PlayByPlay: Collaborative Web Browsing for Desktop and Mobile Devices

Heather Wiltse
Indiana University School of Informatics
901 East 10th Street
Bloomington, IN 47408
hwiltse@indiana.edu

Jeffrey Nichols
IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120
jwnichols@us.ibm.com

ABSTRACT
Collaborative web browsing tasks occur frequently, such as one user showing another how to use a web site, several users working together on a search task, or even one user sending an interesting link to another user. Unfortunately, tools for browsing the web are commonly designed for a single user. *PlayByPlay* is a general-purpose web collaboration tool that uses the communication model of instant messaging to support a variety of collaborative browsing tasks. PlayByPlay also supports collaborative browsing between mobile and desktop users, which we believe is useful for on-the-go scenarios. We conducted user studies of the desktop and mobile versions of PlayByPlay and found the system to be usable and effective.

Author Keywords
Collaboration, web, collaborative browsing, co-browsing, mobile, design, CoScripter

ACM Classification Keywords
H.4.m. Information systems applications: Miscellaneous.

INTRODUCTION
Browsing the web, as it is currently designed, is an inherently individual task. The traditional setup of one display and one set of input devices does not lend itself to use by more than one person at a time. However, situations in which there is a need to collaborate on browsing tasks are common. In a desktop context, these scenarios include planning a trip, researching a topic, showing how to perform a task, etc. Common methods used in a desktop setting include sending links back and forth via email or instant messaging; using remote desktop software; or simply looking over someone's shoulder if the users are co-located.

In a mobile context, collaborative browsing between a desktop user and a mobile user can be advantageous anytime the complexity of the task outweighs the resources of the mobile user. A common scenario involves a mobile user calling a friend whom she expects to be near a computer and asking that friend to perform a browsing task as her proxy. These browsing tasks might range from getting directions to a location to checking an e-mail for a piece of critical information. With current technology, this is largely a one-sided collaboration carried out entirely over a voice channel. The mobile user must give instructions and the desktop user must describe what they are seeing over this voice channel, which leads to errors and frustration. The mobile user may also be required to either remember or write down complex information, such as directions or phone numbers. Even in this age of high-functionality mobile devices, these collaboration scenarios remain valid when the mobile user is walking or driving, or when the browsing task to be performed requires a substantial amount of text entry.

To address these issues, we have created *PlayByPlay* (PBP), a system that allows for collaborative browsing through an instant messaging channel (see Figure 1). Users initiate a collaborative browsing session by opening a chat session, as they would in any instant messaging tool. As the users interact with their web browsers, their interactions are recorded as natural language descriptions and sent to the other users via the chat channel. For example, if a user named Bob clicked on a "Search" button, then "Bob clicked on the 'Search' button" would be sent in the chat stream. These actions can be executed by other user(s), either manually or automatically at each user's discretion. This aspect of PBP is implemented through use of CoScripter technology [9].

An additional important aspect of PBP is the ability to clip and send snippets of web content. These web snippets are integrated into the instant messaging chat stream. Snippets assist the collaborative browsing process in two ways: 1) they allow users to send explicit awareness cues to other users that show not only the current page being viewed, but also the portion of the current page that the sending user thinks is most relevant; and 2) snippets provide a substrate on which users receiving the snippet can interact. This latter functionality is especially important when only one of the users can access the web

CHI 2009, April 4–9, 2009, Boston, Massachusetts, USA.

