Towards an Automatic Service Composition for Generation of User-Sensitive Mashups

Thomas Fischer and Fedor Bakalov
University of Jena
07743 Jena, Germany
fischer.thomas@uni-jena.de
fedor.bakalov@informatik.uni-jena.de

Andreas Nauerz
IBM Research and Development
71032 Böblingen, Germany
andreas.nauerz
@de.ibm.com

Abstract

In today's Web 2.0, mashups allow users to bring together data and services from various Web applications in order to create a new integrated tool that serves their needs. Nowadays, there is an increasing number of frameworks that provide users with a GUI environment to manually assemble different data sources and services into a mashup. However, in order to create such tools, the user must possess a certain level of technical knowledge. In this paper, we introduce a framework that automatically selects and combines Web services to create mashups. We also describe the user model that stores knowledge about user interests and expertise, which are used by the framework in order to generate mashups tailored to the needs of individual users.

Keywords: Mashups, Portals, Adaptive Hypertext, User Modeling, Persoanlization

1 Introduction

Recently, Web 2.0 techniques have gained wide popularity among Web users. Users are nowadays enabled not only to communicate on the Web, but also to create Web content and tools. One of the most well known examples of such tools created by users are mashups. A mashup can be defined as a situational Web application that extracts and combines data from different sources to support special user needs and tasks [Merrill, 2006], [Zang *et al.*, 2008], [Wong and Hong, 2007], [Ankolekar *et al.*, 2007]. An example of a mashup is Housingmaps[1] by Georg Rademacher, which places housing data from Craigslist[2] on a Google Map. There is an increasing number of frameworks that provide users with a GUI environment to manually construct mashups. However, in order to aggregate different data and services, the user needs a certain level of technical knowledge.

In this paper, we propose a framework for automatic generation of user-sensitive mashups in Web Portals based on the semantic description of information services and knowledge about users interests and expertise. The aim of our framework is to provide Portal users with situational applications that merge enterprise internal and external services to support them with background information and related content. In Section 2, we give a brief overview about current related work. The basic concepts and modules of the Mashup Framework are explained in Section 3. Section 4 and 5 contain detailed explanations on service composition and user modelling for generating user-sensitive mashups.

2 Related Work

The present approaches to mashup development concentrate mainly on either manual or tool supported development. Prominent examples are IBM QEDWiki[3], Microsoft Popfly[4], and Yahoo Pipes[5]. In general, mashups are created manually aggregating different data sources with certain operators. However, the overall functionality is limited to the number of available data sources and aggregation operators. Further, with the increasing number of components, the task to efficiently select, combine, and configure components and operators appropriately becomes more complex and time consuming. Rahm et. al. propose a framework architecture for the development of dynamic data integration mashups, which works on a high-level script language to define mashups [Rahm *et al.*, 2007].

Although there could be some efficiency in reusing existing mashups from a registry like ProgrammableWeb[6], users would be forced to select appropriate ones that meet their current context, interests, and expertise. Probably, there will be no significant reuse of existing mashups anymore, because of their specific orientation to the user, who created them. In our view, mashups are situational applications that should be generated automatically.

Lathem et. al. propose the manual creation of mashups from RESTful services [Lathem *et al.*, 2007], which are described by SA-REST, a novel semantic service description language. However, any enterprise mashup framework has also to consider the SOAPful services in order to leverage strategic investments in Service-Oriented-Architectures (SOA) made by companies. Therefore, we propose to use WSDL as Service Description Language that supports SOAPful and RESTful services through different bindings.

Ankolekar et. al. [Ankolekar *et al.*, 2007] propose that the development of mashups should leverage the technologies of the Semantic Web such as RDF [Manola and Miller, 2004] or OWL [Smith *et al.*, 2004] to overcome the limitations of current mashups. In our view, this goes along with a separation of the (semantic) data and presentation of mashups.

[1] http://www.housingmaps.com
[2] http://www.craigslist.org/
[3] http://services.alphaworks.ibm.com/qedwiki/Microso
[4] http://www.popfly.com/
[5] http://pipes.yahoo.com/pipes/
[6] http://www.programmableweb.com/

3 Mashup Framework

Our mashup framework provides the user with personalized mashups automatically generated based on the knowledge about the user and the information about the context the user is acting in. The framework augments the documents that users are reading with background information and related content gathered from different sources and aggregated into one integrated tool. Imagine a finance manager reading a news bulletin about rumors of a planned merger of her company's competitor with another company and the consequences this has for the stock market. Our framework can augment this news bulletin with the information that could help the manager to make the right decisions. For instance, the framework can display a side menu that shows the situation on the stock market, including the stock quotes of the companies mentioned in the text. Another side menu can be displayed to provide an executive summary of the technology mentioned in the article and the list of people from her department who are familiar with this technology.

Figure 1 illustrates high-level components of the system architecture of our framework. In order to automatically augment the portal content with relevant information, we need a component that is able to extract machine-readable semantics from the content. For this purpose, we use the Calais[7] Web service and UIMA framework (Unstructured Information Management Architecture)[8]. These analysis engines are able to extract certain types of entities, such as person, company, location, etc.

