2007 IEEE Symposium on Visual Languages and Human-Centric Computing

Marmite: Towards End-User Programming for the Web

Jeffrey Wong
Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213 USA
jeffwong@cmu.edu

Abstract

Many information tasks on the web require users to make use of multiple websites, for both content as well as information processing or visualization features. Programmers have created "mashups," which customize or combine the functionality of multiple websites by extracting information from web pages or accessing web services APIs. Non-programmers lack the tools or skills create such customizations because of the programming obstacles involved. Marmite is a tool that empowers non-programmers to create functionality similar to those found in mashups.

1. Introduction

Users often find themselves performing tasks that require them to visit multiple websites. For example, a person interested in a book has several options for getting a copy of that book. He can borrow it from his local library, a nearby city library, or order it from various new and used booksellers, including a local bookstore that has a search engine on its website. Going to each website to search for this book is time-consuming. Clearly, this task could be automated.

Programmers have recognized the value of making use of multiple websites. As websites have begun to offer access to their content and functionality through web services APIs, programmers have used these APIs to create "mashups," websites that combine multiple websites to support unique tasks. Unfortunately, non-programmers are unable to create similar customizations for tasks they encounter.

The web has made computing more interesting by giving anyone instant access to information. Yet the tools to make, manipulate, and reuse this information are still lacking. Spreadsheets empowered non-programmers to create complex numerical calculations that play an important part in the functioning of our modern economy. End-users have yet to be similarly empowered in the realm of working with the vast information resources of the Internet.

Towards this end, we have been developing Marmite, a tool that adapts data-flow and incremental programming paradigms from spreadsheet interaction to mashup construction.

2. Obstacles in Mashup Integration

Creating mashups can be difficult even for those with programming experience. Programming a mashup often requires knowledge of more than one programming language, several markup languages such as XML and HTML, and an understanding of how these elements are assembled together on the web. In addition, a mashup programmer must know arcane conventions that govern how the web is constructed, such as how URLs can be formed to make calls to REST-based APIs. To make mashups served on web pages, one must know how to setup web programming environments on a web server and how the code interacts with a user on a web browser.

While there are several paradigms of end-user programming that can be used to help create mashups (e.g. simplified languages [1] and programming-by-example [2]), we chose to adopt the data flow metaphor, as seen in UNIX pipes in the shell command languages. In our tool, data flows through diverse web services in a similar fashion to data moving different programs in UNIX pipes.

To understand some of the problems that end-users might encounter while constructing data-flows, we conducted a usability test with Apple Automator, a visual data flow tool for Mac OS applications. We found that:

1. Users had trouble finding operations that were appropriate for the data they already had in the data flow. We addressed this in the design by a notion of providing suggested next actions.

2. Users had no feedback of the state of the data between operations as the data-flow was executing. We

addressed this in our design by showing the intermediate results between operations.

3. Execution of data flows is time consuming due to the need to make an HTTP request for each web service call. In our design, we provide fine grain execution control so that users can execute data flows on a small sample of data before committing to executing the flow on a larger data set.

3. The Marmite System

Marmite is plug-in for the Firefox web browser. Web services are represented by blocks known as operators. Operators have inputs and outputs and can be connected together in a data-flow. There are 4 types of operators: sources, filters, processors, and sinks.

"Sources" retrieve data from a web service, perhaps forming a query to a database on behalf of the user. Source operators also include an interactive wizard that guides a user through the process of scraping information off a web page. A "processor" augments or transforms data that flows through it. A "filter" removes elements that do not match some user-specified criteria. "Sinks" are final destinations for the data, such as visualizations.

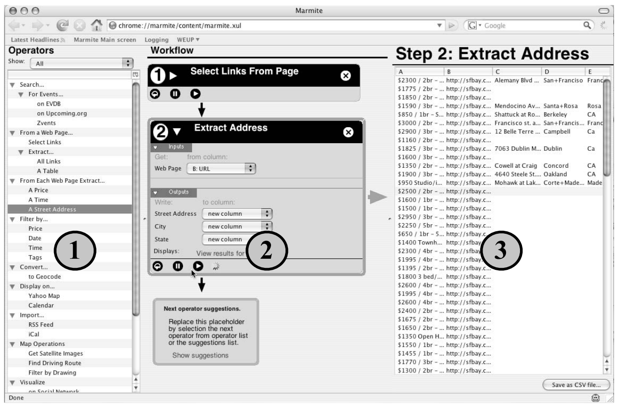
The Marmite screen is shown in Figure 1. Operators are selected from a list on the left and placed in the data-flow in the center. The user can then execute each operation. The panel on the right shows the state of data for each operator and provides live updating

Operators are written in Javascript by programmers for non-programmers. Our vision for Marmite is that programmers or providers of web-services will construct operators that access web-services and share them with the rest of the user community.

