The Mashup Atelier

Cesare Pautasso and Monica Frisoni

Faculty of Informatics
University of Lugano (USI)
via Buffi 13, 6900 Lugano, Switzerland
cesare.pautasso@unisi.ch, monica.frisoni@lu.unisi.ch

Abstract. Can mashups be used to make high school students interested in studying computer science? To answer this question, we have designed the mashup atelier. The goal of this experimental lecture is to make students realize that the Web is not only a medium for passively consuming information but it can be actively reprogrammed as they see fit. The atelier introduces the topic of Web 2.0 Mashups to students without any formal pre-existing computer science education. After giving the atelier several times, we report on the results of a student evaluation survey showing that, if supported with right kind of mashup tools, creative students can become very productive developing interesting mashups in a short timeframe. The feedback we gathered from the students can also be used to improve existing mashup languages and tools, with the ultimate goal of understanding what makes them intuitive and fun to use.

1 Introduction

Mashup [1] development targets the so-called *long tail* of application development, enabling end-users to compose Web services and Web data sources by themselves to fulfill their own specific requirements [2]. Ideal mashup development tools should require very little upfront training and exhibit a gently-sloped learning curve [3,4]. Using such tools, end-users having very little programming experience can quickly become productive and build their own mashups [5].

To check whether mashup development is indeed a suitable target for end-user software engineering, we designed an experimental lecture called "The Mashup Atelier". In this paper we present the feedback we gathered by observing end-users as they build mashups and asking them about their experience attending the lecture. We contribute this valuable feedback to help improving existing mashup tools and languages that also target end-users without programming skills.

The mashup atelier was originally designed in the context of an ongoing national initiative to make high school students aware of the computer science discipline and to awake their interest in pursuing a computer science degree. The goal of the atelier was to use mashups to convey to the students the idea that the Web is not only an interactive medium for consuming information, but it is also a new kind of medium where they can become active participants [6]. By

G. Feuerlicht and W. Lamersdorf (Eds.): ICSOC 2008, LNCS 5472, pp. 155–165, 2009.

showing the students how to program the Web by building mashups, we hoped to unleash their creativity and inspire them to learn more about informatics and computer science.

In this paper we show that – given the right kind of tools – indeed it is possible for end-users to develop simple but interesting mashup applications without previous programming experience. We also show that mashup development can be a useful approach for attracting the interest of young students to the computer science discipline.

The rest of this paper is organized as follows. In Section 2 we define the structure of the mashup atelier and outline the questionnaire that was handed out to the students. We list some of the mashups that were developed during the atelier in Section 3. Section 4 contains the results of the survey and the feedback we gathered by observing the students. We discuss related work in Section 5 and draw some conclusions in Section 6.

2 Methodology

The mashup atelier is structured in two parts and lasts approx. 3 hours in total. During the first part, students are given some theoretical background on the inner workings of the Web and are presented with many examples of Web 2.0 services. This part concludes by introducing the notion of mashup. To do so, several example mashups are shown to students with the promise that they would be soon ready to develop similar ones.

The main part of the atelier (2 hours) is practical. Students attending it have access to a PC pre-configured with the mashup development environment. The practical mashup development work starts with a 20 minute step-by-step tutorial. Afterwards the students may work independently, first to implement a few exercises (or challenges), then to create their own mashups.

During the atelier we interacted with the students, guiding them, answering their questions and observing their progress. Thus, the students were intentionally not left alone to try to develop mashups on their own. On the contrary, students were allowed to collaborate and to share their work with one another. Our goal was not to test each student's abilities, but to create a positive working atmosphere. Towards the end of the atelier we gave the students an evaluation questionnaire (Table 1) to fill out and were able to collect answers from all participants.

The students worked with the Microsoft PopFly mashup development environment [7]. This tool was chosen due to several reasons. First of all, the tool does not require to be installed locally (apart from the requirement of having the Microsoft Silverlight runtime [8]). This way, students can start developing mashups during the atelier and then – if they used their own MSN accounts to log in – they are able to continue working on them from home. Additionally, this free tool is rather mature and stable. It provides a large library of reusable mashup building blocks, which can be explained by referring the students to the concept of function ($y = f(x)$), known to students from their high school

Table 1. Summary of the Feedback Questionnaire

1. Are you a member of a social networking site? (Yes/No) If Yes, which ones?
2. Do you know how to program? (Yes/No) If Yes, with which languages?
3. Did you already know what is a 'Mashup' before attending the atelier? (Yes/No)
4. Did you know how to use Microsoft Popfly before attending the atelier? (Yes/No)
5. What was your impression of the mashup development tool? Why?
6. Was the mashup tool intuitive? (Yes/No) Why?
7. What did you like most about the mashup tool?
8. What did you dislike most about the mashup tool?
9. Will you keep using the mashup tool in the future? (Yes/No/Maybe) Why?
10. Overall, are you satisfied about the mashup atelier? (Yes/No) Why?

math. It features a very quick design-run-test cycle and also enables users to share mashups by publishing them on their homepages. We also thought that the rich visual environment (with 3D animations) provided by Popfly would make it appealing to young high school students.

