AppSolut: A Framework for Composed App Solutions

D. Bano, F. Kluthe, B. Mohrmann, A. Nikaj, S. Vijapurwala

University of Paderborn
Contact: [dbano|fkluthe|basti86|anikaj|shafi]@mail.uni-paderborn.de

Introduction

The project group AppSolut was a one year project as part of the Master in Computer Science curriculum at University of Paderborn. The main goal of the project was to design and develop a framework for composing Android apps (i.e. apps from the Android App Store) into a single app. The motivation of developing such a framework came from the idea of utilize the huge market of Android apps for creating a composition of apps which serves a specific set of desired tasks. This set of tasks (including nonfunctional properties) can be stated by the user, who is not necessarily an expert, using the framework's simple and user-friendly interface. Once the user states his/her request, the framework invokes the automatic matching. This matching is responsible for finding the apps that satisfy the user's request best. Once the matched apps are found, the framework generates a composed app out of them. The challenge of creating compositions out of existing services is well investigated in the area of web services. In our project we try to transfer the existing concepts used in the web service world to the Android world. Here we consider Android apps as web services. In that regard the two main challenges of this project were: 1) the creation of a formal language for describing the apps' content and behavior 2) realizing how apps can be composed.

The idea of enabling non-expert and novice users to create own Apps composed out of existing Apps was also researched by the MIT App Inventor[http://appinventor.mit.edu] and the MAIDL Designer[Chaisatien et al. *Mobile mashup generator system for cooperative applications of different mobile devices*]. In both of these approaches, it is necessary that the user *knows* the Android apps he wants to use in its composition beforehand. This means he has to be aware of the Apps range of functions and their API. By contrast, in our approach we present a way of finding appropriate Android apps by simply describing needed functionalities and quality requirements.

Technologies

During the project we utilized a variety of different technologies but focus on using the Java programming language as far as possible. Thought the whole project we follow a model driven approach. The editor itself was developed as an Eclipse Plug-in. Here the tooling was built up with the help of the EMF and GMF. A database, which holds information necessary for matching, was set up with the help of Teneo. For doing matching, we created our own ontologies which comprised the knowledge of verbs and nouns. We used this knowledge to perform matching which consists of signatures, protocols, pre and post conditions and non-functional properties matching. We used SPARQL for querying the ontologies from our tool and Jena as semantic reasoner for SPARQL queries.

The output of our tool is an automatically generated Android App. This was realized through dynamically generating large parts of the Android code. To achieve that, the Xtend programming language was used. Another technology that we employed was Xtext, which served us to create a DSL with a textual syntax. This textual syntax enables the user of our tool to express complex

requests faster than building them up with graphical means. All the technologies mentioned above were tightly integrated into the Eclipse IDE.

Software Development Process

During this project group, our development process was organized in different phases which took place in a sequential manner. The main phases of our project were the following: seminar, mini-project, requirements engineering, prototype, design, implementation and evaluation. In the seminar phase, we investigated the area of technologies. In the mini-project phase, we learned to work with the new technologies. We followed an iterative software development approach going several times through design, implementation and evaluation phase. During each phase, the team members were redistributed not only in order to give them the possibility to work on different topics and to gain knowledge in different areas, but also to know each other better and to share knowledge with one another.

Lessons learned

While working on the project, we got experience in recent technologies, especially the Android platform and code generation frameworks. Likewise, we also got our hands-on in the field of scientific research, particularly in domain specific languages and matching approaches. In addition to the technical know-how, we also learned how to collaborate as a team. Our team consisted of eight individuals, each with their own origin (five different nationalities), cultural and educational background. This diversity was one of our challenges that turned out to have positive effects on each student and on our combined work. Another challenge was self-organization. Our supervisors gave us the chance and freedom to make design decisions, allocate the tasks, and decide for technologies mainly by ourselves. As a matter of course, we had to follow the basic conditions and argue for our decisions clearly. This definitely facilitated our sense of responsibility and the ability to assert our ideas with the help of arguments. However there were also deficits in our work, particularly in testing. Here we could have improved with a comprehensive testing plan, more man power and/or a person in charge.

Basic information

The work of our eight-headed group lasted eleven months, from October 2012 to September 2013. We were able to finish all stages of the development from requirement engineering to implementation and evaluation. In the end, we came up with a working tool suite. However, in this process we also collected ideas about future work, for example using the tool suite for a more complex and comprehensive scenario. Therefore you could imagine a Logistic Manager, which consists of different apps for route planning, GPS, accounting and maps.

Further information and a presentation about this project can be found on our group's website: http://pgappsolut.cs.uni-paderborn.de/

Presentation

The presentation will start with a brief overview about the AppSolut tool suite, followed by a 10 minutes video demo show provided with explanatory comments. During the video show, we want to provide tablets with which the audience can use the composed app.