Towards Communities of Practice for Mashups

Leif Singer
Leibniz Universität Hannover
Welfengarten 1
30171 Hannover, Germany
+49 (0) 511 762 - 4793
leif.singer@inf.uni-hannover.de

ABSTRACT

Many integration projects in enterprises are too small to warrant their own implementation by IT. This leaves a "long tail of enterprise integration" unaccounted for. To exploit this potential, we propose a community of practice for end user development that will be able to solve their integration needs on their own. In particular, we want to combine a spreadsheet-oriented, browser-based mashup tool with a social network site designed as a company-internal collaboration platform. This should permit many small local integration projects, performed by end users. Employee needs that were too expensive to consider before would then be satisfiable.

Categories and Subject Descriptors

H.5.3 [**Group and Organization Interfaces**]: Computer-supported cooperative work

General Terms

Design, Experimentation, Human Factors.

Keywords

End User Development, Services, Integration, Mashups, Social Software.

1. INTRODUCTION

The phenomenon called "Web 2.0" [1] is strongly associated with user-generated content. Several social networking sites (SNS) came into existence, allowing users to not only create content, such as photos, videos or music. They also permit users to share content with contacts from their social networks as represented on the site and find new interesting content from others through that network.

Subsequently, McAfee coined the term "Enterprise 2.0" for the application of Web 2.0 technologies and principles to enterprises [2]. While this may also imply the use of public social network sites by companies for public relations, this paper focuses on company-internal social network sites that are deployed to improve communication and collaboration in an enterprise. More concretely, common applications are expert search and knowledge management.

A mashup is an application that combines data and other functionality from external sources, such as Web services, to create new functionality. A common example is the visualization of data from a Web service on a map. Being closely associated with the Web 2.0, mashups are also present in the Enterprise 2.0. These enterprise mashups do not only use publicly available services, but also resources that are internal to the company.

Hoyer et. al. identified a "long tail of enterprise integration" [3]: small integration projects that do not warrant a dedicated project from the IT department because they'd be too expensive and would only be of use to a small number of employees. But if those end users were able to create these integrations themselves using a mashup tool, the potential of the long tail could be exploited more effectively. In this scenario, the creation of a mashup is a form of end user development (EUD).

There are several approaches to mashup editors that are suitable for end users. Most use widgets that users may configure and then connect with each other, creating workflow-like structures. But as Halbert argues, end users not capable of or interested in programming need an environment that continuously reflects their changes in concrete data. Keeping an abstract model of the flow of a program in their minds seems to be too demanding to non-programmers [4].

A programming model that removes this burden is that of spreadsheets. All data is visible all the time, and a change in one place gets reflected immediately throughout the whole document. While for example Halbert [4], Nardi [5] and Ko [6] accept the creation of spreadsheets as a form of end user development, Jones et. al. even consider them a kind of functional programming. There have already been several successful attempts at applying the spreadsheet approach to mashup creation (e.g. [7], [8], [9]).

But as Nardi observed, actual end user programmer communities work because their tools support a layered approach that makes them accessible to users of different levels of programming knowledge [5]. She describes a spectrum that begins with end users without any programming experience and ends with professionally trained software developers. Between these extremes, Nardi identifies "local developers", or "tinkerers" – domain experts, possibly without any programming training, but with an interest in computers. These would often become helpful advisors for their non-programming colleagues.

Mashups 2010, December 1, 2010; Ayia Napa, Cyprus.

Considering these kinds of users, Nardi sees a need to support end users as well as local developers in EUD environments. In spreadsheets, formulas are accessible by all, while macros were only used by the local developers. Between all of these users, Nardi found a significant amount of collaboration, either by copying solutions by peers or by actual personal help in problem solving. Additionally, Fischer et. al. stress spontaneous and opportunistic traits they observe in EUD: users will work together when finding out they are working on similar problems. Similarly, collaborators may pull out as quickly when they consider their own problem solved [10].

This paper presents an approach to EUD of mashups incorporating these characteristics. It is a work in progress that will deploy a spreadsheet-based mashup editor in a company-internal social network site. The company's resources will be available to the mashup tool through a series of adapters. Appropriate mechanisms from social software will be used to build and support a community of practice for mashups. Figure 1 illustrates the components that would be needed for such an environment.