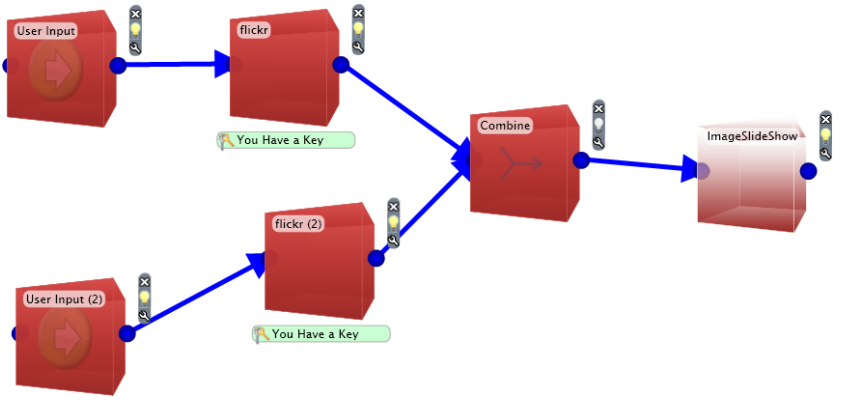
3 Mashup Examples

In this section we briefly describe some example mashups developed by the students attending the atelier. This helps to give an idea of the difficulties involved and of the complexity of the resulting mashups that were developed in the limited time available (1 hour and 30 minutes).

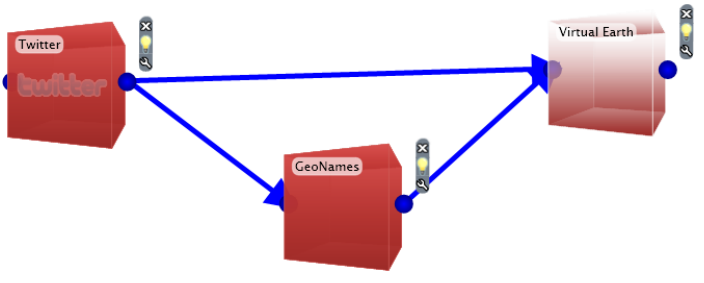
All students were initially guided through the development of a simple example mashup that would display a set of images retrieved from the flickr service. This simple example can be implemented in Popfly using two blocks: the first to search flickr for pictures and the second to display the results. Such warmup exercise helped students to understand how to configure the input parameters of a block and how to draw a connection between two blocks in order to specify data flow. During this first exercise, students were also given a walkthrough the mashup development environment so that they learned how to design, run, test, and modify their mashups.

Starting from this basic example, students were assigned the following exercises to grow the complexity of their mashup and discover more useful blocks and features of the PopFly environment:

- Display the pictures of a predefined search term on a map – this would require the mashup to fetch from flickr the geo-tagging data associated with the pictures and pass this information to the 'VirtualEarth' display block.
- Let the user enter the search terms during the mashup execution – this requires to add an additional user input block and to correctly link it with the flickr search block. Some students testing this extended version of the mashup realized that an incorrect linkage would always result in the same pictures being displayed, as the data they entered would not flow between the boxes at run-time. This shows that support for debugging is an important feature of end-user software engineering tools.



(a) Slide show with multiple sets of pictures



(b) Twittervision

Fig. 1. Mashup Examples

– Display pictures extracted from two different sites (i.e., Yahoo! Images and Flickr) – this exercise would require students to learn how to merge multiple data streams into a single one using the combine operator provided by PopFly. Students with good observation skills would then inquire on why pictures of the second set are not appended to the first and the two sets are simply interleaved by the combine block.

