Mashups: Who? What? Why?

Nan Zang
College of IST
Penn State University
University Park, PA 16802
nzz101@psu.edu

Mary Beth Rosson
College of IST
Penn State University
University Park, PA 16802
mrosson@ist.psu.edu

Vincent Nasser
College of IST
Penn State University
University Park, PA 16802
vwn102@psu.edu

Abstract

In recent years major web services have opened their systems to outside use through the implementation of public APIs. As a result, web developers have begun to experiment with *mashups* — software applications that merge separate APIs and data sources into one integrated interface. Because the APIs and data sources are publicly available, in principle anyone can create a mashup. However, because relatively advanced programming languages are required to integrate these APIs, creating a mashup still requires considerable programming expertise. In this paper we share the results of an exploratory study of web developers and their experiences with building mashups. We profile the characteristics of mashup developers, examine the mashups they create, and the reasons they create mashups. From the results of this initial survey we outline a course for future research.

Keywords

End-user programming, data manipulation, design, survey, motivation.

ACM Classification Keywords

H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

Introduction

The past few years have witnessed a growing interest in integrated web applications, popularly known as mashups. These mashups are a coalescence of different data sources and application programming interfaces (APIs) into an integrated end user experience. One typical example of a mashup is the overlaying of weather data on a map (fig. 1); the integration "mashes" a geospatially-indexed temperature feed with the Google Maps user interface. Another typical example is price data collected from several different shopping sites that is merged into a single list to help consumers find the best deal on a product (see e.g., http://secretprices.com).

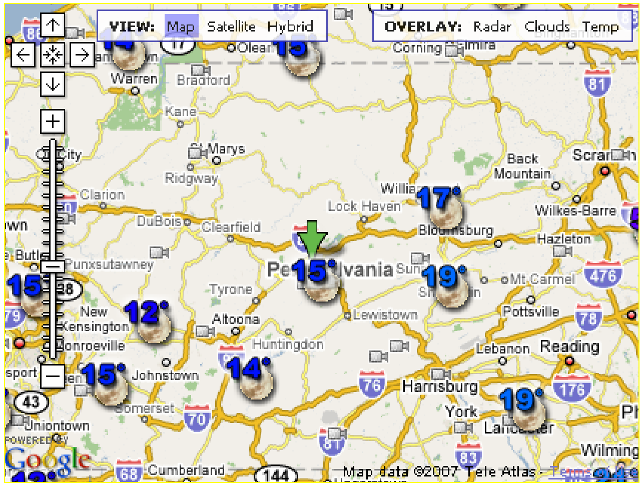

figure 1. Example of a map mashup (taken from http://weatherbonk.com)

Research on mashups typically falls under two themes. One is information systems (IS), where mashups are built in response to a specific need like map synthesis [3], enterprise information integration [5], or web service tracking [10]. The focus in this work is on identifying the relevant feeds and extracting the most useful information to fulfill the IS requirement. A second theme is end-user programming (EUP) [6, 9, 11]. The goal of EUP research is to allow "end-user programmers", people who may write code but are not professional programmers, to use programming techniques as part of accomplishing their task goals [8]. A pervasive issue in EUP systems is the flexibility of the support provided. No software package can fit every need of every end-user. Thus customization may be necessary for an end-user to complete his or her task. Because mashups are essentially customizations of data, they can be considered as a type of web EUP.

The current EUP research into mashups has concentrated on developing tools to support mashup creation. The general interest in mashups as an end-user software topic is evidenced by workshops at major conferences, including CSCW 2006 and CHI 2007. This interest is not unwarranted; as of January 4, 2008, ProgrammableWeb, a site dedicated to tracking mashups, lists 591 available APIs and 2636 existing mashups [1].

With the large number of APIs, and the vast and increasing amounts of web content available, mashup technology allows web developers to create a variety of customized, novel web applications. However, because considerable programming expertise is needed to leverage these web technologies, a large population of less sophisticated end-users is unable to benefit. Moreover, while several studies have explored tools to support mashup creation [4, 7, 11], few have considered the needs of less sophisticated end-users.

Why do people create mashups? What do they create? Could mashups be a useful technique for end users more generally? These are some of the questions yet unanswered.

In this paper we discuss the results of an exploratory survey aimed at understanding the mashup developer population. We suggest that by investigating the self-reported experiences and processes of these users, we can better understand the motivations of mashup developers and the characteristics of the applications they create. Furthermore, we can apply the lessons learned by these developers to generate research questions and approaches for future research.

Methods

To investigate the mashup developer population we created an online survey that queried developers about their experiences when creating mashups using online APIs. The survey included general questions that deal with both mashup creation and general web development. To better understand the mashup developers' backgrounds, we asked how they learned to develop web pages and mashups. Furthermore, we speculated that online documentation would be the main source of assistance for mashup developers, so we also incorporated questions about their experiences when browsing online documentation.