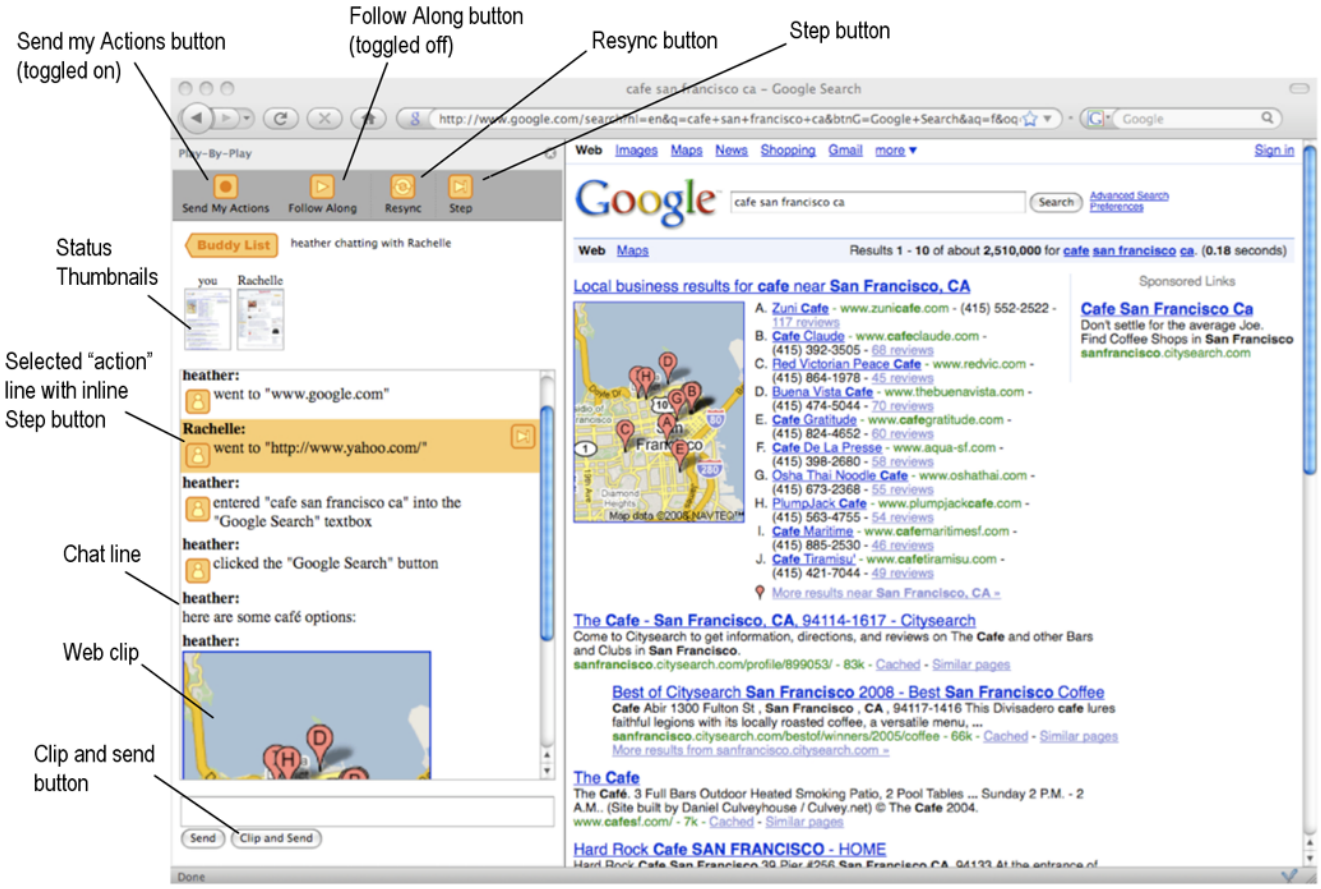

Figure 1. The PlayByPlay user interface embedded as a sidebar in the Mozilla Firefox web browser.

site or web session that is the subject of the collaboration. For example, if one user is purchasing a product from an e-commerce web site, the session associated with the checkout process can typically only be accessed on one computer. By clipping a piece of the page that the user is having trouble with, other users can see the problem and demonstrate a solution.

PBP can be accessed via either a desktop computer or mobile device, thus supporting on-the-go as well as stationary collaborative browsing. We have used the system in a wide range of collaborative browsing scenarios, which we describe later in the paper.

RELATED WORK

Our work draws on two main areas of research: collaborative browsing and searching, and mobile web.

Collaborative Browsing and Searching

A number of systems have been designed to support collaborative web browsing or searching. Many of these are highly synchronous, with all participants viewing the display of one machine and sharing access to the keyboard and mouse pointer. One of the most common is Microsoft's Remote Desktop [10], which allows a user to control a computer remotely. In a collaborative setting, one user can take the lead and control the display that her collaborators watch. Other approaches have been designed more specifically for collaborative web work. Cabri, Leonardi and Zambonelli [2], for example, used a proxy server to distribute web sessions and histories to users in a group. Web pages visited by a member of the group were stored in a cache, and links were color-coded to provide social indicators, such as whether or not another user had visited the link already. Their implementation provided "master-slave" functionality, in which a "super user" controls a session and the "slaves" are forced to follow along (unless they quit the entire session). Other master-slave approaches include GroupWeb [4] and CoWeb [8]. In contrast, PBP's method of sharing web page activity through chat allows each user to decide whether to follow along or do something different, and PBP also makes it very easy to switch between modes.

Other work has focused specifically on improving users' abilities to collaborate on web search tasks. A common overall approach is to allow users to use a divide-and-conquer strategy through splitting up web pages. WebSplitter [5], for example, splits a web page among multiple devices. This divide-and-conquer approach can be replicated in PBP if users choose to each perform their own browsing tasks and share clips from different web

pages. In a somewhat different approach, the CoSearch system [1] supports co-located collaborative search through a shared display and multiple input devices. PBP, on the other hand, is designed primarily for non-co-located scenarios.

SearchTogether [12] takes a more comprehensive approach in supporting both synchronous and asynchronous collaborative web searching. It provides awareness of other users' search terms and history of visited pages, mechanisms for division of labor in searching, and shared storage and annotation of search sessions. We see PBP as a general purpose collaborative browsing technology, and thus we have not implemented special purpose search features such as these. However, such features could be added to PBP in the future.

Existing collaboration methods can be seen along a gradient from completely asynchronous, such as sending links via email, to completely synchronous, such as Remote Desktop. Neither of these extremes is optimal, as they are both inefficient: one provides too little coordination (completely asynchronous), while the other provides too much control for only one individual (completely synchronous). With the exception of SearchTogether [12], most existing approaches support only one mode and consequently support very specific scenarios and interaction styles. PBP is designed to allow for fluid transitions between modes. We believe our design is ideal for supporting many collaboration situations which generally require some degree of *both* close coordination *and* user freedom at different times.

SearchBar [11] assists users with their searching tasks, including associating past searches with notes and other metadata. The association of web history with language is similar in principle to PBP's means of sharing web activity through chat; however, in PBP the language is generated by the system rather than by the user. PBP is also not limited to assisting with search-centric tasks.