Marmite is similar to systems such as Yahoo! Pipes [3] and Microsoft's Popfly [4]. In Yahoo! Pipes and Popfly, multiple flows can be constructed at the same time and merged together, while Marmite is relatively linear. Relatively speaking, Yahoo! Pipes is focused on managing RSS feeds; support for web services requires some basic knowledge of XML paths. Microsoft Popfly also encapsulates web services into programmer constructed blocks. Compared to these systems, Marmite focuses on providing immediate feedback on the state of the data so that users can iteratively construct the data-flow.

4. Formative Evaluation

We conducted a think-aloud evaluation of Marmite with two users from three target user groups: programmers, non-programmers with spreadsheet experiences, and novices who did not fit into either of the two categories. Users were asked to perform 3 mashup-like tasks: extraction and filtering from an



Next operator suggestions.

Replace this placeholder by selection the next operator from operator list or the suggestions list.

Show suggestions

A	B	C	D	E
$2300 / 2br - ...	http://sfbay.c...	Alemany Blvd ...	San+Franciso	Franc
$1775 / 2br - ...	http://sfbay.c...			
$1850 / 2br - ...	http://sfbay.c...			
$1590 / 3br - ...	http://sfbay.c...	Mendocino Av...	Santa+Rosa	Rosa
$850 / 1br - S...	http://sfbay.c...	Shattuck at Ro...	Berkeley	CA
$3000 / 2br - ...	http://sfbay.c...	Francisco st. a...	San+Francis...	Franc
$2900 / 3br - ...	http://sfbay.c...	12 Belle Terre ...	Campbell	Ca
$1160 / 2br - ...	http://sfbay.c...			
$1825 / 3br - ...	http://sfbay.c...	7063 Dublin M...	Dublin	Ca
$1600 / 3br - ...	http://sfbay.c...			
$1350 / 2br - ...	http://sfbay.c...	Cowell at Craig	Concord	CA
$1900 / 3br - ...	http://sfbay.c...	4640 Steele St....	Oakland	CA
$950 Studio/i...	http://sfbay.c...	Mohawk at Lak...	Corte+Made...	Made
$2500 / 2br - ...	http://sfbay.c...			
$1600 / 1br - ...	http://sfbay.c...			
$1500 / 1br - ...	http://sfbay.c...			
$2950 / 3br - ...	http://sfbay.c...			
$2250 / 5br - ...	http://sfbay.c...			
$650 / 1br - 5...	http://sfbay.c			
$1400 Townh...	http://sfbay			
$2300 / 4br - ...	http://sfbay			
$1995 / 4br - ...	http://sfbay			
$1395 / 2br - ...	http://sfbay.c...			
$1800 3 bed/...	http://sfbay.c...			
$2600 / 4br - ...	http://sfbay.c...			
$1995 / 4br - ...	http://sfbay.c...			
$2600 / 4br - ...	http://sfbay.c...			
$2400 / 2br - ...	http://sfbay.c...			
$1675 / 2br - ...	http://sfbay.c...			
$1650 / 2br - ...	http://sfbay.c...			
$1350 Open H...	http://sfbay.c...			
$1550 / 1br - ...	http://sfbay.c...			
$1455 / 1br - ...	http://sfbay.c...			
$1770 / 3br - ...	http://sfbay.c...			
$1300 / 2br - ...	http://sfbay.c...			

Figure 1. The Marmite main screen showing the operator selection list (1), the workflow construction area (2), and the data display area (3).

events database, merging of two databases, and constructing the map visualization from Housingmaps.com, a popular mashup.

Both programmers and one of the spreadsheet users were able to complete all of the tasks within an hour. However, users who failed did not conceive of themselves as constructing a program but rather simply selecting and applying operations to the data, which was presented on the right side of the screen. They would add operators to the data-flow, execute that single operator, and delete the operator. Users had the concept that the data was independent of the operations rather than the system model, which is that data has a "state" after each operator and belongs to that operator. Supporting the need for mashups may be a matter of providing an environment in which one can simply apply operations to live data like a spreadsheet, rather than deliberately constructing a program.

Our future work on Marmite will involve redesigning Marmite to be more like a spreadsheet where operations and flows are secondary to the data. Also, we will be examining methods of automatically or semi-automatically generating operators rather than requiring that they be programmed.

10. References

[1] Bolin, M., Webber, M., Rha, P., Wilson, T., and Miller, R. C. "Automation and customization of rendered web pages." *Proc. UIST '05*. ACM Press, New York, NY, 2005, pp. 163-172.

[2] Little, G., Lau, T. A., Cypher, A., Lin, J., Haber, E. M., and Kandogan, E. "Koala: capture, share, automate, personalize business processes on the web." *Proc. CHI '07*. ACM Press, New York, NY, 2007, pp. 943-946.

[3] Yahoo! Pipes. http://pipes.yahoo.com

[4] Microsoft Popfly. http://www.popfly.ms