Recruitment

We advertised our study using online forums and discussion boards tailored towards mashup developers, because we expected that these would evoke the most likely interest and response. Initially, to ask for assistance in distributing our survey, we contacted 13 forum moderators, including ones for the Google Maps API, Flickr API and DaniWeb. Of those moderators, 4 agreed to include a link for our survey as part of an announcement to the users of the forum. In cases where the moderator did not respond, we created a thread and posted our survey recruitment in hopes that forum members would see the message without an announcement. In total, we successfully posted in 15 forums.

Results

For most questions in the survey, the number of responses (N) changes because not every participant completed every question. In fact, about half of the questions were presented only to developers who agreed that they had created at least one mashup. To simplify results presentation, we use percentages, with the relevant N varying from a low of 24 to 63.

Participants

At the end of the survey period we had gathered 63 responses (N=63); 31 participants reported that they had created mashups. In our sample data, the average age of the participants is 33 years with 88.9% being male and 67.9% having at least a university degree. Over half of our participants (53%) cite work as the reason they develop web pages. For the participants who have created mashups, the average is 34 years with all of them being male and 76% having a university degree or higher.

Programming Experience

When asked about their use of specific web technologies and their programming experience, on a scale from 1 to 4, where 1 is never and 4 is daily, over half of the participants reported they frequently or daily use web scripting and programming. However, they

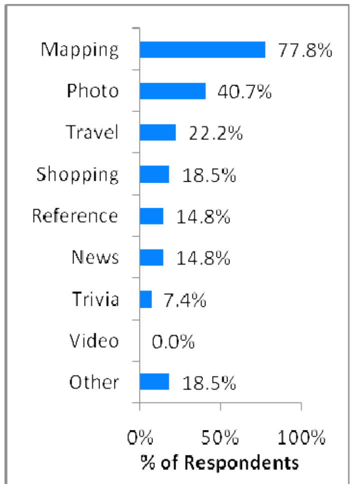

figure 2. Types of Mashups created

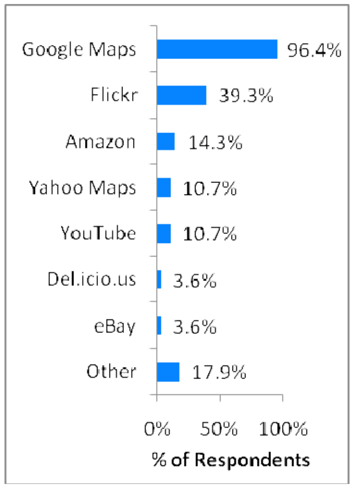

figure 3. APIs used in development

never or rarely program desktop applications or using web-based APIs. Specifically over half of the responses suggest participants often use HTML, CSS, and advanced web programming languages (PHP, etc), but less than a third have frequent exposure to programming a desktop application or developing using a web-based API. Not surprisingly, mashup developers tend to participate in programming activities more frequently. Interestingly, half of these mashup developers do not use web-based APIs on a regular basis. These results are shown in Table 1, where each percentage represents those who participate in the activity frequently or on a daily basis.

	All Developers (N=61)	*Mashup Developers (N=30)*
Use a Web Page Editor	54.4%	58.6%
Use HTML/XHTML Only	56.9%	57.1%
Use CSS	71.5%	78.6%
Use Web programming language (PHP, ASP, etc)	75.0%	73.4%
Program Desktop Applications	32.8%	44.8%
Programmed using web-based API	28.1%	50.0%

table 1: Frequent programming activities

Types of Mashups and APIs
One specific detail we are interested in is the APIs most frequently used and the types of mashups being created. Our data indicates that the majority of the developers surveyed are creating map mashups, with some creating photo mashups (fig. 2). Google Maps is by far the API most commonly used (fig. 3).

Problems when Developing
Another area of interest is the process of mashup creation. What are the problems that developers encounter when creating mashups? In analyzing the open-ended responses given by 20 developers, we found 3 overarching themes: the reliability of the API, documentation, and coding details. API reliability refers to frustrations developers have when dealing with APIs. Our participants described a variety of issues include authentication problems, performance degradation, and dependencies on the "hosting company's priorities for new feature development". The lack of proper tutorials or examples for each API is also a concern for developers. Finally, the JavaScript skills needed to integrate the APIs can be a major hindrance; as one participant declared, "You have to be a JavaScript Freak!"

Learning to Mashup
When learning a new API, mashup developers would most likely go to the documentation provided by the API. When looking at documentation, mashup developers consider the accuracy of the document and the supply of examples to be most important to its usefulness. When prompted specifically for suggestions about how to improve the documentation for the Google Maps API, developers asked for "examples of working code" and "graduated information for

beginners to experts by level". When asked where they learned to develop mashups, all those who answered the question reported that they were self-taught.

Discussion

While the sample size of our survey is small, we can draw some preliminary conclusions about the mashup developer population that can assist future research.

In general, mashup developers are predominately male and on average have at least a college degree. Their expertise seems to be in web-related technologies rather traditional desktop software development. More interesting is that even those who have created mashups use web-based APIs infrequently. One possible reason for this is that while mashup technology is novel and provides web developers with a general ability to integrate data, current APIs may not be seen as useful enough to warrant frequent use [11].