Mobile Web

Designing for situations in which people are on-the-go entails an additional set of challenges. Since it is more difficult to access and use the web on a mobile device than on a desktop computer, the mobile web lends itself to addressing needs for information in situations that are important, urgent, and cannot be addressed any other way. Hinman, Spasojevic and Isomursu [6] refer to these as "mobile moments," and suggest that mobile applications that can support these can be highly valuable for users. This was one of their findings from doing a contextual inquiry deprivation study, in which participants were required to use only their mobile phones to access the internet for four days (with some exceptions, such as filling out an online diary entry as part of the study and thirty minutes of other web activity). Another finding the authors mention is that accessing web pages optimized for desktop computers is a challenge; people need a simpler structure, with just the information they need when using the mobile web.

Similarly, Cui and Roto [3] conducted a contextual inquiry investigation in six cities over three years in which they looked at how people use the mobile web. One of the dimensions of their resulting mobile web activity taxonomy that they developed in their analysis is *information seeking*. They looked at the types of information needs that participants had in mobile situations, what prompted the needs, and how they went about addressing them. For those needs that were addressed at the time they arose (as opposed to later), 30% were addressed using the web. Interestingly, 16% were addressed by calling a proxy – someone the participant believed would be in front of a computer and could more easily locate the required information.

We consider this scenario of calling a proxy when on-the-go to be a form of collaborative mobile browsing. PBP is the first tool that we are aware of to specifically support this interaction. Without PBP, users are forced to perform all of the collaboration verbally and send textual information, such as links, through established channels, such as e-mail or SMS. Remote desktop software does exist for some mobile devices (e.g. VNC for Windows Mobile devices [14]); however, configuration and setup of this software can be difficult, and significant differences in screen size make navigating around the desktop computer cumbersome from a mobile device.

More software support exists for users to "collaborate" with themselves by sending content to their mobile devices from their desktop computers. Most of these tools do not currently support multiple users, although they could be extended to do so in the future. The Joey project [13] developed by Mozilla Labs for the Firefox browser is one example of such a system. Users could use Joey to save pieces of web content to their accounts for later mobile access.

MOBILE/DESKTOP WEB COLLABORATION SURVEY

We conducted an informal survey in order to get a sense for how frequently users engage in mobile collaborative browsing, as well as to get information on other mobile web and collaborative browsing behavior. In addition to basic demographic and web usage questions, we asked participants specifically about the types of situations where they might be on-the-go and call someone else for information.

We recruited participants by sending an e-mail to other employees in the second author's research department. We also posted links to the survey on several internal blogs. Participants received no compensation for filling out the survey. We had a total of 23 participants; 19 were male, and over half were in the 20-35 age bracket. About 65% of participants ranked their expertise in using the web to get information as 'expert'.

Seven of our respondents reported using the mobile web, and all seven use either a smartphone or an iPhone. Interestingly, six of these also reported having called a proxy, suggesting that there is a use case for mobile/desktop collaboration even for mobile users who have high functionality devices. When asked about why they use the mobile web, respondents reported both high- and low-urgency activities (e.g., getting directions or avoiding boredom). This was an interesting insight, with the implication that a design that can support both time-critical and purely leisure activities could be beneficial.

In thinking about scenarios in which an on-the-go user calls someone else for help, we hypothesized that these situations would generally be ones in which there is a fairly urgent and important information need, and that the proxy that the user would call would most likely be a member of one's close social network (e.g., spouse, significant other, close friend, etc.). To address these issues, one set of questions asked participants to indicate the likelihood that they would call someone with regular web access while on the go in different scenarios. There was a high importance/urgency scenario (finding out the status of a flight one is taking in a couple hours); a medium importance/urgency scenario (finding out what time the local post office closes to see if one can make it in time to mail a package); and a low importance/urgency scenario (finding out the answer to a trivia question to settle a dispute with friends). Results were in keeping with the hypothesis that people are more likely to call a proxy when the information need is important and urgent; over 60% of respondents indicated they would be either 'likely' or 'highly likely' to call someone with regular internet access in the flight status scenario, compared with about 56% and about 47% in the moderate and low importance/urgency scenarios respectively.

We also asked respondents if they had contacted someone with regular web access for information while on-the-go, and if they had ever been contacted themselves to act as a proxy for another person. About 81% of respondents said they had called someone while on-the-go, and about 86% said they had been called by someone else. When asked about their relationship to the person in both of these cases, the overwhelming majority indicated that it was someone who is a member of their close social network: spouse, significant other, friend, or family member. In more than half the cases (across both situations of contacting a proxy or being a proxy for someone else), the information need was related to an address or directions. Phone numbers were another common need.

One respondent related an interesting scenario in response to the question about being contacted by someone else to act as a proxy. This respondent reported being contacted by someone who was shopping for a wedding dress. This person asked the respondent to go to a competitor's website to check the price of a dress, and had to explain how to navigate the website to find it.

Guessing that the users of a collaborative browsing system in these types of scenarios would generally be close friends and family, we also wondered if such a system would be used for purely social (i.e., not goal-directed) applications. This is analogous to someone sharing web links with members of her close social network just because she thinks they will enjoy them. So we also asked respondents how frequently they send others links to web content in this type of scenario, and also how often others send them links. Sixty-five percent of respondents said they send others links either 'often' or 'very often', while 80% indicated that others send them links to sites either 'often' or 'very often'. This supports our belief that social sharing of web content is fairly common.

