An HTML5-based Service Composition Approach for End-User Development

Feifei Hang, Supervised by Dr. Liping Zhao
Software Systems Group
hangf@cs.man.ac.uk

1 Introduction

Web 2.0 is a trend that aims at transitioning the World Wide Web to the realm of human-human interactions, where users are both producers and consumers. It currently consists of an increasing set of ad hoc applications that provides users with a platform to author, share, and integrate knowledge and experiences [1]. As a very popular form for interaction with computers, *End-User Development* (EUD), such as Mashup, has became emerging as a driving technology for creating ad hoc applications. According to [2], millions of users daily create their own solutions using scripting or high-level visual programming language. Meanwhile, as a fast growing technology, HTML5 has shown its great advantage especially in the aspects of human-computer interaction.

In recent years, a number of EUD platforms have been produced by various indutry and acadamic players. Well-known products include Yahoo! Pipes [3], IBM Mashup Center [4], and Intel Mashmaker [5]. These platforms have different features but they support users with a similar set of functionalities aiming to relieve users from the tedious engineering work in creating EUD applications. The core functionalities typically include the following:

- Repositories for collecting web resources that can be used to create ad hoc applications. These resources, know as feeds, are usually provided in the form of web services, AJAX APIs or RSS.
- Tools for users to combine feeds into ad hoc applications. These tools attempt to simplify the programming work by using graphic models and templates. However, programming cannot be entirely avoided.
- Infrastructures for testing, hosting, and publishing ad hoc applications. Some platforms even allow users to rate or comment on the published applications.

These features are fundamental and important for EUD and especially for mashups. However, they are not suitable for novice users to effectively create ad hoc applications which best satisfy their requirements.

1.1 Problem Statement

EUD, especially mashup, aims to glue together existing web services and data into completely new applications in a rapid and inexpensive way. With recent development in software architecture and tools, even enterprises have begun using EUD tools to develop new business applications on demand. However, despite their consistent and steadfast growth in recent years, there has not been a flood of EUD-based applications. According to [1], one of the main reasons may be related to the various obstacles faced by an average user in building new personalised applications using existing EUD technologies. Generally, these obstacles can be illustrated as follows:

- *Procedural approach* to building new applications. Existing EUD tools and technologies expect users to equip sufficient knowledge of not only the types of applications to build but also methods to create them. Ideally, the EUD platform should adopt an effective and KISS [6] approach to provide EUD tools and platforms to a wide range of users. In this approach, we envisage, users specify their needs, requirements and the running procedures of their applications in a graphical manner effectively. The underlying platform responds by parsing the involved service feeds, composing new applications, executing the application and returning the results to users.
- *Programming skills* requirements that are beyond the expertises of average web users. Existing platforms assume that users have sufficient knowledge of web applications such as basic knowledge of Web Services, AJAX, or workflows.
- Lack of a framework for *support and management* of EUD applications. Consequently, large-scale adaptation of EUD applications becomes challengeing. If average users unleash hundreds of new applications, these applications need to be organised in such a way that they can be searched, queried, composed, and executed as part of the EUD application. This is a genuine chanllenge and the ideal EUD platofrm must have some framework for support

and management of EUD applications.

1.2 Motivating Scenario

In this section, we introduce the motivating scenario to clarify our definitions of end users and end-user service composition (i.e. end-user development). We define end-user service composition as the service composition performed by users who are not professional developers, that is, users whose main job is outside of computer science, in areas such as humanities, engineering, medicine, or business. These users are usually very sophisticated in the field of their expertise, but they have little time or interest in using service composition tools as a means to accomplish their professional or private task. For a more elaborate view on end users, we recommend [7]. Here, we give a sample scenario that illustrates the type of end-user service composition that we are aiming at, with Audrey as our end-user developer.

Audrey is a travel writer, preparing a book about cities in England. To collect necessary information, she has to visit a group of people living in different places. However, due to the climate condition in England, she often has to check weather report before she starts her journey. Additionally, she also needs to find the exact place where her interviewees are living in. Instead of using maps and watching weather forecast on TV, she decides to compose a simple application that aggregates all the services she needs.

