# Conceptual and Usability Issues in the Composable Web of Software Services

Abdallah Namoun[1], Tobias Nestler[2], and Antonella De Angeli[1]

[1] Manchester Business School, Booth Street East, Manchester,
M13 9SS, United Kingdom
{abdallah.namoune,antonella.de-angeli}@mbs.ac.uk
[2] SAP Research Center Dresden,
Chemnitzer Str. 48, 01187 Dresden, Germany
tobias.nestler@sap.com

**Abstract.** Enabling the diffusion of lightweight service composition approaches among end users necessitates the appropriate understanding and establishment of the correct user requirements that lead to development of easy to use and effective software platforms. To this end, a user-centric study which includes 15 participants is carried out to unravel users' mental models about software services and service composition, their working practices, and identify users' expectations and problems of service composition. Several examples and prototypes are used to steer this elicitation study, among which is a simple composition tool designed to support non-programmers to create interactive service-based applications in a lightweight and visual manner. Although a high user acceptance emerged in regard to "developing service-based applications by end users", there is evidence of a conceptual issue concerning understanding the notion of service composition (i.e. end users do not think about nor do they understand connections between services). This paper discusses various conceptual and usability problems of service composition and proposes recommendations to resolve them.

**Keywords:** web services, light weight service composition, requirements, end user development, presentation layer, usability.

## 1 Introduction

Europe is continuously spending tens of millions of Euros to fund Service and Software Architectures research projects that aim at facilitating the development of interactive service-based systems through user-friendly software platforms. Sadly much of this research effort is dedicated to solving technical complexities and implementation problems rather than understanding what end users really need, how they think about service composition, and the natural working ways to support their activities and increase service creation and consumption. It is not uncommon that some of the funded projects decide on implementation strategies and software solutions for their target user group before even starting the project either because they want to pursue their own research interests or claim to understand the needs of their users without

F. Daniel and F.M. Facca (Eds.): ICWE 2010 Workshops, LNCS 6385, pp. 396–407, 2010.

carrying out user studies. This explains why, despite the rapid advancement in Service Oriented Architecture, user research in that area is still in its early infancy. User studies that focus on understanding user development abilities, needs, and mental models about service composition are evidently required to establish correct and well-informed requirements.

Web 2.0 enables Internet users to add content to the web, interact with and customize web pages, share information, and collaborate to accomplish tasks. In general, these activities do not necessitate acquisition of IT specialist knowledge since, for example, the act of modifying a wiki page is fairly simple for anyone who knows how to use a computer and browse the Internet. A promising approach to empower users beyond web content development is to use and reuse loosely-coupled software components such as web services [1] in order to create composite applications tailorable to their diverse needs. However, there is no substantial proof that naïve users (i.e. users with no Computer Science/IT background) can combine functionalities together to produce augmented software services. The question as to whether "users understand or think about uniting/connecting independent functionalities to form greater assemblies" is the key to the success of many service composition research projects.

Changing and customizing web content is different from composing software services since the latter is more complex and challenging. Creating computer programs usually requires strong modeling abilities and problem-solving skills which most ordinary web users do not acquire, a fact that has always intimidated end users. Transforming ordinary consumers of services into actual producers of services invites various unanswered research questions: what do users understand by web services? are they just black boxes, user interfaces they interact with, snippets of code, etc? are end users willing to uptake development activities at the cost of time and efforts? how can end users be helped to better understand service connections and encouraged to develop software applications? which representations and metaphors should be used to enable easy development of service-based applications? Such understanding is crucial to using the appropriate representation of services and service composition in authoring and modeling environments.

In this paper, we present a user study in which end users with no computing expertise viewed and commented on several examples of service composition and interacted with an early prototype of a visual composition tool. The current paper endeavors to:

1. Capture users' true understanding of web services and composition of service-based applications. We refer to this as "mental models" which explain how end users visualize web services and the inner workings of service composition. This finding will impact the way software services should be represented to end users in development environments.
2. Identify conceptual and usability problems that relate to service composition, as probed by realistic examples and prototypes of a visual composition tool.
3. Improve the design of service development environments through a set of design recommendations; thus enhancing the diffusion of services among Internet user.