While not formal, the survey suggests that people do engage in mobile/desktop collaborative browsing and gave us some interesting qualitative data to help motivate our design.

ARCHITECTURE

There are several important aspects to the design of PlayByPlay:

- All communication between clients uses an instant messaging (IM)-style, which allows new users to understand the system in terms of a familiar mental model.
- All messages, even those generated by the computer, are sent in human-readable language. This has several benefits: a user can read along with what another user is doing without actually performing the interactions; a user can double-check that the actions that will be performed are correct before executing them; and the user can re-visit the chat log at a later time and still interpret, and even re-execute, the steps that were taken previously.
- Distinct browser sessions are used at all end points, unlike previous systems, such as GroupWeb [4], which have shared a single browser session across all users. An advantage of our approach is that users are always free to stop sharing their web actions and go off on their own. There are other pros and cons of this decision, which we will discuss later.
- Pieces of the current web page can be clipped by a user. These clips are sent to the other user(s) as both an image and HTML/CSS. The images provide the receiving user with additional awareness of what the sending user is doing. This use of images is similar to that of Smart Bookmarks [7], though PBP's images are generated as result of a user selection rather than by the system. The HTML/CSS clip can be opened and used as a proxy substrate for viewing the actions of the sending user or demonstrating actions for the sending user to apply on the page from which the clip originated.

While we were most interested in supporting collaboration between mobile and desktop devices, we recognized that there are many relevant scenarios that our system could address for collaboration between desktop users. We also noted that a capable desktop client would be important for the mobile scenarios, where the desktop user will typically be performing most of the web interaction. Therefore we chose to build a system that would be useful for both desktop/desktop and desktop/mobile collaboration.

Implementation

The features of PBP could, in principle, be implemented on top of most existing IM infrastructures, such as AIM, MSN Messenger, etc. The current version of PBP uses our own web-based IM implementation, which supports most of the features of any common IM system. For example, each user in our system has a buddy list and users can start conversations with other users at any time. We chose to build our own IM system for two reasons: 1) we wanted to have the flexibility to add new features to the IM stream without being restricted by the design of an existing system, and 2) we believed that a web-based IM system would be easier to port to mobile devices, such as the iPhone, where highly capable web browsers are already available. In the future, we plan to build a new version of PBP on top of a popular IM framework.

We have built three clients for PBP using our web-based IM framework. The main desktop client is implemented as an extension to the Mozilla Firefox browser (see Figure 1). We have also built a web-only version and an iPhone-specific version. These latter two operate entirely within web pages and have limited functionality due to security restrictions placed on JavaScript code within a web page. Specifically, it is not possible for JavaScript originating from one server to interact with a web page originating from a different server. As a result, in the web-only and iPhone versions of PBP it is currently not possible to send web clips and web actions can only be recorded and executed within clips that are opened from the chat stream. In the future, we may be able to increase the capabilities of the iPhone version by creating a native application using the iPhone SDK. Creating such a native application would require heavy integration with the existing Safari web browser, however, and this does not seem to be possible with the current version of the iPhone SDK.

A key feature of PBP is the use of natural language descriptions of web activity, which may be executed from the chat log at any time. This feature is implemented using the CoScripter system [9], one of the first systems to employ this type of representation. Other systems, such as Smart Bookmarks [7], have used similar natural language representations.

Our current web clipping algorithm is implemented in Firefox. Clips are recorded in two formats: as an image using Firefox's canvas API, and as HTML extracted from Firefox's internal DOM tree representation for web pages. We found that both of these representations are necessary, because the HTML of a clip may look substantially different than the original clip if it is rendered out of context from the original page. The change in appearance is often due to the lack of style sheet information that exists outside of the clip. We have explored more robust means of clipping style information along with clips, but have not yet found a satisfactory solution.

USING PLAYBYPLAY

Figure 1 shows the desktop PBP interface within a sidebar in the Mozilla Firefox web browser. The overall design is similar to that of most IM clients, with a scrolling chat log above a text field for entering new messages.

At the top of the PBP sidebar is a toolbar with four buttons: *Send My Actions*, *Follow Along*, *Resync*, and *Step*. The first two of these buttons are toggles that control the level of interaction between the chat stream and the web browser. When a user toggles the *Send My Actions* button on (as shown in Figure 1), her web activity is automatically turned into human-readable descriptions and sent as messages in the chat stream. If the *Follow Along* button is toggled on (shown in the off state in Figure 1), then PBP will attempt to execute all recorded web actions received in the chat stream from other users. If the user does not want to execute these web actions automatically, she can select a chat line corresponding to web action and press the *Step* button to execute that step manually. An additional step button is also provided in the chat line for the user's convenience.

The *Resync* button is an experimental feature that allows a user to attempt to re-synchronize with another user based on the actions recorded in the chat stream. Our technique is patterned on an algorithm used by Smart Bookmarks [7]. When this button is pressed, PBP reads through the chat stream, finds the most recent action of the form "user went to 'http://somewebsite.com'," and then executes this action and any subsequent actions from the same user. If the remote user has been sharing all of the web activity since visiting that url, then the resync will be successful. Otherwise, it will likely fail. Using the Resync button is also potentially dangerous if the other user's actions include performing some transaction, such as buying an item on an e-commerce web site. This feature is currently included in PBP because it was requested by several users in our user study; however, more work is necessary to determine if it is useful enough to override concerns about confusion and undesired actions it may cause.

