (Service Oriented Detection for Antipatterns) to help engineers, for detecting SOA antipatterns automatically in SBSs. This tool provides the means for both static and dynamic analysis of SBSs.

The remainder of this paper is organized as follows. Section 2 surveys related work on tool support for the detection of OO code and design issues. Section 3 presents our detection tool, SODA, along with the underlying approach and some results. Finally, section 4 concludes and sketches future work.

2 Related Work

With the goal of detecting OO code and design related issues, a number of tools have been introduced in the literature [2 ; 5 ; 6]. Nevertheless, researchers and developers have rarely considered tools to perform detection for SOA antipatterns, i.e., in SBSs. Král *et al.* [3] specified briefly seven SOA antipatterns, but did not discuss their detection. To this end, we try to fill the gap, by proposing a tool, called SODA, as the tool support for detecting SOA antipatterns.

3 Overview of SODA Approach

We developed the tool SODA being inspired from our approach of the same name, SODA, proposed in [7]. Figure 1 represents the three main steps of SODA: (1) Specifying SOA antipatterns in the form of rule cards from their textual descriptions, (2) Generating detection algorithms conformed to the antipattern specifications, and (3) Detecting automatically SOA antipatterns and involved suspicious service(s) in the analyzed SBS.

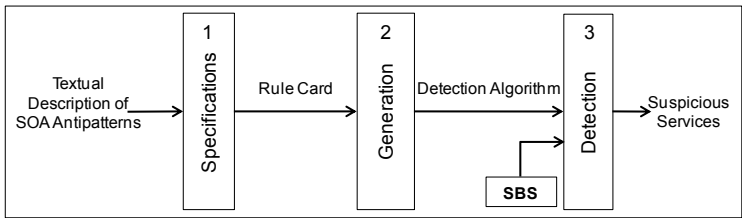

Fig. 1. SODA Approach for the Detection of SOA Antipattern

In [7], we perform a domain analysis to specify SOA antipatterns by studying their definitions and specifications from the literature to pinpoint significant static and dynamic properties (represented as *metrics*). We then use these properties as the basis for the vocabulary to define our own domain specific language (DSL), and formalize rule cards. A rule card is the specification of a certain SOA antipattern at a high-level of abstraction using a combination of multiple

singleton rules. Starting from the specifications of SOA antipatterns described with rule cards, we generate detection algorithms automatically from rule cards, by applying a simple *template*-based technique. We also develop a framework, called SOFA (Service Oriented Framework for Antipatterns) [7], that supports metric-based detection of SOA antipatterns in SBSs. SOFA assists the tool SODA, and provides all services needed for the detection of SOA antipatterns, such as, static and dynamic analyses, essentially in the form of *metrics*.

3.1 Description of SODA Tool

Figure 2 presents the snap-shot of our SODA tool. We mark different sections of the tool from 1 to 7. Section 1 enlists the SOA antipatterns that can be detected. For the selected antipattern, Section 2 provides textual description, while Section 3 shows the corresponding rule card; Section 4 presents the results, i.e., suspicious service(s); Section 5 provides values for all metrics (from the associated rule card), for each service; Section 6 exposes the generated association rules. Finally, Section 7 helps to visualize the suspicious service(s) within the analyzed SBS.

Most of the dynamic and static metrics calculated by SOFA use only the service interfaces that are freely available. An extension of our tool, called SODAAR (Service Oriented Detection for Antipatterns based on Association Rules) enables SODA to identify suspicious service(s) by mining association rules [8] to discover interesting relations between services, i.e., patterns, using execution traces. Association rules are implications of the form A $\rightarrow$ B (i.e., if-then statement), where A and B may be a single service or a subset of services. In SODAAR, each execution trace is considered as a *transaction* and invoked methods identified within traces as *items*. Based on these association rules, we can classify suspicious services. Considering the metric-based framework, i.e., SOFA and our extended SODAAR, we developed a complete tool, SODA.

Principal Features of SODA:

1. SODA does direct import of an SBS as a JAR package.
2. SODA has a straight forward detection interface for the users, which is handy both for beginners and experts.
3. SODA shows all the detection details, i.e., metric values, corresponding rule cards, antipattern descriptions etc.
4. For the detection, SODA supports both well-known metric based and execution trace based analysis of SBSs.
5. Also, SODA exposes all the execution traces, association rules generated from those traces, and relations among them, that is also useful to the users to better understand the SBS analyzed.

