Mashups on the Web: End User Programming Opportunities and Challenges

Nan Zang
The Pennsylvania State University
nzz101@psu.edu

Abstract

In the last 10 years, the social nature of the web has created a shift in the way people use the computer. In particular, advanced programming activities are no longer reserved for those who are professionally trained programmers. Moreover, the increased socialization of the web has encouraged users to create more readily available content for everyones use. However in some cases what a user wants is not necessarily easy to accomplish and in many cases still re-quire programming skills.

In this paper, I describe some past studies of web mashups, an integrated web application that combines data from multiple sources into a single interface. Mashups provide us with a unique opportunity to study how both professional programmers and non-programmers approach an inherently programmatic technology. In some cases the issues encountered by the end user coincide with those of the programmer. Using lessons learned from a survey study, interviews and a think-aloud study, I propose directions for future research.

Categories and Subject Descriptors D.m [*Software*]: Miscellaneous – Software Psychology

General Terms Human Factors

Keywords End-user programming, mashups, APIs, and tools

1. Introduction

The Web, without a doubt, has a major influence on our every day lives. We use it as a tool to gather information and as an avenue of communication. We use it for work and for entertainment. It is everywhere we go, and over the past few years it has become a ubiquitous source of information. This abundance of information on the web is in part due to the evolution of the Internet as a communication medium, but also due to the huge number of online authoring tools and services that has encouraged the average web user to share their own creation on the web. For example, in January 2009, the video sharing web site YouTube announced that there is an average of 15 hours of video uploaded every minute [1]. Essentially, the web is designed for the active participation of its users; the more users, the more content that is created, and the more useful the web becomes.

However not all of these new technologies are easy to understand and use. Many of the novel systems introduced require skills that not every end user may have. For example, web Application Programming Interfaces (APIs) – one of the core technologies used by almost every major web system, including Google, Yahoo, Microsoft and Amazon – requires advanced programming skills to employ. One approach to solving this skill barrier is to introduce easier to use or simpler tools. However, what happens when this easy to use tool does not meet the requirements of the user? The user then needs to find and adopt another tool for a specific task. What if the user cannot find a tool that fits his or her needs? Moreover, with each additional tool needed, the user must commit cognitive resources to learning and remember its use. As with any technology or tool, there may be a substantial learning curve before one can reap the benefits.

Generally, software developers cannot come close to supporting every need of every user in the tools they create. Tasks are generalized and simplified, and tools are created for the common needs. Instead of introducing a new tool for every need, developers employ numerous methods to help users help themselves. For example, in spreadsheet software macro writing functionality is introduced to allow users to create their own solutions to repetitive tasks. However, again this solution requires some technical skills. In most cases, users still need to maintain a level of programming or computational expertise to leverage these features.

The remainder of this paper examines our work in the area of end-user programming. While this work focuses on more novice programmers or non-programmers, we found that in some cases both programmers and these end users

Copyright is held by the author/owner(s).
PLATEAU '09 Workshop at Onward! 26 October 2009. Orlando, Florida, USA. ACM .

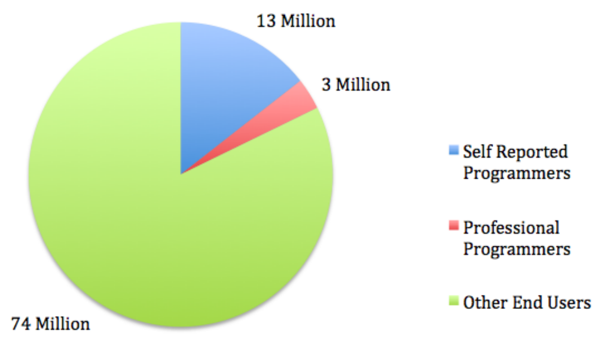

Figure 1. Number of US Programmers and End Users by 2012

encountered similar problems when dealing with programming tasks on web.

2. End-User Programming

The term end user can have many different meanings based on the context in which it is used. In particular, our work has focused on a subset of end users primarily identified as end-user programmers. This group is typically defined as people who participate in programming tasks but whose primary job function is not programming. One early definition of this population explains end users are not:

> “ ’casual,’ ’novice,’ or ’naïve’ users they are people such as chemists, librarians, teachers, architects, and accountants, who have computational needs and want to make serious use of computers, but who are not interested in becoming professional programmers. [2]”

However, this description has evolved over the years. As reflected by the research in the early 1990s, the majority of computer users and programming was done either around workplace or in schools [2]. Thus many of the end users described during that period were either professionals who need to get work done or students who are being taught using a computer. If we think about these topics in todays world, the end user population is far more diverse.