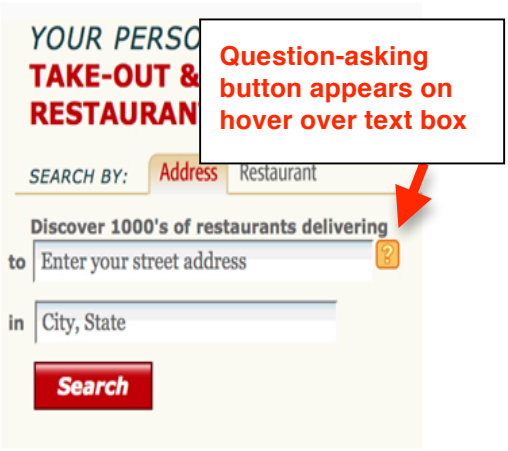

Figure 2. Question-asking interface within a web page.

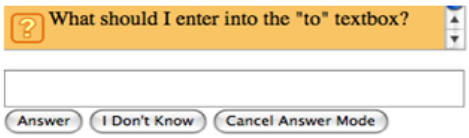

Figure 3. Question-answering interface in the PBP chat client.

Above the chat log in the PBP interface are the *status thumbnails*. These thumbnail images show each user's current page. Mousing over a thumbnail shows a larger image of the page and the URL of that page in the browser's status bar. Clicking on one's own image toggles broadcasting an image of the current page, and clicking on an image from another user opens the URL on that page in one's own browser. The thumbnails provide awareness not only of other users' locations, but also of when they navigate to a new page.

At the bottom of the PBP sidebar is the *Clip and Send* button. Pressing this button allows a user to select a portion of the current web page and send it to the other user(s). Clips appear as images in the chat stream, and double-clicking on a clip in the chat stream causes the HTML version of the clip to be opened in a new tab in the web browser. A clip of a map can be seen in Figure 1 near the bottom of the chat stream.

Finally, there is an interface to facilitate asking another person a question about what to enter into a form element, including text fields, checkboxes, and radio buttons. Hovering over a form element makes a question mark button appear (as shown in Figure 2), and clicking that button generates the appropriate question about what to enter in that particular box that is sent to others in the chat session. For example, using this feature on a text box for an address would generate a question such as: "What should I enter in the address text box?" The receiver of a question is then switched into question mode (as shown in Figure 3), so that whatever he types and sends is entered as the text for the field in question on the sender's side. If the target form element is a check box or radio button, then the system uses a simple dictionary mapping to map the receiver's response to a yes or no value and takes the appropriate action. If the receiver does not know the answer to the question, he can simply type "I don't know" or one of several other similar phrases, or he can click the *I Don't Know* button. If he does not want to respond, he can click the *Cancel Answer Mode* button to go back to the regular chatting mode.

Passwords are an important special case for PBP. Fortunately, most password fields are marked as such in their HTML documents and are thus easy for PBP to recognize. When the user types in a password with *Send My Actions* turned on, PBP will send a message of the form: "Bob has entered a password into the 'Password' text field." If the user would rather send the value of his password, he can send it in the chat directly or use the special question/answer feature.

PlayByPlay on the iPhone

The PBP interface for the iPhone is accessed as a web page through the Mobile Safari web browser, either by entering the server URL, opening a bookmark, or tapping a link sent in a text message to the iPhone by the desktop user. The login and buddy list pages look very much like pages from a standard iPhone application and the main chat page looks very much like the sidebar in the Firefox version of PBP (see a). Missing from the iPhone interface is the toolbar with the *Send My Actions*, *Follow Along*, and other buttons, which do not work in this version because it is embedded within a web page and cannot control other web pages. We also removed the status thumbnails because of screen space constraints and the lack of a mouse over event on the iPhone's touch screen interface.

The most important feature of the iPhone interface is its special capability for supporting recording and playback of web actions on clips. When the user clicks on a clip in the chat stream, the clip is opened in a new window within the Mobile Safari browser (see b). This clip window contains the HTML version of the clip and three buttons across the top of the page. The *Send My Actions* and *Follow Along* buttons have the same function as those in the Firefox sidebar. The design of the Mobile Safari browser and the screen size on the iPhone prevents more than one window from being visible at a time, which prevents the user from seeing new chat messages while viewing a clip; however, it is very easy to switch between windows in the Safari browser.

Despite its limitations compared to the desktop version, there is reason to believe the iPhone version of PBP is effective at supporting scenarios in which a desktop user acts as a proxy for a mobile user. In proxy scenarios, the desktop user performs most or all of the interactions. The mobile user can follow these interactions at a high level by watching the chat log. When more detailed awareness is needed, the mobile user can ask the desktop user to clip the section of the page that is being used. By opening the

Figure 4. PlayByPlay iPhone interfaces. a) The main chat window. b) A view of the HTML of the clip shown in the main chat window.

clip, the mobile user can observe the desktop user's interactions happening in the context of the page. In the few situations where the mobile user needs to supply information back to the desktop user, he can do so by interacting with a clip, by responding to a question about what to enter into a particular form element, or by communicating with the desktop user through another channel such as a voice phone call.

USAGE SCENARIOS

The utility and features of PBP can be better illustrated by considering a few possible usage scenarios.