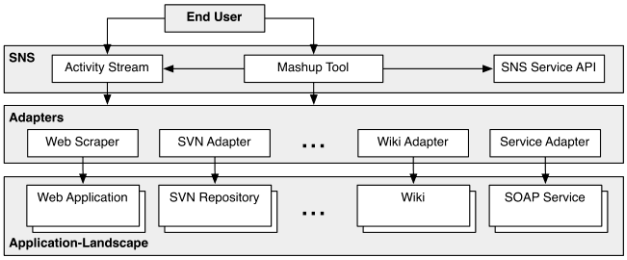

Figure 1. The components of the proposed approach.

This paper is structured as follows: the next section will explain the proposed vision in more detail. Section 3 will present related work by others, and section 4 will close with conclusions and an outlook on our future work.

2. A MASHUP SOCIAL NETWORK SITE

To create a community of practice for the creation of mashups by end users, the first mandatory element is a mashup editor. We have a prototypical implementation that recreates the user interface known from spreadsheets in a web browser. It can import XML data from HTTP URLs and displays them in nested tables reflecting the structure of the data. Some formulas are available to transform and aggregate values.

To assist in collaboration, we chose to use mechanisms from social software. For this, we developed a social network site that supports easy extension. This allows us to quickly integrate new document types – such as mashups – as well as new social mechanisms. Users have a profile, can connect to each other, may post short messages and have an activity stream that syndicates all their contacts' status messages. Documents may post messages as well, e.g. to indicate changes.

Beginning with these essential components, we are now planning to add the following mechanisms.

2.1 Adapters for Integration

To be useful for enterprise users, the mashup tool must have access not only to services available anyway, but also to as many of the company's other resources as well. This may include web applications from the intranet, ERP systems, file servers, source control systems and others. As a first exploratory step, we have created an adapter for Subversion repositories that creates a newsfeed from the latest commit messages.

But a feed-like view makes sense only for a few applications. Therefore, we are now working on a classification scheme for applications. The following list gives examples and is inspired by Rosen [11].

- Task Services provide small business functions, such as the conversion from coordinates to an address or the verification of a credit card.
- Entity Services access data sets, e.g. a customer database with associated addresses and past orders, possibly with parameters for search.
- Process Services implement potentially long-running business processes, such as customer orders.
- Feed Services provide time-based updates, such as change notifications in a repository or file system, or simple news feeds.

We plan to develop a list of service types that is more complete and, if possible, for each type provide patterns for mapping it to each of the other service types. An obvious example is the conversion of an entity service to a feed service: instead of returning all matching items, the service would only provide a list of recent changes to the database.

2.2 Extensions to the Mashup Tool

For now, our mashup tool supports only basic formulas. To provide an additional level of expressiveness for the "local developer" users identified by Nardi [5], we want to create a macro-language for mashups.

Since the tool is web-based, an API based on a subset of JavaScript seems like a good choice. This would provide access to the mashup's data and meta data as well as additional functions, such as forms and buttons for interaction. By attaching event listeners to the spreadsheet's DOM elements, even a macro recorder can be implemented.

To encourage the combination not only of services and applications, but also of mashups, the mashup editor should provide a mechanism for marking a set of cells as results. Other mashups could then import these results just like any other service's data.

Finally, a set of visualizations for the created mashups would surely be useful, as these are present in almost all spreadsheet applications and are therefore a commodity to most spreadsheet users. Additionally, the tool could provide means to export the contents of a mashup to other document types, such as text files, actual spreadsheet files or pages in a Wiki.

2.3 Social Mechanisms

Software engineering both as a practice and a science improves the creation of software. Even though social software has been in use for a few decades now, building social mechanisms into software is still one of the less controllable aspects of software development. But as can be seen in the Web 2.0 phenomenon, connecting people creates enormous advantages.

Since software development and end user development are inherently social activities, it makes sense to support these aspects with the experience that is at our disposal right now. Some initiatives are trying to create for social software what software engineering does for software in general: create processes and patterns that are repeatable and produce consistent results. To create productive mashup communities in enterprises, we want to apply these experiences to the platform we are proposing in this paper.

The "Community Lab" project has produced some such results. Rashid et. al. present a study in which users of a social network site were shown the respective values of their contributions. They compare different methods of calculating that value and their effects on participation levels [12]. Similar studies were documented by Beenen et. al. [13] and Ludford et. al. [14].