Using data from a survey conducted by the US Bureau of Labor Statistics, researchers have estimated that by 2012 there will be over 90 million Americans using a computer at work, with 13 million self-reported programmers, but less than 3 million professional programmers [3]. With this increasing user population, programmers cannot hope to create systems that fulfill the nuanced needs of every user. Instead, designers and developers must find a way to allow users to help solve their own problems. This is the underlying goal of end-user programming (EUP) research.

3. Mashups

Recently there has been an upsurge of research interest in mashups - a web application that combines multiple data sources and presentations into one interface. From an EUP perspective, mashups provide users with the opportunity to integrate and assimilate many different web resources into a personalized view. However, because of the relatively complex programming methods needed to create a mashup, more non-programmers and novice users are left out. Specifically, programmers usually leverage APIs or screen scraping to retrieve online data for a mashup. As a result, numerous tools have been created and studied that attempt to help these naïve end users to create these advanced web applications [4, 5].

Mashups are interesting to us because they provide an opportunity to study end users working with cutting edge technology. Moreover, the goals of a novice user creating a mashup could easily match with the motivations of a more experienced programmer. I would suspect that the majority of the 4361 mashups listed on ProgrammableWeb – a website that tracks mashups – could be used end users to accomplish a variety of goals and complete various tasks on the web [9]. However, the web applications listed there were created by programmers; novice end users do not have the skills necessary to create such mashups. Again, the end users must turn to and rely on developers.

4. Survey of Mashup Creators

We began our inquiry into mashup development by first surveying the developer population [10]. Because mashups tend to be context-specific, we assumed that mashup development would be a solution used by developers, but not a primary job function. So we advertised our survey on numerous web development and API related forums. This effort resulted in a total of 63 responses, with 31 who had created mashups before. When we compared all the developers to the 31 with mashup experience, we found that the 31 developers had more expertise with programming and web technologies.

As a part of this study, we wanted to understand how these developers learned to create mashups. As to be expected for a novel technology, our surveyed developers all taught themselves using the documentation for different APIs. Probing more deeply, we found that these developers found that the documentation is very inadequate and do not provide the correct level of abstraction. Documentation was cited as being very sparse and not regularly updated. One participant specifically pointed out that the Google Maps API would be updated but the documentation did not reflect the changes until much later. Moreover, many developers pointed out a lack of examples and tutorials. Our participants expressed that they realize that they are taking advantage of free and public services, but stress that when approaching a new web service and API, they immediately look for examples and use those to explore. Further, they depend on online documentation that when they are building mashups, as there was no formalized training in mashup development during the length of the survey. They suggested that having graduated information from beginners to experts by level and a variety of examples that showcased the API's functionality would have helped them when learning to create mashups.

Another insight from this work was that the mashups created by developers are quite limited in variety. There are a limited number of APIs being used for the majority of mashups, and many mashups have only slight differences to each other. The Google Maps API and mapping related mashups were the most common. While it is unclear why this was the case, it is possible that the visual nature of maps combined with the Google Maps API being one of the first major public APIs made this type of mashup more attractive to developers. It may be that while there is an abundance of data on the Web that is interesting, there are fewer useful visualization techniques that are accessible to non-programmers. From a tool developer's prospective, one possible way to further support end users could be to provide these users with simple ways of integrating their data with appealing visualizations.

Overall, we concluded that developers currently use mashups as a solution to data integration problems. The numerous public APIs available serve as a toolset for developers to create interesting applications. The problems encountered by the developers seem to arise from interacting with the APIs and not necessarily with any specific programming language used to create a mashup.

5. Survey of End Users

To more carefully study a novice user population more aligned with EUP research, we distributed a survey to the student population at Penn State University [11]. We speculated that these students would be the types of users that could benefit from being able to create mashups. They grew up with the Internet being prevalent in their lives, and they rely on the web as a tool to solve problems and gather information.

From this study, we identified some key variables that influence how end users approach and think about novel web technologies: Technology Initiative – how active do they pursue new web experiences, and disseminate these to their friends; Usefulness – a self-rating of perceived usefulness of a technology; Sharing hobbies online – the frequency of sharing their own hobby related activities and creations online. Of these three, Technology Initiative and Usefulness seem to play an important role in judgments our participants make about pursuing mashups. We also asked them to rate the difficulty of creating a mashup, assuming that this would be a variable related to their motivations for pursuing mashups. However, this was not the case, as difficulty did not appear to have a major effect on their decisions to pursue mashups in the future.