Scenario 1: Teaching a Process

Suppose that John is a new employee at a large corporation, and he is trying to learn how to navigate the company's intranet pages to perform necessary administrative tasks. One of these tasks is ordering business cards, and he asks Jane, a more senior employee, for help.

Jane asks John to start a PBP session with her. She then demonstrates how to order business cards by toggling on the sending of her actions and going through the necessary steps. John follows along and executes each step. When Jane reaches a page that requires authentication, the system detects the password field and sends a message stating that Jane has entered her password, rather than the actual text as it would for other text fields. When John reaches this point he enters his own password. He can then execute *in his own account* the actions that Jane performs in *her account,* since the structure of the pages is the same. When Jane has finished demonstrating the process, John renames the session to "Ordering Business Cards" and keeps it in his PBP account for later reference.

Scenario 2: Awareness from Mobile Device

Hank is driving to a new restaurant to meet a friend, but he forgets to write down the address and only remembers the restaurant's unusual name. He calls his friend Liz, who happens to be sitting in front of her computer, and asks her to search for the place and send him directions. She first goes to Google, clips the search page, and sends it to Hank. She then types in the restaurant name as she heard Hank spell it out. Hank glances at that clip on his iPhone and sees that Liz has made a slight misspelling in the name. He corrects her over the phone, and she then performs the search using the correct name. Once Liz has found the address, she types it into Google Maps and gets a link to a map for the location. She sends this link to Hank. Because Hank is using an iPhone, he can tap on this link in his browser, perhaps while stopped at a traffic light. This will automatically open the built-in Google Maps application to the linked map and he can easily get directions to the restaurant from his current location with just two additional finger taps.

Scenario 3: Quick Information & Verification Requests

Joe is going to a party at Dave's house in about an hour. He is in charge of ordering food to be delivered, but he needs Dave's help for some parts of the order. Joe goes to grubhub.com, where he will enter the order. He doesn't know Dave's exact address, so he uses the question-asking feature to ask Dave, who happens to be online (see Figure 2). This puts Dave's interface into question-answering mode, so all he has to do is type his address and send the message (see Figure 3). The address Dave entered as his answer is automatically filled into the text box on Joe's end, so Joe then proceeds to the next step.

Joe also wants Dave to look at the order before he submits it to make sure that he has ordered food that the people attending will like, and enough of it. So he sends a clip of his order right before he submits it, which Dave confirms to be correct.

Scenario 4: Planning a Trip

Two friends are both going to be in New York on business, and they want to find a good restaurant where they can meet for dinner. They start a PBP session and both start looking for possibilities on the web at sites like Yelp, Google Maps, etc. When they find something that looks interesting, they send a clip of the information to the other person for consideration. When they are ready to make a decision, they scroll back through the chat history to review the options they have collected. Once they actually arrive in New York, they refer back to this chat session on their mobile devices while on-the-go to double check which restaurant they decided on, and to get back to the original website to get the address.

Scenario 5: Sharing Web Content

Four teenage girls are best friends, and also obsessed with fashion. They all regularly browse a variety of web sites devoted to the topic. When they find a look they like, they clip it into their ongoing PBP session. All four girls check

this session regularly for updates and add their comments (as well as the latest gossip).

Summary

PlayByPlay supports a wide variety of interaction styles, from asynchronous to synchronous. As illustrated in *Scenario 5*, PBP sessions can be used in much the same way as email could be used to send links to web content for another person to view whenever she gets the chance. PBP provides the advantage of allowing for an ongoing conversational context though, unlike email. PBP can also be used in a highly synchronous manner. One user can act as leader and send her actions while other users follow along, thus functionally replicating the utility of synchronous collaboration systems while at the same time providing users with freedom and independence not found in master-slave approaches. Users also have the ability to choose when to send their actions and when to execute another user's sent actions, and can maintain a general awareness of where another user is browsing through the status thumbnail. Thus, PlayByPlay allows for a variety of modes and interaction styles, and for fluid switching between them.

EVALUATION

To evaluate PlayByPlay, we conducted two informal qualitative user studies. The first addressed stationary users using PBP through the Firefox extension, and the second addressed a mobile user collaborating with a stationary user.

Stationary User Evaluation

For the stationary evaluation we demonstrated the features of PBP to pairs of participants, and then asked them to complete three tasks to explore PBP's features.

Participants were all employees at the second author's research lab, recruited through the same e-mail distribution list used for the survey. There were a total of fourteen participants; thirteen were male, with ages fairly evenly distributed between 20 and 59. Eleven participants rated themselves as 'expert' in using the web. Participants were given a $10 lunch coupon for the lab cafeteria as compensation for participating in the study, which took 45-60 minutes to complete.

For the evaluation, users were seated in a usability lab facing opposite walls with a room divider between them, so as to simulate being in different locations. Participants were given an overview of the PBP system by the experimenter and then given a few minutes to try it out on their own before moving on to a set of three specific tasks. Participants were invited to voice their feedback at any time during the learning and exploring phases of the study and during the breaks between tasks in the latter phase of the experiment. The experimenter also prompted users with specific questions to encourage directed feedback on certain aspects of the system.

The first task was similar to *Scenario 3* above, in that one user was asked to place an order from grubhub.com while the other was asked to help. This task involved using the question-asking interface, as well as sending and interacting with a clip, in this case a menu.

