WSDarwin: A Framework for the Support of Web Service Evolution

Marios Fokaefs
Department of Computing Science
University of Alberta
Edmonton, AB, Canada
Email: fokaefs@ualberta.ca

I. THE WSDARWIN FRAMEWORK

Service-oriented architectures have emerged as a popular development paradigm for modular, extendible distributed systems, relying on interoperable components. However, their evolution is a challenging problem, due to the fact that their constituent parts typically reside outside a single entity’s domain of ownership and control. In my thesis, I have developed WSDarwin, a framework to support the evolution of service systems, which contributes three distinct tools.

First, the framework offers an Eclipse plug-in to automate the adaptation of service clients in Java towards changed web services. The tool comprehensively compares standard service interfaces (WSDL for SOAP services and WADL for REST services) and identifies simple and complex changes between two versions of the service. The changes are then mapped to the auto-generated client proxy, which is changed accordingly to be adapted to the new version of the web service. Finally, the tool automatically invokes the client’s test cases through the Eclipse interface to confirm that neither the service changes nor the adaptation process have affected the proper function of the client application.

Second, WSDarwin offers a web application that specifically targets clients of REST services. The application invokes the WSDarwin functionality as a web service and presents the results to the user through a lightweight and interactive Javascript interface. The tool can automatically generate the WADL interface for a service by exercising this service using a given set of URL requests and analysing the responses. Additionally, the tool compares different versions of a WADL interface, similarly to the Eclipse plug-in, identifying both simple and complex differences. Finally, it maps the elements from different services that belong to the same general domain to support the interoperability of service clients and allow them to easily migrate between services from different providers.

The final offering of the WSDarwin framework is a system to support the decisions concerning the evolution of a web service within a greater ecosystem of competitive providers and clients. The basic premise of the system is that a service ecosystem is a competitive environment, whose participants have conflicting interests but a common goal; technological progress and satisfaction of functional requirements. Such an environment is best described as a game, so we use Game Theory to model the complex relationships between providers and clients and help them make the optimal decision concerning the evolution of the system. A mathematical model is also proposed to express the economic parameters of this decision. These parameters are constrained by the technical aspects of the system and are subjected to externalities, which are the indirect effects of one party’s decision to the rest of the ecosystem.

The WSDarwin framework is currently at the stage where it has already been built and is now being tested and evaluated in terms of its usefulness, its accuracy, its efficiency in facilitating the evolution process and its effectiveness in addressing the associated problems.

The Eclipse plug-in will be evaluated in terms of its usability and the simplicity of its interface. A large scale experiment with actual developers has been planned. The developers will be asked to complete a set of tasks concerning the evolution of a service system and the adaptation of a client application. For all tasks, usability data like mouse clicks, key impressions and focus changes will be gathered to assess the simplicity, usefulness and level of integration of the WSDarwin interface. The tasks will be completed once using WSDarwin and once more using off-the-shelf tools. This will allow us to assess whether WSDarwin provides a more efficient and integrated solution over manual effort and more traditionally used tools by measuring the time it took the developers to complete the tasks using either set of tools.

The accuracy of the web application tools will be evaluated in terms of precision and recall. First, an “oracle” will be created that will include manually created WADL interfaces for popular REST services including GitHub, Facebook and Twitter among others, manually compared versions of these services and manually compared services from different providers including IMDb, RottenTomatoes, GoogleMaps and BingMaps. Then, the performance of WSDarwin will be compared against the oracle to determine the degree to which the tool can automatically perform the desired tasks.

The evaluation of the decision-support system is not trivial since it requires the consideration of data, which can be considered sensitive and it is usually kept private by software companies including costs and effort. For this reason, I am considering a synthetic but realistic ecosystem based on real-world cloud services of Amazon, Google and Microsoft. For such big companies, there is an amplitude of data publicly available. The rest of the required data can be randomly generated within realistic ranges without affecting the robustness or the validity of the results.