## 2 Existing Work on Data Mashups and Service Composition

At present the web offers users the capability to build personal pages through customizable web portals, such as iGoogle[1] and MyYahoo![2], where they add web feeds and gadgets (i.e. programs that provide services) to their personalized pages. In principle, users browse a list of services, e.g. Weather, Google Map service, and add the desired ones to their pages. They can also edit these services and modify the look and feel of their pages by applying a desired theme or moving the gadgets within the page outline. Widget-based applications and customizable web portals are easy to use but do not support the creation of complex software applications because services can not be combined with each other, in other words users can not create or manage connections between web resources and services. It would be more useful if ordinary users are enabled to compose software components together in order to produce rich and complex service-based systems that fulfill their specific needs, but are also allowed to extend and customize these applications easily.

Although Web 2.0 facilitates the formation of web-based communities and consumption of services, such as: wikis, video-sharing, and social-networking sites, it does not allow the formation of powerful applications that consist of web services interacting together ([11], [14]). Service composition is well-understood and covered by existing approaches for technical developers using composition languages (e.g. BPML, BPEL4WS, WSCDL … etc) [13], however tools and methodologies for enabling end-user service composition have been largely ignored ([6], [12]). The current technical aspects of service composition are not of much interest to ordinary users who want to capitalize on the benefits offered by Service-Oriented Architecture. What really matters to end users is how these technologies are presented to them and how they can use them to perform their desired tasks in an easy manner and without the need to learn programming languages.

Promising approaches for a lightweight user-driven application design are Mashup platforms (overview provided by [7]), which appeared with the growth of Web 2.0. The graphical composition style constitutes a first step towards user empowerment and increasingly shifts the creation of individual applications to domain experts. Mashup platforms like Yahoo! Pipes[3] or Open Mashup[4] enable the creation of more sophisticated service-based applications by aggregating web feeds, pages and services from different sources. Unlike customizable web portals, users can define relationships between modules by dragging and linking them together within a specialized visual editor. The output of one module can serve as the input of another. However, mashups mainly focus on data aggregation and still lack concepts to create composite service-based applications by end users ([5], [12]). Furthermore, they require modeling skills and good understanding of computing concepts such as message and data passing which most web users do not have [15].

Efforts to make programming accessible to a wider range of audience exist in the literature, such visual programming [2] which uses visual notations and languages to

[1] http://www.igoogle.com
[2] http://www.my.yahoo.com
[3] http://pipes.yahoo.com
[4] http://www.open-mashups.org

facilitate programming, and programming by demonstration [4] which enables users to demonstrate the desired program by examples in a step by step manner. As regards service composition, Namoune et al conducted focus groups to discuss the risks and benefits of end user composition of service-based applications. End users were mainly concerned about the privacy and security of their personal data, as well as the underlying technical complexity they might encounter when composing service-based applications [9].

Despite the continuous progress in service-oriented technologies, service composition by end users (non-programmers) is an area in its early stages. Therefore, identifying the needs and specific requirements of ordinary users is a crucial prerequisite to the design of *"easy to use"* and "*easy to understand*" service composition environments. The challenge to service and Human-Computer Interaction (HCI) research lays in finding new methods to open up service composition to a larger population supplying non-technical users with an intuitive development environment that hides the complexity of services and service composition. This goal although desirable, opens other interesting challenges to HCI.

## 3 Experimental Set Up

The study at hand was conducted in the form of a contextual interview/inquiry with the aim of exploring end users' natural mental models and the potential problems of service composition. This knowledge will help us create and shape service development environments that facilitate end user composition. Beyer and Holtzblatt argue that contextual interviews are very useful for identifying user contextual needs and how specific actions are performed in detail [3]. Moreover, they enable researchers to understand users' environments and their work conduct; thus portraying actual user behavior. 15 non-technical students from the Manchester Business School participated in the study. Each individual interview took approximately one hour to complete.