After working through these exercises of increasing complexity, the students had some time left for browsing the block library and creating their own mashups. Several students extended the previously described mashup collecting images from multiple sources into a mashup to retrieve and mix multiple sets of images (e.g., cats and dogs) from the same source (Figure 1a). Inspired by the twittervision.com mashup, some students were able to access a twitter feed, geo-code the location of its entries and display them on a map. To do so, they had to discover the 'Twitter' block from the standard PopFly library and learn how to correctly use it in conjunction with the 'GeoNames' block also found in the library (Figure 1b). One student was able to use the 'RSS' block to aggregate multiple technology-related newsfeeds and display them into a widget that could be embedded into his homepage. Another student successfully attempted to publish a mashup with the daily horoscope onto her facebook profile.

4 Survey Results

4.1 Background

All 43 students (29 male and 14 female) attending the four editions of the Mashup atelier held in September 2008 at the University of Lugano, in the Italian-speaking part of Switzerland, participated in the survey. Their age distribution (between 16 and 21 year-old) is shown in Figure 2. Since their native language is Italian, both the feedback questionnaire and the answers we collected were in Italian and had to be translated to be included in this paper.

All students declared their familiarity with the Web and knowledge about some common Web 2.0 services. Half of the students also used one or more social networking sites (e.g., 10 Netlog, 9 MySpace, 6 Facebook, 5 WebTI, 4 MSN). We were surprised to find out that at least five students are also running their own blog.

Figure 3 shows which programming languages (including HTML) the students have been in contact with. 26 declared not to have programming experience. If we exclude HTML, then only 9 students had previously learned at least one programming language (the most popular one being Visual Basic). We can thus identify two sub-groups of students, the ones without programming experience and the ones with some programming experience.

None of the students we interviewed knew the term 'Mashup' before the atelier. Also, no student had previously been in contact with the Microsoft PopFly mashup development environment.

4.2 Overall Impression

Figure 4 shows the number of answers to the question concerning the overall impression of the students after working with the mashup tool for two hours. Multiple answers were possible. The majority of the students found PopFly useful and interesting. Only 4 students complained about some limitations and 1 student found it useless. Only one student admitted he did not know how to evaluate the tool given he had never seen one like it.