The second task was similar to *Scenario 1* above. It consisted of asking one participant to 'teach' the other how to change the capitalization of his name in an online directory. This is a rather complicated process for the particular directory that we chose, making this type of teaching scenario quite realistic.

For the final task we asked participants to use PBP to accomplish a collaborative search task. We wanted to see how participants would collaborate and use the awareness features of PBP to monitor their partner's search strategies, current location of search, and progress made toward their joint goal. Specifically, we asked participants for this task to work together to find and 'clip' into the chat stream four images related to specific topics surrounding the Olympics: the original site of the Olympics, the 2008 site in Beijing, the original award, and the current medals.

We scheduled participants over a three day time period, and made changes to the system as we received feedback. Thus, users later in the study saw a different, and hopefully improved, version of the system compared to users earlier in the study.

Stationary User Evaluation Results

Throughout the study, we found many small tweaks were necessary to improve the quality of PBP's interface. For example, the language used for the toolbar buttons ("Record" changed to "Send My Actions") and the phrasing of recorded web actions in the chat log were changed to reflect the context of the user rather than that of the system ("click on the 'Search' button" changed to "user clicked on the 'Search' button"). We also received feedback that the recorded web action lines were difficult to distinguish from regular chat lines. This led us to add the person icon on action lines to differentiate them (as shown in Figure 1).

Users also requested an easy method for getting back in sync with the remote user. To address this need, we added the *Resync* button approximately halfway through the user study. Although this button is somewhat dangerous, as discussed earlier, the users who experienced it were able to make use of it. This suggests that the benefit of this button may outweigh its danger; however, our subjects were mostly above average users and the tasks used in our study did not include transactions that a user would not want accidentally repeated.

A pervasive problem with the system, especially early on in the study, was a lack of awareness features. During the teaching scenario, a very common comment from the participants playing the 'teacher' role was that they wanted to know where their 'pupil' was to verify that he was in the right place. Participants also said that during the collaborative search task it would be nice to have a high-level awareness of the site their partner was looking

at, as opposed to the fine-grained level of detail provided by the sent actions. This feedback and specific suggestions from participants led us to implement the status thumbnails. The thumbnails can also be used to determine whether the remote user is sharing web actions, as a recorded web action should appear at least once every time the thumbnail changes. Awareness of whether the other user was sending his actions was another feature participants requested. An indicator for when the other user is typing and an indicator for when the two users are in sync were also requested, although we have yet to implement these.

We found that special awareness features were particularly requested during the collaborative search task. For example, users requested an indicator of whether the other user had visited their current page. This is consistent with previous results in the collaborative search space [1, 12]. We are undecided on whether such features should be added to PBP. While collaborative search is an important and probably large subset of collaborative browsing tasks, it may be better supported by systems specifically targeting the search domain rather than more general-purpose systems such as PBP.

Participants generally liked the question-asking interface, but found it confusing at times. PBP (and the underlying CoScripter technology) relies on the descriptive text provided by websites to identify form elements (e.g., the 'address box'), and this text is not always clear. Participants suggested that the question also include a clip of the area immediately surrounding the form element in question in order to provide context, and we plan to add this feature in the future. It is noteworthy that subjects were comfortable with the special answer mode that is entered when a question is asked. We had expected that this might be a source of usability problems.

Overall, participant feedback was positive. The functionality seemed reasonably clear, and participants enjoyed using the system. Several participants even asked if they could get PBP on their own machines to try using outside the lab.

Mobile User Evaluation

Following the evaluation of PBP for stationary users, we conducted an additional study of PBP for mobile users. This study was more informal than the first study, and structured more as a discussion than as a set of tasks. Instead of pairs of subjects collaborating, we chose to have a single subject using the mobile device with the experimenter collaborating from the desktop client. As with the original study, we scheduled participants over two days and made some modifications to the software between subjects.

Four participants were recruited from the set of people who had participated in the first study. We chose to re-use participants so that the users in the mobile study would already be familiar with the basics of the system. Subjects once again received a $10 lunch coupon for participating in the study, which took about 30 minutes to complete.

During the study, participants sat at a desk adjacent to the experimenter. They were given an iPhone already running the PBP software over a Wi-Fi connection to our local network. Subjects were told to imagine that they were talking on the phone with the experimenter while interacting with the device. Note that this is possible in the real world both when using the 3G and Wi-Fi data connections, and thus represents a reasonably realistic scenario. We could have used a typing-only scenario, but our results would have been greatly limited by the slow typing speed that most people have with the iPhone's built-in keyboard.

The participants were then shown two promising mobile/desktop collaboration scenarios, corresponding to scenarios 2 and 3 as described above. For the initial demonstration, we used mapquest.com, which has a moderately complex form on its front page that is easy to access and clip for the mobile user.

After demonstrating these two scenarios, the experimenter and the subject engaged in a brainstorming session where specific mobile scenarios were discussed and then tried using the relevant web sites for those scenarios. As much as possible, the experimenter tried to elucidate scenarios that the subjects themselves had previous experience with; however, the experimenter would occasionally suggest a specific scenario as a starting point or to direct the brainstorming in a new direction.

Mobile User Evaluation Results

Three scenarios commonly came up in the study: the two used in the initial demonstration and another in which the stationary user sends information to the mobile device without any need for interaction by the mobile user. In this third scenario, the stationary user would either be sending a link, a phone number, or a clip which has an image that is useful without needing to open the clip. Of the two initially demonstrated scenarios, the participants all agreed that the scenario in which the mobile user can monitor the stationary user filling values into a form was the most compelling. The scenario that required the user to enter values was interesting to the participants, but text entry was a clear barrier to using PBP for such a purpose.