### 3.1 Procedure

During the interview we set up a focus in regard to the objectives of the ServFace project[5] and how these objectives will be fulfilled. In this study, the focus is to enable ordinary users to build composite software applications that are tailored to their needs using a light-weight simple composition tool called *ServFace Builder*. To guide the interview several widespread examples (e.g. iGoogle, Google Map Search Service), low fidelity prototypes, and a high fidelity prototype of a future authoring tool were used. Each participant was instructed to perform the following tasks.

1. Define web services, widgets, and web applications, and explain how these software artifacts work from a user perspective. Following the participants' answers, the interviewer provided the exact definition of each term with examples
2. View a mock-up of the composition tool and make initial comments and impressions, the interviewer asked elaborating questions such as: what do you see? What do you think it does? How do you think it works?

[5] www.servface.eu

3. Walk through a simple service composition example "student course enrolment" in which the purpose and main features of the tool were explained by the interviewer (Figure 1). Through this selected example that suits participants' environment and background we aimed to effectively communicate the idea of "service composition by end users"
4. Indicate their views regarding service composition and evaluate the mock-ups of the tool
5. Go through a service composition scenario and build a composite service using an early prototype of the tool (Figure 2). Concrete task description: "You are a team assistant for a team of 50 people. You can use MS Office well and like to play around with the tools you use and customize them to fit your needs. Since your colleagues often need to attend conferences it is your responsibility to organize their trips. In the past you spent a lot of time searching and booking suitable flights and hotels. In the future, you want to build an application that allows your colleagues to book their trips without spending a lot of time on it. *Thus, your main task is to build a software tool that facilitates travel booking using the ServFace Builder*"
6. Indicate their final views regarding the composition tool and the general composition approach

### 3.2 Materials

#### 3.2.1 Motivating Examples

To simplify the definition of software services, widgets and web applications to our non-technical audience, we used examples that most people are familiar with, in particular: Google Map Search Service, Date and Time Gadget by Google, and Google Suit (email, Calendar, Documents, Web, Reader, etc). It is worth noting that these examples were only shown to participant after they had provided their definition and examples.

#### 3.2.2 Low-Fidelity Prototypes and Student Scenario

Participants were shown a set of mock-ups of our service composition tool using Microsoft PowerPoint (Figure 1). The mock-ups demonstrated how a student can create a composite application that allows her to register for a particular course of study. The interviewer went through the process of visual service composition and explained the necessary steps. Subsequently, participants were invited to make comments or ask questions.

#### 3.2.3 High-Fidelity Prototype

The evaluated ServFace Builder was developed in the frame of the EU-funded research project ServFace. The tool utilizes the advantages of web service annotations [8] enabling a rapid development of simple service-based interactive applications in a graphical manner. The tool applies the approach of service composition at the presentation layer, in which applications are built by composing web services based on their frontends, rather than application logic or data [10]. During the design process, each web service operation is visualized by a generated UI (called service frontend), and can be composed with other web service operations in a graphical manner. Thus, the