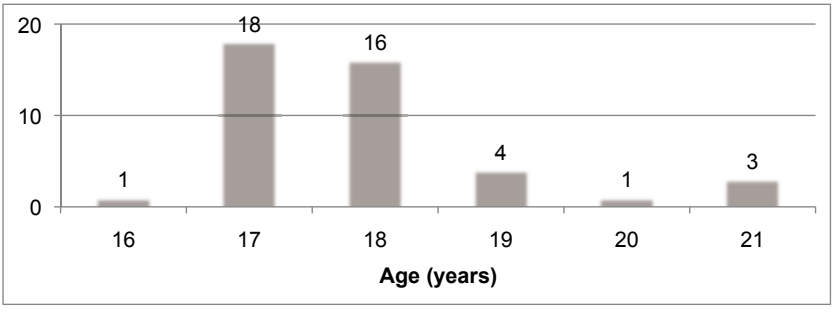


Fig. 2. Age distribution of the students attending the Mashup Atelier

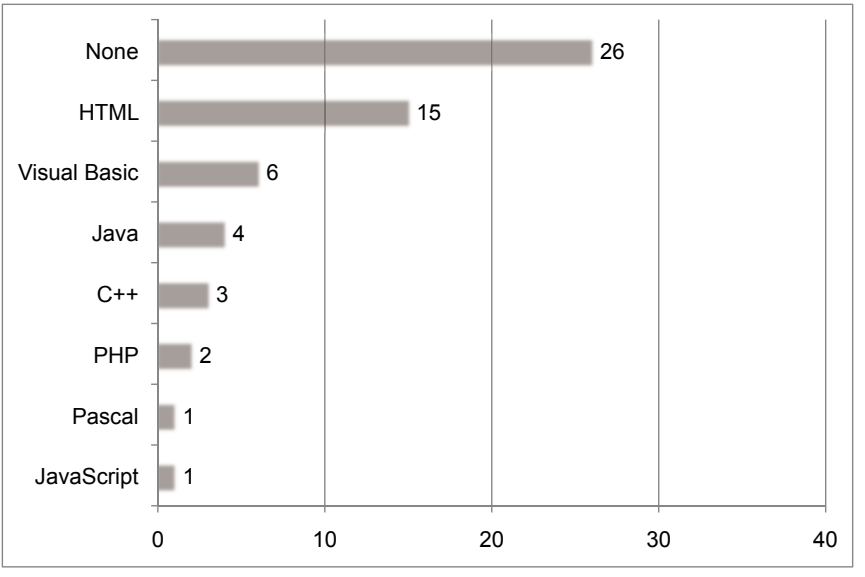


Fig. 3. Existing Experience with Programming Languages

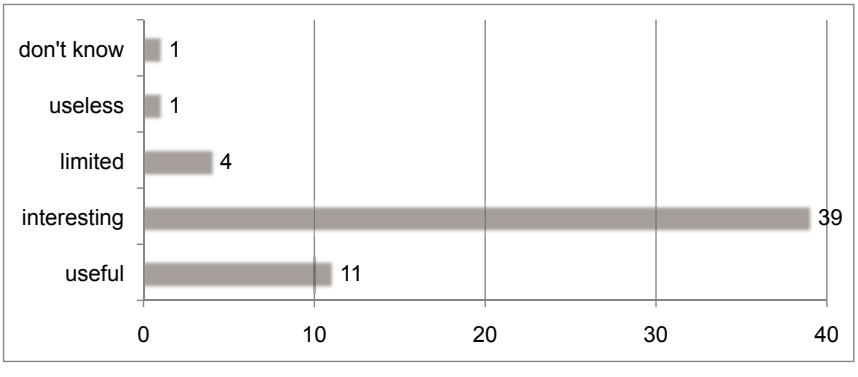


Fig. 4. What was your impression of the mashup tool?

From the positive side, the students that found the tool useful grasped its potential to "save valuable time, since all data can be put in one page". Students also found it an interesting idea to make multiple Web pages "overlap" when various data sources are combined. Many students enjoyed using their mashup to browse the flickr picture database and were also attracted by the various display blocks at their disposal. One was positively impressed because he "did not think it would have been possible to create a mashup so easily".

Some of the negative opinions were caused by glitches in the tool. One student had problems saving his homepage. One could publish the mashup but

complained that the PopFly logo is embedded into the published mashup. Another student with some previous knowledge about programming complained that the tool lacks support for interactive debugging. One student pointed out that he found the tool both interesting because of its visual block-based programming paradigm, but also limited because he would not have enough knowledge to be able to add his own personalized block to the library.

4.3 Was the Mashup Tool Intuitive?

As shown in Figure 5, the large majority (including 24 students without programming skills) of the students agreed in finding PopFly an intuitive mashup development environment.

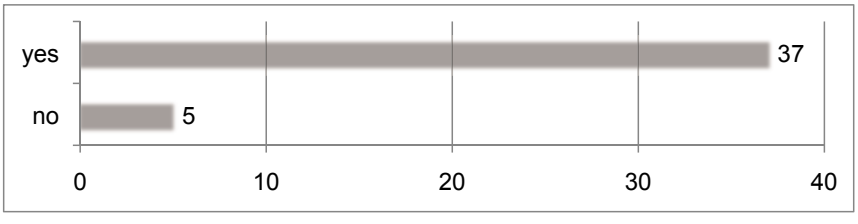


Fig. 5. Was the mashup tool intuitive?

The students giving a positive answer motivated their choice for many reasons. Some were related to the properties of the development environment. "The environment supports fast trial and error". "It is fun to use". "To work with it, is enough to try out some blocks from the library and figure out how to connect them". "Blocks have tooltips with descriptions of their functionality". "Once you understand how to connect the boxes, it is easy". "Even if I am not a computer expert, I could more or less finish all of the exercises". Other students also commented on the language itself. "It does not use a technical language". "The language uses symbols rather than words". "The language basic connection mechanism is not too complex and anyone can use it thanks to its clear graphical notation". "It works even if you have never seen a programming language". "The graphics look visually simple and the available commands are not too complicated." "Even if it is in English, it is rather understandable and simple".

The 5 negative answers were explained as follows. One student admitted that he "needed to ask the teacher for help", despite his knowledge of Pascal and Visual Basic. Two students with knowledge of HTML recognized that the tool "requires good computer science skills" and "it is not very interactive". The two students without programming knowledge stated that "It was the first time I used it", and "It looks rather complicated".

4.4 Positive Feedback

In general, the students liked learning about the notion of mashup, and the idea of creatively mixing together information coming from various Web pages.

The students were enthusiastic to have a sort of control to some known and powerful Web applications, such as "Google Maps" or "Virtual Earth", and to easily have access to pictures from "Flickr" and "Yahoo! Images". The idea that they could embed them into their personal web page, and show to their friends what they could create, made them appreciate the mashup philosophy and let them discover an alternative way to use the Web.