She opens her web browser, visits an HTML5 web page, connects to service feeds, including an online map service and a weather reporting service, and chooses a layout for her application from templates. She then composes a mashup application, which allows her to input the postal codes of her interviewees and then shows corresponding weather information and location on a notification area and a map respectively. Although she could use these services individually, the possibility to quickly compose an aggregate service in a EUD platform enables her to get a simple unified user interface, adapted to her needs, saving her time, so that she can focus on her main task. Yet, she is able to compose those service without having any knowledge of programming.

1.3 Our Approach

In this section, we propose our approach as well as its potential architecture. Shown by Figure 1, our approach (i.e. EUD platform) mainly serves the following two purpose:

1) *Being a service composition platform* for end-users. As described in Section 1.1 and Section 1.2, with helps of the built-in Services Parser, the underlying platform will be able to parse and process the corresponding services descriptions, e.g. WSDL documents, WADL documents [8], etc., as soon as particular services are being selected for composing. Moreover, an UI designing component will help end-users build the UIs of their new applications and bind the behaviours of composed services, e.g. showing results in a popup dialog or getting user inputs from a "textfield", with the UIs.
2) *Generating middleware for heterogeneous services*. As a EUD platform for composing services, middleware will be generated depending on the selected services for bridging and transferring data between the newly created end-user applications and services.

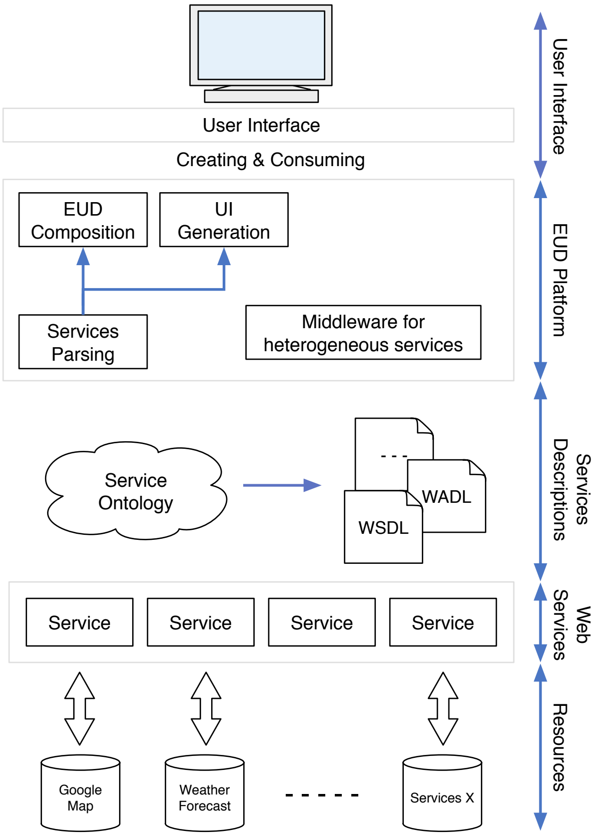


Fig. 1. EUD platform system architecture

1.4 Challenges and Questions

One of the biggest challenges in this research lies in the feeds that could be used to compose. To date, most available platforms such as Yahoo! Pipe are only capable for processing HTTP-based/RESTful web services [9]. SOAP-based web services usually have no ways to be composed in the newly created applications. Moreover, according to [10], web services are being required to expose the CRUD interfaces (i.e. `GET`, `POST`, `PUT` and `DELETE`) in a RESTful context. Also, as illustrated in Figure 2, using heterogeneous HTTP verbs is the premise of a RESTful web services ranking at level 2 in Richardson Maturity Model. However, as far as we know, none of the existing EUD platforms has the ability to build