As with the stationary user study, we found that awareness was a challenge for using the mobile system. The lack of a chat log on the clip view screen (see b) is a particular problem, as users cannot see when they send messages by interacting with the clip and may not notice when actions from the stationary user are received and executed. This issue can be negotiated around when there is an additional voice channel, but is problematic when only the chat channel is available.

Two of the participants briefly tried using the system without the benefit of the voice channel. From this experience, we found that awareness that the mobile user

is typing is very important for the stationary user. Text entry takes so long on the mobile device that the stationary user quickly becomes concerned that there is a problem when there is not. A typing indicator is now a high priority for future implementation, as this was also requested by stationary users in the earlier study.

DISCUSSION

Browsing the web is an everyday activity for many people, in work and personal contexts, whether sitting at a computer or using the web on a mobile device while on-the-go. There are also common scenarios in which people need or want to share some aspect of their web browsing sessions. This may be in order to accomplish a specific task, like researching a topic; to plan an activity, like a vacation or a night out; or just to share something interesting. The focus may be on the *process* of browsing, such as collaborating to think of search terms or to split up efforts; or on the *product* of web activity, such as sharing the results of a web task or other information. Collaborative browsing consisting of a stationary user at a desktop computer acting as a web proxy for a mobile user is a particularly interesting case.

PlayByPlay can support all of these scenarios. When users are working on a task at the same time, they can monitor the search terms and web sites that other people are using. One person can also lead and others follow along, as in master-slave approaches; however, with PBP, the 'slaves' have the freedom to stop following along at any time without quitting the session. Not sharing the same display has the disadvantage of not guaranteeing that all session participants are in sync, but it does provide the advantage of user freedom. PBP also allows for fluid switching of leadership, although this still must be negotiated through chat. In addition, users can share only relevant pieces of web sites easily by clipping. Chat sessions also persist in users' accounts until deleted and can be renamed, so they can easily be used as reference.

CONCLUSION

We have presented *PlayByPlay*, a system that supports collaborative web browsing in both desktop and mobile contexts. It supports a range of interaction modes, from completely independent to tightly coupled browsing, and allows for fluid switching between modes in order to use whatever setup is most efficient for the task at hand. It also allows for easy sharing of pieces of web content. This functionality and the variety of interaction modes it supports make PlayByPlay potentially useful in a variety of situations.

ACKNOWLEDGMENTS

Our thanks to the USER group at the IBM Almaden Research Center, our survey participants, our user study subjects, and the CoScripter project members for their feedback on this work. We also wish to thank the reviewers for their helpful comments.

REFERENCES

1. Amershi, S. and Morris, M. CoSearch: a system for co-located collaborative web search, in *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*. 2008: 1647-1656.
2. Cabri, G., Leonardi, L., and Zambonelli, F. Supporting Cooperative WWW Browsing: A Proxy-based Approach, in *Seventh Euromicro Workshop on Parallel and Distributed Processing*. 1999: p. 138-145.
3. Cui, Y. and Roto, V. How people use the web on mobile devices, in *WWW '08: Proceeding of the 17th international conference on World Wide Web*. 2008: 905-914.
4. Greenberg, S. and Roseman, M. GroupWeb: A WWW browser as real time groupware, in *CHI '96: Conference companion on Human factors in computing systems*. 1996: 271-272.
5. Han, R., Perret, V., and Naghshineh, M. WebSplitter: a unified XML framework for multi-device collaborative Web browsing, in *Proceedings of CSCW*. 2000: 221-230.
6. Hinman, R., Spasojevic, M., and Isomursu, P. They call it surfing for a reason: identifying mobile internet needs through pc internet deprivation, in *CHI '08: CHI '08 extended abstracts on Human factors in computing systems*. 2008: 2195-2208.
7. Hupp, D. and Miller, R.C. Smart Bookmarks: Automatic Retroactive Macro Recording on the Web, in *User Interface Software & Technology*. 2007: 81-90.
8. Jacobs, S., Gebhardt, M., Kethers, S., and Rzasa, W., Filling HTML forms simultaneously: CoWeb—architecture and functionality. *Computer Networks and ISDN Systems*, 1996. 28(7-11): 1385-1395.
9. Leshed, G., Haber, E., Matthews, T., and Lau, T. CoScripter: Automating & Sharing How-To Knowledge in the Enterprise, in *CHI*. 2008: 1719-1728.
10. Microsoft, "Get started using Remote Desktop with Windows XP Professional," 2006. http://www.microsoft.com/windowsxp/using/mobility/getstarted/remoteintro.mspx.
11. Morris, D., Morris, M.R., and Venolia, G. SearchBar: a search-centric web history for task resumption and information re-finding, in *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*. 2008: 1207-1216.
12. Morris, M. and Horvitz, E. SearchTogether: an interface for collaborative web search, in *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*. 2007: 3-12.
13. MozillaWiki, "Labs/Joey," 2007. https://wiki.mozilla.org/Labs/Joey.
14. RealVNC, "VNC Enterprise Edition for Windows Mobile Platforms," 2008. http://www.realvnc.com/products/beta/ce/.