Our results also show that an overwhelming portion of existing mashups involves maps, typically the Google Maps API. Although our data does not clearly show why this occurs, we can surmise that it is because maps are the most visual and adaptable of the mashup options. Many types of data can be manipulated to extract or generate location information and thus are easily plotted on a map [3]. Other mashups, such as video and shopping have limited domain-specific uses.

Another important finding is the problems encountered when developing mashups. Of the three sources of problems, those involving programming difficulties are ones that can be resolved by a tool. There are already online tools that allow users to create a mashup without programming. However, most of the current tools only provide the ability to manipulate data or the ability to create visualizations for the data. Few allow the end-user to complete all their tasks in one place. Furthermore, while XML and RSS are efficient avenues of data transfer, passing data from one server to another adds an additional degree of complexity and requires the resulting mashup to depend on the reliability of an additional service provider. In regards to documentation, tutorials, examples and other help documents can also be supplemented into a tool. However, due to the rapid changes that are possible with web-based systems, it would be difficult for third-party documentation to maintain its accuracy.

The way that developers learn how to create mashups revolves around the documentation provided by each API. Accurate documentation is vital for the development of mashups; without it, there would be no understanding of how an API operates [2]. However, documentation is still quite sparse. Providing more comprehensive tutorials and examples could assist developers in the early stages of learning. Furthermore, the introduction of tiers of documentation could help transition novice developers to more advanced levels.

Future Work

The results of this survey have given us a modest glimpse of how mashup developers currently operate. However, the data also point to some aspects of mashup development we have yet to uncover. In particular, while we may try to guess why developers create so many mapping mashups, we do not fully understand the motivations involved. Are mashups truly useful to the average end-user? Or are they just a novel toy to play with?

Thus far, we have only looked at a population of web developers that would be considered relatively advanced programmers. Our next step is to seek a better understanding of a more novice end-user population. Will less sophisticated users be interested in the same sorts of mashups as the developers we surveyed here? Or will they come up with other novel ways to integrate and present data? We plan to conduct focus groups where we will present novice end-users with different types of data, and ask them to combine two or more types and create a visualization to display the result. With further study of mashups from a novice end-user perspective we can better gauge the usefulness of mashups as a whole and find ways to support their activities.

Also, in light of the preliminary results of this small survey, we are in the progress of revising and extending the survey to focus on some of the more dominant themes including documentation and problems encountered during development.

Programming is a difficult task, but programming a mashup introduces many factors that are out of the end-user's control. While as researchers, we may not be able to resolve the inherent reliability issues of web-based APIs, it is possible to serve the end-user by developing a tool that supports their programming needs.

Acknowledgements

We thank all online forum moderators who helped distribute this survey and the helpful members of the Google Maps API forum who guided us when we created our first mashup.

References

[1] *ProgrammableWeb*. http://programmableweb.com/, (Retrieved January 4,2008).

[2] Bloch, J. How to design a good API and why it matters. In *Proceedings of the Companion to 21st ACM SIGPLAN* (Portland, Oregon, USA, 2006). ACM Press.

[3] Elson, J., Howell, J. and Douceur, J. R. MapCruncher: Integrating the World's Geographic Information. *ACM SIGOPS Operating Systems Review*, 41, 2 (April 2007), 50-59.

[4] Ennals, R. and Garofalakis, M. MashMaker: Mashups for the Masses. In *Proceedings of the SIGMOD '07* (Beijing, China, June 12-14, 2007). ACM.

[5] Jhingran, A. Enterprise Information Mashups: Integrating Information, Simply. In *Proceedings of the VLDB '06* (Seul, Korea, September 12-15, 2006). ACM.

[6] Lingam, S. and Elbaum, S. Supporting End-Users in the Creation of Dependable Web Clips. In *Proceedings of the WWW 2007* (Banff, Alberta, Canada, May 8-12, 2007). ACM.

[7] Murthy, S., Maier, D. and Delcambre, L. Mash-o-matic. In *Proceedings of the DocEng '06* (Amsterdam, The Netherlands, October 10-13, 2006). ACM.

[8] Myers, B. A., Ko, A. J. and Burnett, M. M. *Invited Research Overview: End-User Programming*. ACM, City, 2006.

[9] Sabbouh, M., Higginson, J., Semy, S. and Gagne, D. Web Mashup Scripting Language. In *Proceedings of the WWW 2007* (Banff, Alberta, Canada, 2007). ACM.

[10] Tatemura, J., Sawires, A., Po, O., Chen, S., Candan, K. S., Agrawal, D. and Goveas, M. Mashup Feeds:: continuous queries over web services. In *Proceedings of the Proc. 2007 ACM SIGMOD* (Beijing, China, 2007). ACM.

[11] Wong, J. and Hong, J. I. *Making Mashups with Marmite: Towards End-User Programming for the Web*. ACM, City, 2007.