This study resulted in numerous insights, but most important was the concept of web-active end users. These users are extremely active online and pursue all aspects of online life. While this is a common quality among many college students, the web-active user is motivated to take the additional step by finding better tools to support and enhance their common activities. Moreover, not every web-active end user is the same. When we compared those who had high Technology Initiative to those with lower Technology Initiative, we found that their interests diverged. Users with lower initiative tended to describe social- and people-motivated mashups. For example, one participant wanted a way to bring together all of the different social networking services she used into one interface. Another participant combined pictures of his friends with famous quotes. High initiative users focused on more complex, data-intensive mashups. For example, using news articles as a reference point and integrating additional reference materials, such as Wikipedia articles, other news sites, and video from YouTube. Another participant suggested incorporating product reviews, and missing specs of products when browsing online retailers. This reinforces and clarifies some of the details from the prior study [10]. The types of mashups end users can easily create are important for their initial perceptions of a technology or tool. Being able to support all mashups equally may not be as important as being able to support a specific set of mashups. Focusing support on mashups that are clearly useful to the end user will most likely encourage them to adopt a tool.

A problem area for end user mashups was how end users think about data. We asked them to naïvely describe the steps needed to create a mashup. While most of the respondents could generally describe how to gather data and what to display at the end, the integration of data was not well defined. We decided to study this in more detail since it is the most vital part of the mashup building process.

6. Yahoo! Pipes: Mashups for EUP

The process of building a mashup can be decomposed into three stages: data gathering, data manipulation, and data presentation. End users can easily understand both gathering and presentation; one assumes that to they would know what data they want to mashup and the resulting visualization before approaching a mashup task. To examine data manipulation in more detail, I interviewed 12 students who fit the web-active end user program and also had them create the mashup pictured in Figure 2 using Yahoo! Pipes. I chose Pipes partially due to its focus on data integration, but also because it is essentially a visual programming language. A more detailed report of this work and the results have been reported in [12]. I will briefly review it here.

I began by asking the participants to describe some of the web sites they frequently visited. To assist them I provided five categories of information: News, sports scores,

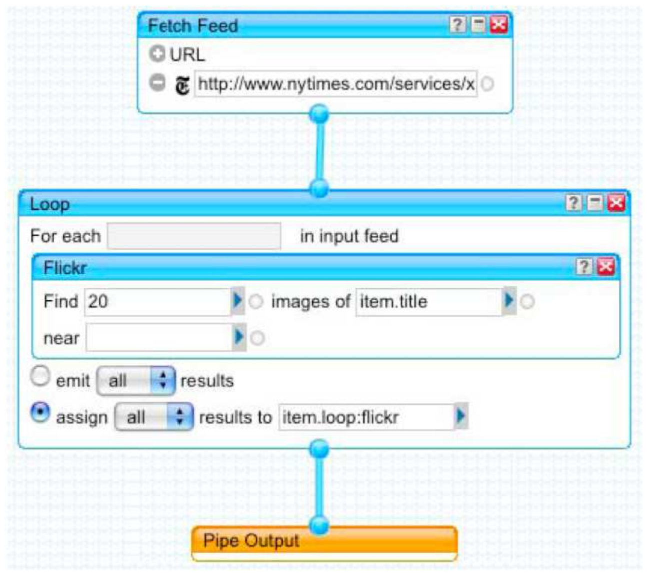

Figure 2. Mashup of New York Times and Flickr using Yahoo! Pipes

weather, shopping, and pictures. They were asked to come up with places online they could find each and then if there were any categories or websites that did not fit. For the most part, users were able to come up with a wide variety of data sources for each category. There were a total of 64 unique information sources. As we would expect, for each of the categories the most popular websites were also the ones mentioned the most by our group. For example, when thinking about shopping information the participants looked to Amazon, and for sports they used the ESPN website. However there were many more obscure webpages mentioned. A common theme in the responses were close connections to local, real world locations. For instance one participant recently moved and no longer had access to a physical copy of her previous town newspaper, but she now she would visit the newspaper's website as a substitute. Websites for local clothing and electronics stores were referred to as common sources of shopping information. I purposely left out social networking to see if the participants would mention the category. All but one referenced social networking and Facebook as one of the sites they spend most of their time on. The participants cited portal services, like MSN, iGoogle, Yahoo, and even Facebook as a common location for much of their general information needs.