Figure 2 shows the detection results for *Multi Service* antipattern. An elaborative presentation about the SODA tool, more detection results and further materials are available at http://sofa.uqam.ca/tool.html.

4 Conclusion and Future Work

The detection of SOA antipattern is important for assessing the design and QoS of SBSs, to ease the maintenance and evolution of SBSs. In this paper, we

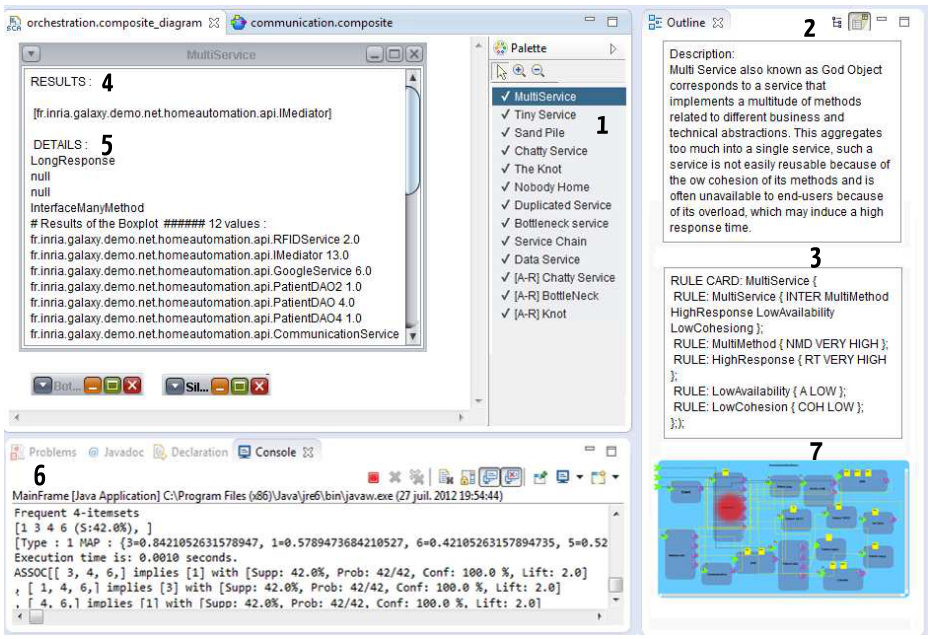

Fig. 2. Detection of SOA Antipatterns with SODA

presented a tool, SODA, for the detection of SOA antipatterns. SODA incorporates the framework, SOFA, i.e., metrics and rule-cards based analysis. SODAAR, an extension of SODA is also introduced that is based on execution trace analysis. As the future work, we intend to develop a version of SODA as an Eclipse plug-in and provide a graphical interface to visualize the detected antipatterns easily by engineers.

References

1. Dudney, B., Asbury, S., Krozak, J., Wittkopf, K.: J2EE AntiPatterns. John Wiley & Sons Inc (2003)
2. Fokaefs, M., Tsantalis, N., Chatzigeorgiou, A.: JDeodorant: Identification and Removal of Feature Envy Bad Smells. In: Software Maintenance, 2007. ICSM 2007. IEEE International Conference on. pp. 519–520 (October 2007)
3. Král, J., Žemlička, M.: Crucial Service-Oriented Antipatterns. vol. 2, pp. 160–171. International Academy, Research and Industry Association (IARIA) (2008)
4. Kral, J., Zemlicka, M.: Popular SOA Antipatterns. Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, Computation World 0, 271–276 (2009)
5. Marinescu, R.: Detection Strategies: Metrics-based Rules for Detecting Design Flaws. In: In Proc. IEEE International Conference on Software Maintenance (2004)
6. Moha, N., Gueheneuc, Y.G., Duchien, L., Le Meur, A.F.: DECOR: A Method for the Specification and Detection of Code and Design Smells. IEEE Trans. Softw. Eng. 36(1), 20–36 (Jan 2010), `http://dx.doi.org/10.1109/TSE.2009.50`
7. Moha, N., Palma, F., Nayrolles, M., Conseil, B.J., Guéhéneuc, Y.G., Baudry, B., Jézéquel, J.M.: Specification and Detection of SOA Antipatterns. International Conference on Service Oriented Computing (To appear in ICSOC, 2012)
8. Oracle: Data Mining Concepts 11g Release 1 (11.1) Part Number B28129-04, `docs.oracle.com`