**Fig. 1.** Mockups of the Potential Simple Composition Tool –ServFace Builder

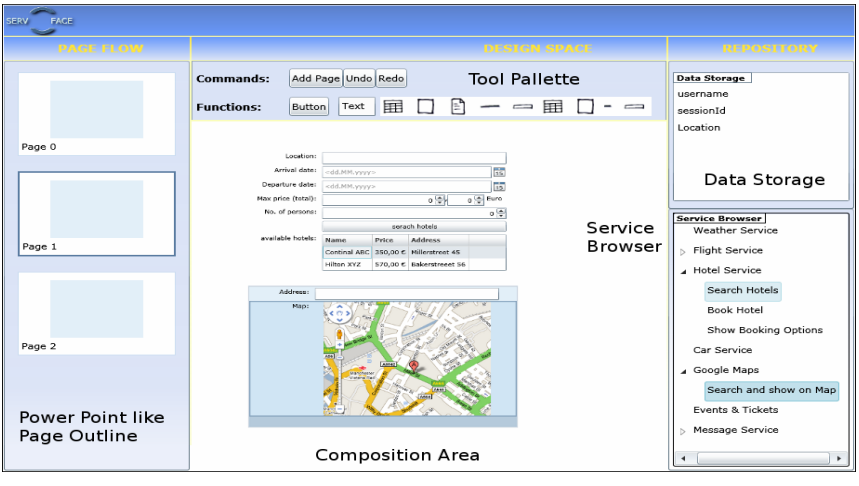


**Fig. 2.** Early Prototype of the Simple Composition Tool –ServFace Builder

user, in his role as a service composer and application designer, creates an application in WYSIWYG (What you see is what you get) style without writing any code. It is worth noting that the tool depicted in figure 2 has been improved and enriched with many features now based on users' feedback and design recommendations gained through this study. However, for the interest of this paper, figure 2 shows the version of the tool we used to steer the contextual interviews.

# 4 Results

## 4.1 Users' Mental Models of Software Services

Users provided diverse definitions for web services. 3 users defined services as services which are available on the web, while another 3 users defined services as the provision of information or knowledge. The remaining users referred to web services

as: a tool to build the web and applications, online communities, search engines, and interactive elements. 6 users argued that unlike traditional web pages which are static, services are interactive elements which enable them to perform tasks. When prompted to report examples of services 7 users mentioned search engines (Google, Yahoo) and E-commerce sites (Amazon). Others mentioned social-networking systems (such as: Facebook), website tools, and learning environments (the University portal).

Once the interviewer provided the exact definition of services to the participants and presented the Google Map as an elaborating example, all users were able to explain how it can be used. They had a very clear idea of how to interact with it, data can be entered by typing "search query or term" in the text field and clicking the "Search Maps" button. The service then returns the results back to the users in the form of an image (i.e. a map). Some users (3 users) also indicated that they can interact with the service by editing its options (e.g. show satellite imagery and show traffic). All users referred to the information they supplied as "*input*" and to the results returned by the service as "*output*", showing that this terminology is commonly shared amongst users with no technical background.

The concept of widgets appeared to be more difficult. Although some users acknowledged to have heard the term, no one was able to define or guess what a widget is. 5 users defined web applications as applications that run on the web, for example iGoogle, Google docs, and Hotmail. Once the "Google suite" was introduced to the participants, they all commented that it is useful to have one application with many services bundled together as this is more convenient, saves time, reduces workload, and prevents errors.

To sum up users were able to provide a very basic and general definition of services abstracted from technical details. However, they were able to describe how they would interact with web services and web applications. Users seem to perceive service-hosting sites as single services instead of a collection of web services as shown by the provided examples. Users considered the term "widgets" technical and were not able to define it. Users knew about Web 2.0 and had already used its technologies like blogs. Surprisingly many of them had already built their own websites which confirms that web users are becoming proactive about developing the web.

### 4.2 Users' Perception of Service Composition

All users liked "to develop their own software applications that suit their needs and interests" with the help of authoring tools. They argued that this will allow them to perform complex tasks more easily and rapidly, both in their personal and professional life. They also pointed out that assembling many services, for instance booking a flight, finding a hotel, booking and insuring a car service, within a single application is a powerful feature missing in today's applications which are usually designed for one purpose. In normal circumstances, users have to access different online resources and services to accomplish their goals, which is a cumbersome process. Users appreciated that the introduction of service composition will reduce the amount of work required to perform tasks (i.e. logging into one service instead of many services/ applications), reduce the chance of making mistakes, and it will be more convenient to group heterogeneous services together in one application. Furthermore, integrating services using their front-ends only without worrying about the integration of data and business logic reduces the complexity of application development.