The semantics extracted from the content are used by the personalization engine to identify the concepts that are relevant to the user. The selection of concepts is based on the personalization rules defined in the personalization model and the knowledge about the user interests and expertise stored in the user model.

After the personalization engine has identified the user-relevant concepts, the engine invokes a composition request, where it specifies the selected concepts and sends it to the service composition module. The goal of the service composition module is to find the services that provide background information and related concept on the selected concepts. The composition module creates the composition (mashup model) by an multi-objective evolutionary algorithm, which operates on the semantic service descriptions of the service registry. The service registry maintains the references to the semantic and technical service description of each registered REST or SOAP service. After the composition, the module provides the generated mashup model to the mashup handler, which in its turn invokes the services and requests the presentation module to display mashup on the Web page.

[7]http://www.opencalais.com/

[8]http://incubator.apache.org/uima/

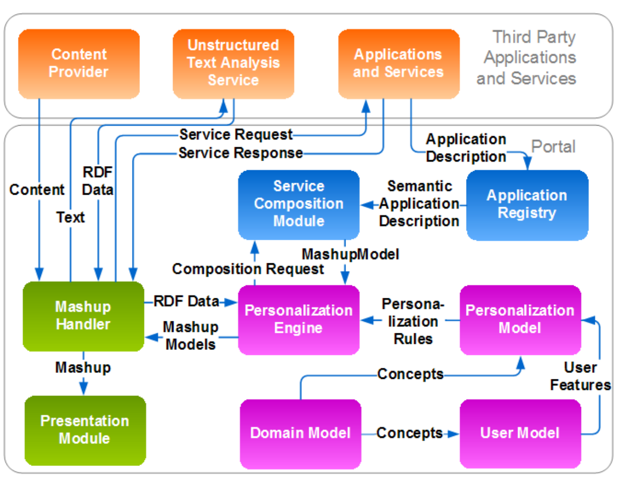

Figure 1: Mashup Framework

4 Service Composition

Our framework provides mashups build upon data retrieved from information services. We propose to use a module for service composition that enables the system to retrieve all user relevant background information because single services will often not be able to provide all needed information. Service composition could be defined as creation of a new composite service from a combination of existing services [Singh and Huhns, 2005], [Küster *et al.*, 2005]. The resulting plan should determine how the system achieves some desired information state.

We first propose several requirements and objectives for an automatic service composition for generation of user-sensitive mashups. The service composition should be correct considering data flow and semantics. Ensuring the semantic correctness depends on whether the semantic service description language is expressive enough to describe the information services. Further it is not enough to evaluate only the inputs and outputs of the information services [Klein *et al.*, 2005], because these describe the services insufficiently. Therefore, the planner should rely on the information state transition of the services.

The composition algorithm should scale for a registry with thousands of services, because the mashup has to be provided immediately to the user. However most AI planning approaches do not achieve this objective[Küster *et al.*, 2005]. A problem simplification could be achieved due to the fact that real world effects are not modeled for information services. A real world effect could be buying a book, or selling an item. Anyhow it is possible that the service provides references to business processes, which provide those functionality.

Further we propose that the planner should also be able to incorporate user prefererences on service properties[Lin *et al.*, 2008]. Example of such preferences could be the quality or price of service. Due to this, the utility of each service and service composition will be different for different users. For example a high data quality and low return time of a stock quote service may have a higher utility for a portfolio manager than a news reporter. The incorporation of utility enables the use of Multi-Attribute Utility Theory (MAUT), which allows to determine the total utility of a service composition based on the preference structure of a specific user. The planner should be able to deal with missing knowledge. This means that the planner also finds services, which does not directly contribute to the target information state, but which are necessary for the overall information flow [Küster *et al.*, 2005].

For that reason we propose to use an multi-objective evolutionary algorithm for service composition that considers the special problem domain and that operates efficiently on the search space. The algorithm uses the semantic service descriptions of the library to determine the correctness of a possible solution. Further it allows the incorporation of user preferences by a special fitness function. In addition it can deal with incomplete knowledge, because it always assesses the possible solution as a whole. This means that the fitness function can determine if a service is important for the information flow of the whole solution, even if it not directly contributes to the target information state.

5 User Modeling and Personalization

In our framework, we aim to automatically generate personalized mashups that satisfy information needs of individual users. For this purpose, we construct a user model that reflects user features, such as demographic characteristics, interests, and expertise as well as a personalization model that defines how the user features and mashup content are matched up. Additionally, we construct domain models that both the user model and personalization model refer to. We have chosen the finance domain for our proof-of-concept implementation. Therefore, in our domain model we define the concepts that the users from financial realm may work with.