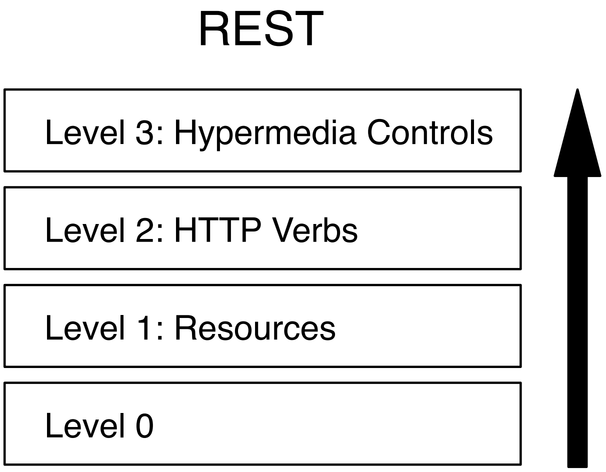


Fig. 2. Richardson Maturity Model

compositions of RESTful web services by using the HTTP verbs other than `GET`.

Therefore, to sum up, the challenges and questions will be addressed in this research so far at this stage are:

1) How can SOAP-based web services be composed through an EUD platform?
2) How can all the CRUD interfaces of RESTful web services be used through an EUD platform?
3) How can web services description documents such as WSDL be processed and used in an EUD platform?
4) How can HTML5 technologies be helpful for building an EUD platform?

1.5 Research Scope

From the previous overview, the reader can observe that research on an advanced HTML5-based EUD service composition approach should consider three interconnected aspects as captured by Figure 3.

1.6 Research Contributions

As a replacement of HTML4.01 and XHTML1.0, the history of HTML5 can be traced to 2004 which is the year that Web Application 1.0 (a.k.a. the predecessor of HTML5) was initially introduced. However, we did not witness an explosion of HTML5-based technologies until current years. During the entire period, little research has been addressed towards the combination of service composition and end-user development under in an HTML5 environment; therefore, we try to tackle this gap. In general, the main contributions of this research so far are:

- Design and implementation of an HTML5-based service composition platform for end-user development.
- Preliminary development of an HTML5 based theory for service composition and end-user development.

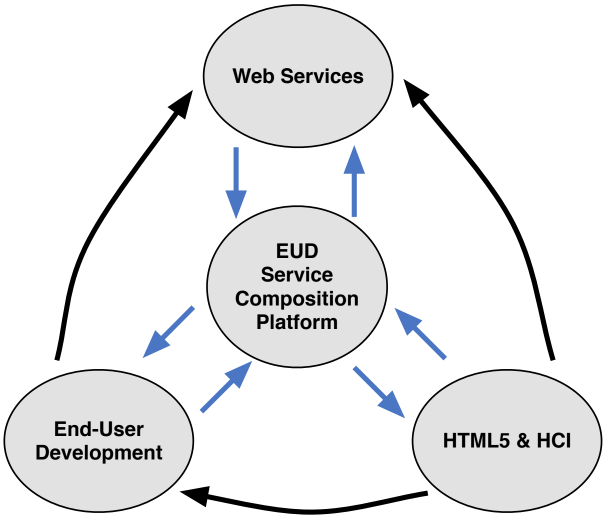


Fig. 3. Research Scope

2 Related Work

In [2], Obrenovic and Gasevic proposed an end-user service composition tool, called AMICO, based on spreadsheets. They managed to compose heterogeneous services, including SOAP services, RESTful services, etc., within a spreadsheet. However, in order to create compositions, end users have to learn and use predefined formulas such as `AMICO_WRITE(..., ...)` and `AMICO_READ_LOOP(..., ..., ...)`. As mentioned, end users are a group people who are not professional developers. Therefore, we believe that the formula-based composing approach is not a good solution in terms of providing an end-user development tool.

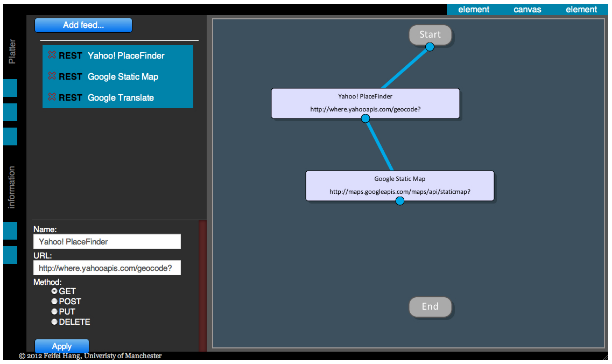
As mentioned in Section 1, Yahoo! Pipes provides users an online platform for creating their own applications. By using the graphical programming language, users can assemble service compositions by simple drag-n-drop without writing any code. However, as far as we know, Yahoo!! Pipes is not compatible for many types of services such as SOAP-based services and the RESTful services which use “`form`” to transfer data.