Ren et. al. examine the differences between communities based on common bonds and those based on common identity [15]. Among other things, they show how these types of communities react on the loss of members: while this weakens those based on interpersonal bonds (e.g. a circle of friends), it is less problematic for those based on a common group identity (e.g. a community of movie fans).

These and similar results are interesting and valuable, but depend on many variables that may not be controllable. We will therefore need to select a subset from existing approaches and evaluate those for their fitness in our project.

We recognize that social mechanisms that work will not necessarily have been published yet, therefore we will evaluate some of those as well: for example, an activity stream that syndicates all activities of a user's contacts. We will evaluate a low-effort mechanism for the distribution of information across a user's contacts, of which Facebook's[1] "Like" and Twitter's[2] "Retweet" features are examples. And we will compare different connection models, e.g. the synchronous (Facebook's "Friends") and the asynchronous (Twitter's "Followers") models.

Another common feature of social network sites are contact recommendations. As our goal is to support a community of end user developers, we will evaluate approaches that allow us to identify the role of users in their community – whether they are novices, regulars, or expert users.

The observation of usage patterns will also help in finding out whether those known from traditional spreadsheet software apply – if that was the case, many existing ideas that support the quality and ease of spreadsheet creation could be reused in this new environment.

The following section describes a possible scenario to illustrate the some of the components and mechanisms mentioned above.

2.4 Scenario

Mrs. Miller from the London sales department for electrical appliances sends personal letters to her best customers each month to keep in touch and to inform them about current offerings. In the past, she copied the newest sales numbers from the ERP system and matched those with customer data from the customer database. Meanwhile she has created a spreadsheet mashup that creates these matches automatically for her.

Mr. Slater, who works in the same department, is a contact of Mrs. Miller in the company-internal social network site. Because of this, his activity stream includes the activities of Mrs. Miller. Therefore, as she edits her mashup one day, Mr. Slater notices this. Even though he has no use for the mashup, he's finds it interesting. He clicks a link labeled "Interesting!" next to Mrs. Miller's activity and by this creates a new activity himself, for all his contacts to see.

Mr. Smith cares for the Birmingham customers. He is an old friend of Mr. Slater and is one of his contacts in the social network site. As he see Mr. Slater's activity – marking Mrs. Miller's mashup as interesting – he takes a look at the mashup. He immediately recognizes that he could make good use of that as well. He writes a message to Mrs. Miller, who explains to him how she keeps in touch with customers using her personalized mailings. Mr. Smith likes the idea and thus makes a copy of Mr. Miller's mashup. He customizes it for his own region.

In this fashion, the mashup, along with Mrs. Miller's mailing practice, spreads in the enterprise. Users create their own personalized copies of the mashup and get notified of changes made to the original one that they copies from. Using a version control mechanism, they can pull those changes into their own copies if they wish. One user even creates a macro that creates drafts for the mailings from a template.

After some time, the IT department notices this cluster of similar mashups from analyzes of access logs. They figure there is significant need for a proper integration of these systems and consider a new project to implement it.

3. RELATED WORK

The "EzWeb" project is a result of the "FAST" EU project. The "FAST" platform lets developers create "gadgets". End users can then combine these into mashups in the "EzWeb" platform [16]. EzWeb provides a central registry, in which users may register their mashups so they may be found later by other users searching for matching terms. The task of wrapping legacy software in services accessible by the platform seems to be completely left to the respective enterprise. In contrast, we strive to at least provide common mapping patterns, if not generalized implementations. The actual support for collaboration seems to be limited to the central mashup and gadget registries.

A similar system, the Rooftop Marketplace by SAP Research, explicitly supports social mechanisms: the roles and phases typically encountered in marketplace situations [17]. Notably, they differentiate the roles of the end user, the consultant and the developer. These seem to correspond to the roles identified by Nardi and mention in the first section of this paper. Insights gained from the project were applied to the aforementioned EzWeb project.

4. CONLUSIONS & OUTLOOK

To create a community of practice among mashup users and creators in an enterprise, we proposed a platform that is based on the user roles and their needs from end user development research. To support the role of Nardi's local developer – an end user with interest in programming, but lacking any training in it –

[1] http://facebook.com

[2] http://twitter.com

we propose adding a macro language to our browser-based mashup tool. We want to improve the collaboration amongst end users by evaluating experiences and mechanisms from community design and public social websites.