The students liked the following specific features of the mashup development tool. On the one hand, they appreciated how quickly they could play with the blocks of the library, experiment with their mashups and build visually appealing slide shows. On the other hand, a few students also liked the rich graphical notation and the search feature of the library browser tool. The ability of easily gaining control over a certain Web data source or display widget was found very positive by many students. They enjoyed searching for pictures and displaying them using different kinds of visualizations. They were fascinated by the fact that data and images coming from different well-known Web services could be merged together into a new mashup application designed by them.

Some students were also very excited about the possibility of sharing their mashup and publishing them on their homepage. After learning about this feature, one student quickly invited all of her friends present in the classroom to join her new PopFly social network and forced them to "become a fan" of her mashup.

4.5 Negative Feedback

23 students also gave some constructive criticism from their experience with the mashup development tool. Only 2 students wished the tool was available in the Italian language.

The block library attracted many suggestions for improvement, showing that it is one of the most important features of a mashup development tool. Some students felt overwhelmed and confused by the size of the library with hundreds of reusable building blocks. Others were satisfied of having so much choice at their disposal. Thus, whereas it is hard to argue in favour of a smaller or a larger block library, its accessibility becomes very important. To help them getting started, we gave them a list of 10 useful blocks and suggested they would ignore the user-contributed blocks (since they may not always work). The automatic suggestion feature was also used by the students to filter the block library and make it display only blocks that could somehow be connected to the current one. The results of this feature were not always clearly understood, as they did not distinguish the blocks that could be used to provide input from the ones that could be used to consume the output. Also, some pointed out that the descriptions associated with the blocks and their configuration parameters were not always easy to understand.

After becoming familiar with the block library, many students realized that the set of mashups they could build is limited by the available blocks they could find and understand how to use. Thus, they would not be able to build a mashup unless they could find a suitable set of blocks to implement the required

functionality. Thus, PopFly supports very well a bottom-up composition scenario where users are guided building mashups starting from existing blocks. However, it only offers partial support for top-down design and decomposition. This became evident to some students, since they realized that they would not be able to implement their own blocks to supply the missing functionality (For example, to scrape information out of a Web page). This is due both to their lack of JavaScript programming skills, but also to the lack of documentation of the PopFly block development API. The students that dared to switch to the advanced configuration view of a block were simply lost facing the low-level code implementing the block.

Some students were also annoyed by the requirement of obtaining and entering a registration key, as this would break their creative exploration flow. Thus, it would be convenient to provide pre-registered blocks for demo purposes that can be later on configured with the proper credentials. Four students also complained about the limitation concerning the maximum number of display blocks that could be added to their mashup. Others were not successful in using the display widgets to show their own pictures or could not find a block to play music.

Regarding the usability of the graphical notation, several students – at first – were confused by the positioning of the input/output connectors of a block. Without an explicit explanation they did not independently grasp the convention of using the connector on the right-hand side to represent the output and the one on the left to represent the input of the block. Whereas this is a common notational convention of many visual languages, for complete beginners it remains an arbitrary convention that needs to be explained. Once this was clarified, drawing arrows between boxes simply required them to learn the correct mouse click/drag sequence.

Another limitation pointed out by the more technically savvy students is related to the requirement for having the Silverlight extensions installed both for developing mashups but also for displaying them once they are shared and published on their own hompage. Also, they were disappointed that once their mashup was embedded into their personal Web page, they would not be able to hide the Microsoft PopFly logo.

4.6 Will You Keep Using PopFly in the Future?

Also in this case, as summarized in Figure 6, the majority of students answered that they would be interested in further usage of PopFly at home.

The positive replies were motivated because the students were curious to find out more about the tool's potential applications. They found it interesting, useful, and even "cute and fun". Two students emphasized the possibility of embedding mashups into their own web pages and they planned to use the tool to build interactive photo albums, only if they could find out how to build mashups feeding them their own pictures and personal data.

The uncertain replies were due to some difficulties that were encountered during the atelier, lack of time to explore the tool in depth, and a vague: "you never know if it will become useful someday".

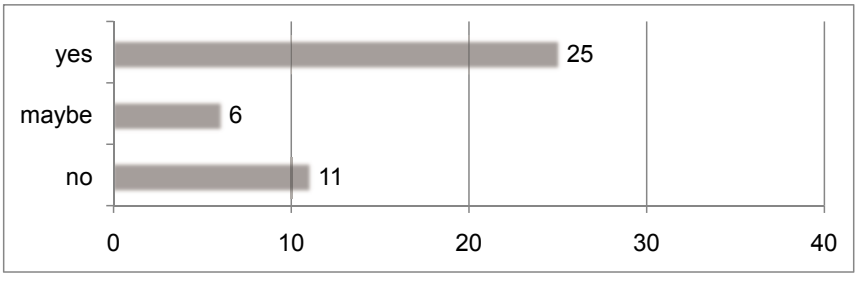


Fig. 6. Will you keep using the mashup tool in the future?