### 4.3 End User Composition Problems

As previously indicated one of the main objectives of this evaluation is to identify conceptual problems that can be generalized to other mashup editors and service composition environments and propose measurements to resolve them. Motivated by the travel booking scenario, participants pinpointed several drawbacks with the light-weight composition. Table 1 lists and explains both conceptual and usability problems that our end users faced while composing services. Other usability problems that are very specific to the ServFace Builder only are not reported below. Each of the detected problems was rated on a three-point rating scale (low, moderate, high).

**Table.1.** Conceptual and Usability Problems of Service Composition by End Users

| Conceptual Problem | Severity | Usability Problem | Severity |
|---|---|---|---|
| **1-Awarreness of service composition/ connection:** despite introducing the concept of "building applications by users" in the walkthrough examples, participants had problems understanding the purpose of the composition tool (i.e. that the tool is designed to develop applications). Instead, they thought single services are software systems which operate independently. All users failed to mention that web services can be connected together. After we informed them about the possibility of combining services together and asked whether they want the tool to perform it on their behalf, all users preferred to be involved in the process of combining services because they do not fully trust the system and feared it might cause problems. | High | 1-Direct and visual manipulation of services: selecting and placing services into the main canvas was not intuitive to the participants and caused problems. Upon placing services into the design area, users had difficulty trying to move them around to create an organized visual layout. Moreover, users wanted to adjust the size of services layout but this was not supported by the tool at the time of the study. | Moderate |
| **2-Definition of execution flow of services and logic of application**: users were confused about specifying the execution order of the services they added to the design space. In other words, they had problems specifying which service the application should execute first, which one should come next, and so forth. | High | **2-System support**: users were not sure if they were doing the right actions and complained that the system did not inform them about the consequences of their activities, emphasizing that proactive help from the system is important. | High |
| **3-Understanding of technical terms**: users were intimidated by some technical jargon used in the tool and their meaning such as service operation and parameters, and they asked for explanation. Inability to understand elements and concepts used within a design tool may result in users quitting the tool. | High | | |

**Table 1.** (*Continued*)

| | | |
|---|---|---|
| **4-Security**: there was a security concern from users in relation to using web services that require supplying sensitive information such as: bank details. Users were worried that services retrieved by the tool could disclose their personal information to third party service providers or could be compromised by malicious intruders and hackers. | High | |
| **5-Distinction between design and runtime**: users had difficulty understanding the difference between design time and run time. Some users started to input data into entry fields of the services during the development phase and expected the application to process results accordingly. Clearly they had misconceptions about the two phases. | Moderate | |

## 5 Discussion and Recommendations for End User Composition

Although users were some times confused about the purpose of the tool, they showed a high likeability towards "composing service-based applications that are tailorable to their needs". This agrees with the current trends that end users are becoming proactive about developing the web [11]. Assembling various services within a single application was favored by the participants because it saves time and effort, is convenient, and offers multiple functionalities, agreeing with [9].

The most interesting result of this evaluation revealed that the tool was not self-reflective of its composition aspect as users did not attempt to create links between services in the task scenario. This may be attributed to the innovative idea of combining different software components together which end users are unfamiliar with. In contract to customizable web portals (e.g. iGoogle) and social networking sites (e.g. Facebook), where users usually search for and add external services to their pages without having to define relationships between services, this design tool and the alike require users to wire atomic services together.

Another challenge exposed by this tool is how can end users with minimum service composition experience specify the steps they are required to undertake in order to build an application. In particular, it is not an obvious task for users to recognize the services that contribute towards the accomplishment of a particular user goal.

Surprisingly adopting common practices in future design tools does not necessarily improve user experience and their usability. For example, although some users were able to link the page flow section on the right hand side of our service composition tool to the Microsoft Power Point slides section, they demonstrated poor ability to use it appropriately. It is important for designers to come up with and compare several design alternatives before committing to a particular design solution.

Other aspects of the tool that created confusion were related to the service computing terms, such as: service operation and parameters. This can be attributed to users' unfamiliarity and poor knowledge of technical terms.