The basic elements of the proposed platform are already available to us. Currently, we are integrating the mashup editor with the social network site and evaluating existing and new integration patterns for services and application. Our next steps are the extension of the mashup tool and preliminary, informal evaluations of the platform amongst students. Once these tasks have been completed, we want to evaluate or approach in the industry.

5. REFERENCES

[1] O'Reilly, T. 2007. What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. *International Journal of Digital Economics* 65, 17-37.

[2] McAfee, A.P. 2006. Enterprise 2.0: The Dawn of Emergent Collaboration. *MIT Sloan Management Review* 47, 3, 21-28.

[3] Hoyer, V., Stanoesvka-Slabeva, K., Janner, T., and Schroth, C. 2008. Enterprise Mashups: Design Principles towards the Long Tail of User Needs. In *IEEE International Conference on Services Computing 2008 (SCC'08)*, IEEE, New York, NY, 601-602.

[4] Halbert, D. C. 1984. *Programming by Example*. Doctoral Thesis. Department of Electrical Engineering and Computer Sciences, Computer Science Division, University of California, Berkeley, CA, USA.

[5] Nardi, B.A. 1993. *A small matter of programming: perspectives on end user computing*. The MIT Press, Cambridge, MA, USA.

[6] Ko, A.J., Abraham, R., Beckwith, L., Blackwell, A., Burnett, M., Erwig, M., Scaffidi, C., Lawrance, J., Lieberman, H., Myers, B., Rosson, M.B., Rothermel, G., Shaw, M., and Wiedenbeck, S. (in press). The State of the Art in End-User Software Engineering. Accepted for publication in ACM Computing Surveys, ACM, New York, NY, USA.

[7] Tuchinda, R. 2008. *Building Mashups by Example*. Doctoral Thesis. Department of Computer Science, University of Southern California, CA, USA.

[8] Kongdenfha, W., Benatallah, B., Saint-Paul, R., and Casati, F. 2008. Spreadmash: A spreadsheet-based interactive browsing and analysis tool for data services. *Advanced Information Systems Engineering*, Springer, Heidelberg, Germany, 343-358.

[9] Wang, G., Yang, S., and Han, Y. 2009. Mashroom: end-user mashup programming using nested tables. In *Proceedings of the 18th International Conference on World Wide Web*, ACM, New York, NY, USA, 861-870.

[10] Fischer, G., Nakakoji, K., and Ye, Y. 2009. Metadesign: Guidelines for supporting domain experts in software development. *IEEE Software*, IEEE Computer Society, 37-44.

[11] Rosen, M. 2007. SOA Service Usage Types. *Business Process Trends* (online), URL: http://www.bptrends.com/deliver_file.cfm?fileType=publication&fileName=12%2D07%20SOA%20Service%20Usage%20Types%2DRosen%2Dfinal%2Epdf

[12] Rashid, A.M., Ling, K., Tassone, R.D., Resnick, P., Kraut, R., and Riedl, J. 2006. Motivating Participation by Displaying the Value of Contribution. In: *Proceedings of the SIGCHI conference on Human Factors in computing systems*, ACM, New York, NY, 955-958.

[13] Beenen, G., Ling, K., Wang, X., Chang, K., Frankowski, D., Resnick, P., and Kraut, R.E. 2004. Using Social Psychology to Motivate Contributions to Online Communities. In: *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, ACM, New York, NY, USA, 212-221.

[14] Ludford, P.J., Cosley, D., Frankowski, D., and Terveen, L. 2004. Think Different: Increasing Online Community Participation Using Uniqueness and Group Dissimilarity. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, New York, NY, USA, 631-638.

[15] Ren, Y., Kraut, R., and Kiesler, S. 2007. Applying Common Identity and Bond Theory to Design of Online Communities. In: *Organization Studies*, 28, 3, SAGE, Thousand Oaks, CA, USA, 377-408.

[16] Lizcano, D., Soriano, J., Reyes, M., and Hierro, J.J. 2008. EzWeb/FAST: Reporting on a Successful Mashup-Based Solution for Developing and Deploying Composite Applications in the Upcoming "Ubiquitous SOA". *The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, IEEE, 488-495.

[17] Hoyer, V., and Stanoevska-Slabeva, K. 2009. Towards a reference model for grassroots enterprise mashup environments. In (Newell, S.; Whitley, E.; Pouloudi, N.; Wareham, J.; Mathias-sen, L. Eds.) *Proceedings of the 17th European Conference on Information Systems*, Verona, Italy.