Therefore, the downsides of the above works motivated us to do this research project.

3 Works Undertaken So Far

At this point of time, a very first prototype of our proposed approach is under construction. Figure 4 shows the current design and look of our platform. Instead of forcing end users using built-in web services, we provide the ability of adding external web services into the platform. In order to store all the external web services, Web SQL Database [11] is being used to store all the essential data and informations.

In order to enable the ability of using CRUD interfaces when composing RESTful web services, we leverage the power of XMLHttpRequest which is a native object in all modern browsers. As can be seen from Figure 4, end



Name: Yahoo! PlaceFinder
URL: http://where.yahooapis.com/geocode?
Method:
● GET
○ POST
○ PUT
○ DELETE

Fig. 4. HTML5-based EUD Platform

users can choose from all the CRUD interfaces when merging RESTful web services into their compositions. However, due to the security reason, the HTTP verbs `PUT` and `DELETE` cannot be invoked directly in parts of modern web browsers. Therefore, a middleware, as mentioned in Section 1.3, will be created during our future research.

Also, by using one of the most important concepts of HTML5 - Canvas, we provide end users the ability of building their own applications by dragging and dropping in our platform.

4 Evaluation

One of the challenges in creating an EUD platform is the evaluation of the entire approach. To do this, we need to measure the effectiveness and usability in order to identify if the platform can be helpful to our target user group. In this research, a focus group, which potentially involves a group of non-expert end users (like the travel writer, Audrey, in Section 1.2), will be established. By measuring the time that they spend on building each of their own applications in our platform, we will be able to exam the effectiveness of our approach. Also, by validating all their applications, we can therefore examine the correctness of our platform. Hence, evaluation studies that involve human interactions will be part of our future work.

5 Conclusion

In this paper, we produced a detailed introduction of our research on creating an HTML5-based service composition approach for end-user development. Broadly speaking, we find the possibility of producing a such platform by using HTML5 techniques. Also, we aim at filling the existing gaps that illustrated in Section 2. However, in order to facilitate the development of the proposed approach, certain aspects of web services such as transactions and securities will not be taken into the account of this project.

References

[1] A. Bouguettaya, S. Nepal, W. Sherchan, X. Zhou, J. Wu, S. Chen, D. Liu, L. Li, H. Wang, and X. Liu, "End-to-end service support for mashups," *IEEE Transactions on Services Computing*, 2010.

[2] Z. Obrenovic and D. Gasevic, "End-user service computing: Spreadsheets as a service composition tool," *IEEE Transactions on Services Computing*, 2008.

[3] Yahoo!, "Yahoo! pipes." [Online]. Available: http://pipes.yahoo.com/pipes

[4] IBM, "Ibm mashup center." [Online]. Available: http://www.ibm.com/developerworks/lotus/products/mashups/

[5] R. Ennals, E. Brewer, M. Garofalakis, M. Shadle, and P. Gandhi, "Intel mash maker: join the web," *SIGMOD Rec.*, 2007.

[6] Wikipedia, "Kiss principle." [Online]. Available: http://en.wikipedia.org/wiki/KISS principle

[7] B. Nardi, *A Small Matter of Programming: Perspectives on End User Computing*. MIT Press, 1993.

[8] M. J. Hadley, "Web application description language (wadl)." [Online]. Available: http://wadl.dev.java.net/

[9] L. Richardson and S. Ruby, *RESTful Web Services*. O'Reilly, May 2007.

[10] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, 2000.

[11] W3C, "Web sql database." [Online]. Available: http://www.w3.org/TR/webdatabase/