When thinking about combinations of data the participants brought up a variety of examples. For example, one participant suggested combining course listings and scheduling with reviews from Rate My Professor. Interestingly, there were few mentions of ideas involving maps and other geographic data; only one participant mentioned location based customizations. This is unexpected because of the large number of mashups created thus far that have been map-based. This points to certain discrepancies with the tools being built. While expert programmers may concentrate on mapping-related mashups, the more naïve and non-programming users focus on other areas. For example, Shopping and Picture based combinations were mentioned the most during the interviews.

Finally, to gauge their understanding of the common underlying data structures on the web, I tested each participant on their comprehension of XML. The reason for this is to better gauge their understanding of the data they were working with in a raw form. Would it be possible for users with similar expertise to understand and then possibly work with XML without much support. I found that in almost every case the participants were able to identify the attributes and content in the code. From their explanations, it seemed that they used tags as a guide to understanding the content, but then made qualify judgments by comparing the two. For example, if a tag was marked "title" and then the content was a garbled string, they would second guess themselves and say "I think it's a title [...]" but not be completely sure, even though they correctly identified the title tag previously. Also, the participants had a hard time understanding HTML embedded into the XML. For example, "description" tags, commonly used as a message body for RSS feeds, many times contain links and images. The HTML representations were often dismissed as being "some code". Generally, the participants were able to pick apart the XML and build an understanding of the underlying data. This is promising because many of the currently available mashup tools, including Yahoo! Pipes, hide the underlying data, claiming that it is added complexity. But these results tell us otherwise. With the proper scaffolding, a novice end user could use the underlying data structures to make better connections between different data sources.

Once they began thinking about different types of data and combining data, I asked them to create a mashup using the Pipes tool. They were given a short tutorial of the interface and then 15 minutes to familiarize themselves with the tool. I used a think-aloud protocol to get at the participant's thought processes while they worked with Pipes. Their actions revealed a mismatch between the Pipes tool to their mental models. Specifically, the names of each model proved to be a major factor in how the end user approached the mashup task. For instance, many users used the Filter module to take the functionality of the Loop module in the mashup in Figure 2. Because theyre deci-sions were solely based on their interpretation of the module name, Filter seemed like a way to "find something that is related to whatever is on the [article] page". Of course, computationally filter actually removes unwanted items in the input feed. This result is similar to prior studies of language use in programming systems [6]. Given enough time, a user may be able to master each module, but currently, they may not even get that far.

7. Discussion

Our work thus far has focused on these Internet users and their perceptions and use of information on the Web, specifically focusing on mashups. From our surveys we realized that mashups are not only for non-programmers, but also those who are experienced developers. Dealing with the myriad of APIs on the web is a test of patience. The lack of adequate documentation, along with the dependency on services beyond the developers control makes mashup development quite difficult. Both non-programmers and developers must have a place to start, and as a relatively new platform the Web poses added difficulty for those who want to program for it. When we looked at end users specifically, the characteristics of a population we call web-active users play an important role. We believe that they are a population of users that are motivated to take on more activities, including even learning to build mashups, if they judge it to be worth while.

A limitation of this work has always been the introduction of the idea of mashups to our participants. It is extremely difficult to describe concepts to a person without giving examples and context. These examples in both our end user survey and in the interview study are biased by this. It most likely would be true that if we introduced a mapping example that participants would think about location based mashup ideas. However, given this possible source of bias, we were still able to illicit many ideas that are unrelated to our initial introduction. We believe that many of our results still provide valuable ideas for developers.

From a tool developer's prospective, it would make sense build a better understanding of this user population and design tools that target their specific motivations. This could include providing simple ways to visualize the data they work with, or packaging information in such a way that is engaging to them. There are currently a small number of tools that really support end user programming on the web, and even few for creating mashups. Tools like Yahoo! Pipes and Microsoft Popfly use visual programming languages to alleviate some of the initial learning curve of code. Using direct manipulation seems to be an accepted method, but does adding these extra layers of abstraction truly help the end user. In particular, as our work has found, many times hiding the underlying "code" may cause even more confusion. From our work we can easily conclude that Pipes is not a tool for non-programmers. The way in which the tool is organized and the modules named suggests that it was mostly built simplify data aggregation for developers. For a less programming savvy user to take advantage of Pipes he/she must be able to specify actions in a programmatic way. This may not be possible for this group of end users; there is a underlying difference in the way in which programmers solve problems.