The negative replies did not give much insight on how the tool could be improved, as this group of 11 students seemed to be satisfied with browsing existing Web sites and participating in existing social networking tools. They were not interested in further exploration and did not find an immediate need for the tool. One stated that: "he does not normally use the computer for this kind of things".

5 Related Work

The potential for using mashups for educational purposes was first discussed in [5]. In the same context, [9] is a more recent paper documenting the usage of mashup development tools with teenagers. A comprehensive survey on end-user programming languages and environments can be found in [4].

Whereas most existing research on mashup development environments shares the goal of making it fast and easy to serendipitously reuse [10] and compose existing Web services and Web data sources into a mashup, only a few contributions explicitly target inexperienced end-users. For example, the Mashmaker tool [11], is presented as a user-friendly mashup environment based on a functional programming language. Also, Marmite [12] addresses the needs of end-users with its highly interactive development environment. We did not considering using these tools in the mashup atelier due to their limited availability.

6 Conclusion

This paper reports on our initial experiences with the Mashup Atelier, an interactive, project-based lecture for introducing young high school students with the topic of Web 2.0 Mashups. This experimental atelier was designed with three main objectives. The first was about testing whether young students with no programming experience are capable of building mashups within a short timeframe. The second was to gather feedback from the student impressions to benefit future generations of "intuitive" mashup development environments and languages. The third was to see if mashup development could be effectively used to get young students interested in learning more about computer science. Overall,

the mashup atelier received very positive feedback and a high degree of student satisfaction (41 out of 43 students were satisfied with it, while the remaining 2 did not answer the question). It is still early to measure how many of the high school students who attended will choose to enroll in a computer science degree program. Nevertheless, from the results of our survey we observe that exposing students to mashup development using a visual and interactive tool such as Microsoft PopFly was very useful to get their attention.

Acknowledgements

This work is partially supported by the Informatica08 initiative. The authors would also like to thank Mauro Prevostini for his excellent logistical support with the atelier organization and Monica Landoni for her help in improving an early draft of the student feedback questionnaire.

References

1. Wikipedia: Mashup (web application hybrid), `http://en.wikipedia.org/wiki/Mashup_web_application_hybrid`
2. Hoyer, V., Stanoesvka-Slabeva, K., Janner, T., Schroth, C.: Enterprise mashups: Design principles towards the long tail of user needs. In: Proc. of IEEE International Conference on Services Computing (SCC 2008), pp. 601–602 (July 2008)
3. Rode, J., Bhardwaj, Y., Pérez-Quiñones, M.A., Rosson, M.B., Howarth, J.: As easy as "Click": End-user web engineering. In: Lowe, D.G., Gaedke, M. (eds.) ICWE 2005. LNCS, vol. 3579, pp. 478–488. Springer, Heidelberg (2005)
4. Kelleher, C., Pausch, R.: Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. ACM Comput. Surv. 37(2), 83–137 (2005)
5. Lamb, B.: Dr. Mashup or, Why Educators Should Learn to Stop Worrying and Love the Remix. EDUCAUSE Review, 7 (2004)
6. Jhingran, A.: Enterprise information mashups: integrating information, simply. In: VLDB 2006: Proceedings of the 32nd international conference on Very large data bases, Seoul, Korea, pp. 3–4 (2006)
7. Microsoft: Popfly, `http://www.popfly.ms/`
8. Microsoft: Silverlight, `http://www.microsoft.com/silverlight/`
9. Yardi, S.: From functional to fun: End user development for teenagers. In: Proc. of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2007), pp. 272–274 (September 2007)
10. Vinoski, S.: Serendipitous reuse. IEEE Internet Computing 12(1), 84–87 (2008)
11. Ennals, R., Gay, D.: User-friendly functional programming for web mashups. In: ICFP 2007: ACM SIGPLAN International Conference on Functional Programming - SIGPLAN Not., vol. 42(9), pp. 223–234 (October 2007)
12. Wong, J., Hong, J.: Marmite: end-user programming for the web. In: CHI 2006 extended abstracts on Human factors in computing systems, Montreal, Quebec, Canada, pp. 1541–1546 (2006)