Our user model consists of two parts: static and dynamic. The static part defines the user's demographic characteristics, such as date of birth, gender, mother tongue, etc. The dynamic part represents the user's interests and expertise; this part is constructed as an overlay model [Brusilovsky, 2001]: The interests and expertise of an individual user are represented as a subset over the domain model. In the overlay user model, we refer to concepts from the domain ontology and specify fuzzified values that represent the degree of interest the user has in these concepts and also indicate the fuzzified values that show how much expertise the user possesses in these concepts.

In the personalization model, we define the personalization rules that govern what and how the mashup content should be provided to the user. The personalization rules are specified in the Event-Condition-Actions form [Consortium, 1996] shown in Listing 1, where *event* denotes a situation when the user encounters a certain concept in the document she is reading, *condition* is combination of user features and context descriptions, and *actions* are the information gathering-actions that should be provided to the user when the event occurs. In order to combine different user features and context descriptors, we represent them as dimensions. In its turn, the actions are specified at the intersection of these dimensions. E.g., in order to represent a personalization rule for an event when a student of a business school, interested in banking, with no knowledge in this field, is reading a news article that contains a bank in the text, we need to create three dimensions: *User Interests*, *User Expertise*, and *Document Concept* and plot values 'Banking', 'Novice', and 'Bank' respectively. At the intersection of these values, we specify what information should be delivered to the user, which in this case, could be the website of the bank, an encyclopedia article about the bank, and news related to this bank.

```
on
      (event)
if
      (condition)
then
      (actions)
```

Listing 1: Personalization Rule Formula

6 Conclusion

In this paper, we have introduced our framework for automatic generation of user-sensitive mashups, based on composite information services. In our future work, we will evaluate an prototypical implementation of our framework in the IBM WebSphere Portal.

References

[Ankolekar *et al.*, 2007] Anupriya Ankolekar, Markus Krötzsch, Thanh Tran, and Denny Vrandecic. The two cultures: mashing up web 2.0 and the semantic web. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 825–834. ACM, 2007.

[Brusilovsky, 2001] Peter Brusilovsky. Adaptive Hypermedia. *User Modeling and User-Adapted Interaction*, 11(1-2):87–110, 2001.

[Consortium, 1996] Corporate Act-Net Consortium. The active database management system manifesto: a rulebase of adbms features. *SIGMOD Rec.*, 25(3):40–49, 1996.

[Klein *et al.*, 2005] Michael Klein, Birgitta König-Ries, and Michael Müssig. What is needed for semantic service descriptions - a proposal for suitable language constructs. *International Journal on Web and Grid Services (IJWGS)*, 1(3/4):328–364, 2005.

[Küster *et al.*, 2005] Ulrich Küster, Mirco Stern, and Birgitta König-Ries. A classification of issures and approaches in service composition. In *First International Workshop on Engineering Service Compositions (WESC05)*, pages 00–00, Amsterdam, Netherlands, December 2005.

[Lathem *et al.*, 2007] Jon Lathem, Karthik Gomadam, and Amit P. Sheth. Sa-rest and (s)mashups : Adding semantics to restful services. In *Semantic Computing, 2007. ICSC 2007. International Conference on*, pages 469–476, 2007.

[Lin *et al.*, 2008] Naiwen Lin, Ugur Kuter, and Evren Sirin. Web service composition with user preferences. In Sean Bechhofer, Manfred Hauswirth, Jörg Hoffmann, and Manolis Koubarakis, editors, *ESWC*, volume 5021 of *Lecture Notes in Computer Science*, pages 629–643. Springer, 2008.

[Manola and Miller, 2004] Frank Manola and Eric Miller. RDF Primer - W3C Recommendation. "http://www.w3.org/TR/rdf-primer/", 02 2004. visited on May 19th 2008.

[Merrill, 2006] Duane Merrill. Mashups: The new breed of web app. "http://www.ibm.com/developerworks/library/x-mashups.html", 08 2006. visited on March 28th 2008.

[Rahm *et al.*, 2007] Erhard Rahm, Andreas Thor, and David Aumueller. Dynamic fusion of web data. In Denilson Barbosa, Angela Bonifati, Zohra Bellahsene, Ela Hunt, and Rainer Unland, editors, *XSym*, volume 4704 of *Lecture Notes in Computer Science*, pages 14–16. Springer, 2007.

[Singh and Huhns, 2005] Munindar P. Singh and Michael N. Huhns. *Service-Oriented Computing: Semantics, Processes, Agents*. John Wiley and Sons, 1 edition, 2005.

[Smith *et al.*, 2004] Michael K. Smith, Chris Welty, and Deborah L. McGuinness. OWL Web Ontology Language Guide - W3C Recommendation. "http://www.w3.org/TR/owl-guide/", 02 2004. visited on May 19th 2008.

[Wong and Hong, 2007] Jeffrey Wong and Jason I. Hong. Making mashups with marmite: towards end-user programming for the web. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1435–1444, New York, NY, USA, 2007. ACM.

[Zang *et al.*, 2008] Nan Zang, Mary Beth Rosson, and Vincent Nasser. Mashups: who? what? why? In *CHI Extended Abstracts*, pages 3171–3176, 2008.