From our work with mashups, we realize that the basic concepts of this technology are not necessarily a new idea, but the way in which developers integrate APIs and create visualization is quite novel. As a result, even seasoned programmers must learn new techniques. To a greater extent, end users must learn as well. The key difference here is that programmers have already developed a baseline of what is commonly called computational thinking [7]. It is this mindset that allows programmers to solve problems using programming languages. Relating to our work, programmers can decompose problems in a way that acceptable by computer systems. While most EUP research focuses on lowering the skill boundary for end users by developing easier to use tools or creating programming languages that better match the mental models of the developer, it may be more important to find ways to raise the basic level of computational thinking for end users.

Generally computational thinking is a term reserved for educational domains, but I believe it would be useful when thinking about end-user programming. End users must learn additional skills when approaching new tools, regardless of the activity. If this is the case, why not provide additional support that teaches skills they can use in other situations? However, a significant problem is that users must be motivated by their task enough to learn something new. Our work has pointed to possible ways of engaging these users by leveraging their interests and taking into consideration existing activities. By continuing to build extremely simple tools we perpetuate the expectation that everything must be "easy". In the long run, it may be more beneficial to encourage users to increase their computational skills.

8. Conclusion and Future Work

This work has highlighted some key areas that EUP research is taking when working towards addressing end users problems. Great strides have been made to better understand end user programming activities, and many tools have been created to support the specific tasks of the end user that leverages advanced and novel technologies. However, much of the recent work in this area focuses on the building new tools, but not realizing the needs of the end user. Rather than concentrate on the tools and technologies surround the web, we should seek better ways of engaging the user. It does not matter how novel a tool may be, if there are no users then there is no clear impact.

While usability is a concern, this work has also found that there is very little emphasis on educating users as they continue to explore the web. The EUP tools that have been built still require a programming or computational mindset. The way in which the driving technologies operate will continue to be based upon programming concepts. By supporting and even encouraging users to take an additional step and pursue a better understanding of these technologies would surely help alleviate some of the skill barriers to programming.

Our next step is to broaden our prospective to look at the process that users go through to gather information, integrate it into a meaningful form. A survey study is currently being conducted to investigate how web-active users mentally decompose their activities online. This is akin to creating step-by-step instructions. This survey should allow us to better understand how users think about the information they are working with and how they naturally think about their information processes on the web. We hope to follow up the survey with a paper prototyping task where the participants will work with snippets of information and perform analysis on them in order to accomplish a task. This method may allow us to actually investigate their information integration processes without the complexity of computer interfaces.

Acknowledgments

This research was partially supported by The National Science Foundation (CCF-0405612).

References

[1] E. Schonfeld. YouTubes Chad Hurley: We Have The Largest Library of HD Video On The Internet. *TechCrunch*, January 30, 2009.

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

[3] C. Scaffidi, M. Shaw, and B. Myers. An Approach for Categorizing End User Programmers to Guide Software Engineering Research. In *ICSE'05 Workshop on End-User Software Engineering* ACM Press, 2005.

[4] J. Wong, and J. I. Hong. Making Mashups with Marmite: To-wards End-User Programming for the Web. In *CHI'07* ACM Press, April 2007.

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

[6] J. F. Pane, and B. A. Myers. Usability Issues in the Design of Novice Programming Systems. CMU-HCII-96-101, Carnegie Mellon University, Pittsburgh, PA, 1996.

[7] J. M. Wing. Computational Thinking. In *Communications of the ACM*. volume 49, issue 3. ACM Press, 2006.

[8] A. Cypher. *Watch What I Do: Programming by Demonstration*. MIT Press, 1993.

[9] ProgrammabeWeb. http://www.programmableweb.com. Retrieved Oct 8, 2009.

[10] N. Zang, M. B. Rosson, and V. Nasser. Mashups: who? what? why? In *CHI '08 Extended Abstracts*. ACM Press, April 1, 2008.

[11] N. Zang, and M. B. Rosson. Whats in a mashup? And why? Studying the perceptions of web-active end users. In *IEEE Symposium on Visual Languages and Human-Centric Computing 2008*. September 15, 2008.

[12] N. Zang, and M. B. Rosson. Playing with Information: How End Users Think About and Integrate Dynamic Data. In *IEEE Symposium on Visual Languages and Human-Centric Computing 2009*. September 20, 2009.