In light of user understanding of software services and service composition and according to the identified service composition problems the following tentative design recommendations are suggested:

***Rec 1***- Service understanding and representation in service composition environments: users showed a poor understanding of the technical details of web services; thus, we encourage service designers to represent services via user interfaces to naïve users since visual representations can communicate and express the purpose and details of these services more effectively. Abstract representations of services (i.e. black box representations) are difficult to interpret and understand by non-programmers, whereas snippets of code are designed for serious programmers.

***Rec 2***- Service composition strategy: there are two fundamental issues to service composition, (1) connecting services and (2) identifying the order by which these services should be executed. To resolve these issues we propose to use a semi-automatic approach (i.e. system-guided composition) that seamlessly creates links between services while giving users the power to modify those links as they see most appropriate. Whenever a new service is selected from a list of services and added to the design space (e.g. service 3, Figure 3), the system should check for service compatibility issues and create the desired connection (Serv 3 and Serv 19, Figure 3). The system also highlights the possible links between a new added service and existing services. For the second issue we recommend to use a task modeling view by which users can specify the goal of their application and the tasks they need in order to accomplish their goal. Once the task analysis tree has been created users can associate services to particular actions (e.g. book hotel). The overall aim is to specify the services that contribute towards the accomplishment of a user goal without worrying about service connections. Furthermore, it would be very useful if a library containing several task analysis templates of possible assemblies is made available to users who can then reuse and extend them according to their specific needs.

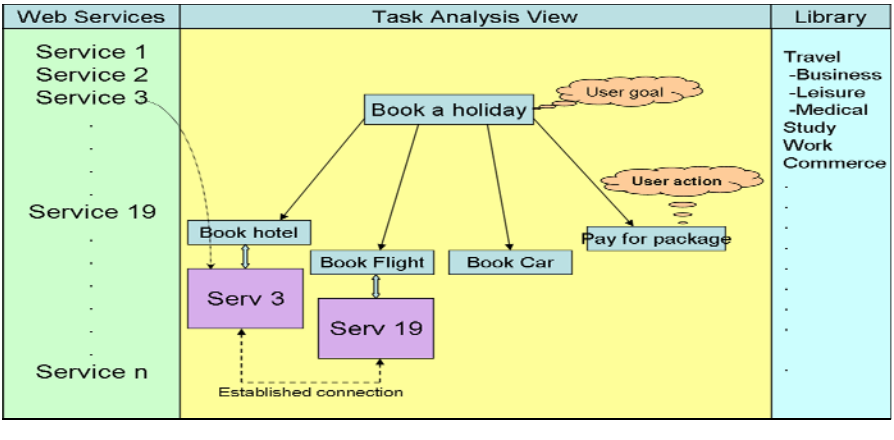


**Fig. 3.** A Potential Service Composition Design Solution

***Rec 3***- Service-related terms: technical jargon (i.e. service operations, parameter, Data types, widgets…) is not well understood by ordinary web users; therefore, we advise to use friendly and self-explanatory terminology to elevate technical complexity and enhance users' understanding of service composition aspects.

***Rec 4***- Visual Manipulation of services: service composition environments should provide a convenient and large design area. In addition, users should be enabled to easily interact with and visually manipulate services and their features (i.e. adding, moving and deleting services from the canvas, changing their dimensions, color…).

***Rec 5***- Secure services: the system has to deal with security and privacy aspects professionally. In that respect, the retrieved services must be trustworthy and reliable and this should be clearly communicated to the users through special symbols (e.g. a secure digital e-sign, verisign identity protection), as well as providing guarantees from service providers to compensate service consumers in case of frauds.

***Rec 6***- Continuous user feedback: users were sometimes inquisitive about the current state of their design; hence, we suggest adding proactive assistance (e.g. intelligent software agents) that continuously gives assurances and notifies users about the consequence of their actions, especially in critical situations.

Furthermore, we noticed that most of our users designed their application in one page instead of multiple pages. Therefore, we recommend enabling service composition within one design page as it is more convenient and less confusing. In regard to the design tool's name, users suggested using names that reflect their personality and give them a feeling of ownership and control over the tool such as: "myTool, myApplication, youDesign … etc".

## 6 Conclusion and Future Work

This paper discusses users' mental models of services and service composition, and the main issues end users with no modeling skills or programming expertise may face during the composition of services using service development environments. Users favored the idea of being able to build their own applications but results showed they were not thinking about linking services together to form augmented assemblies and had difficulty distinguishing between design time and runtime concepts. We therefore propose to support service composition environments with intelligent mechanisms that automatically define connections between services (e.g. control and data flow) "system-driven composition" while allowing users to control and customize these connections. In future work, we plan to improve the current service composition tool by addressing some of the identified and inferred problems to simplify the process of building service-based applications. Subsequently, a user study will be conducted to evaluate the usability of the latest version of the ServFace Builder.

**Acknowledgments.** The work presented herein is supported by the EU-funded project ServFace.

## References


1. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services: Concepts, Architectures, and Applications. Springer, Heidelberg (2004)
2. Burnett, M.: Visual Programming. In: Webster, J.G. (ed.) Encyclopedia of Electrical and Electronics Engineering. John Wiley & Sons Inc., Chichester (1999)
3. Beyer, H., Holtzblatt, K.: Contextual Design: Defining Customer-Centered Systems. Morgan Kaufmann Publishers, San Francisco (1998)
4. Cypher, A.: Watch What I Do: Programming by Demonstration. MIT Press, Cambridge (1993)
5. Daniel, F., Casati, F., Benatallah, B., Shan, M.C.: Hosted Universal Composition: Models, Languages and Infrastructure in mashArt. In: Proceedings of ER 2009 (2009)
6. Dustdar, S., Schreiner, W.: A Survey on Web Service Composition. International Journal of Web and Grid Services 1(1) (2005)
7. Hoyer, V., Fischer, M.: Market Overview of Enterprise Mashup Tools. In: Bouguettaya, A., Krueger, I., Margaria, T. (eds.) ICSOC 2008. LNCS, vol. 5364, pp. 708–721. Springer, Heidelberg (2008)
8. Janeiro, J., Preussner, A., Springer, T., Schill, A., Wauer, M.: Improving the Development of Service Based Applications Through Service Annotations. In: Proceedings of WWW/Internet (2009)
9. Namoune, A., Wajid, U., Mehandjiev, N.: Composition of Interactive Service-based Applications by End Users. In: The Proceedings of UGS 2009 - 1st International Workshop on User-generated Services at ICSOC 2009, Stockholm (2009)
10. Nestler, T., Dannecker, L., Pursche, A.: User-centric composition of service frontends at the presentation layer. In: The Proceedings of UGS 2009 - 1st International Workshop on User-generated Services at ICSOC 2009, Stockholm (2009)
11. O'Reilly, T.: What Is Web 2.0?, `http://oreilly.com/web2/archive/what-is-web-20.html` (Retrieved April 20, 2010)
12. Ro, A., Xia, L.S.Y., Paik, H.Y., Chon, C.H.: Bill Organiser Portal: A Case Study on End-User Composition. In: Hartmann, S., Zhou, X., Kirchberg, M. (eds.) WISE 2008. LNCS, vol. 5176, pp. 152–161. Springer, Heidelberg (2008)
13. Van der Aalst, W.M.P., Dumas, M., Ter Hofstede, A.H.M.: Web Service Composition Languages: Old Wine in New Bottles? In: The Proceedings of the 29th Conference on EUROMICRO (2003)
14. Wong, J., Hong, J.I.: Making Mashups with Marmite: Towards End-User Programming for the Web. In: Proceedings of CHI (2007)
15. Zang, N., Rosson, M.B., Nasser, V.: Mashups: who? what? why? In: CHI 2008 extended abstracts on Human factors in computing systems, Florence, Italy (2008)