Artifacts as Instant Messenger Buddies

Saul Greenberg, Nathan Stehr and Kimberly Tee
Department of Computer Science, University of Calgary
Calgary, Alberta CANADA T2N 1N4
saul.greenberg@ucalgary.ca

ABSTRACT
Artifact awareness is one person's up to the moment knowledge of the artifacts that other group members are working with. Such awareness contributes to the overall information necessary for fluid group coordination and interaction. Yet current systems treat artifact awareness quite differently from the interpersonal awareness of group members using the artifact. Our approach differs. We exploit commercial Instant Messengers (IMs) for artifact awareness, where we treat an artifact – such as a document – as a 1st class buddy. As a person uses the artifact, changes are triggered in the artifact's online, idle and offline state. Further information about artifact events is published to the IM's 'display name' or 'personal message' field. Important artifact activities – such as major version updates – are delivered as chat messages. Others can engage in a chat 'dialog' with the artifact that includes directives to receive and transmit artifact versions. The group's conversation around that artifact is also recorded as part of the chat history.

Author Keywords
Artifact Awareness, Instant Messengers.

ACM Classification Keywords
H.5.3 [**Group and Organization Interfaces]**: Computer supported cooperative work.

INTRODUCTION
An important component of the awareness necessary for casual interaction is *artifact awareness*: one person's up to the moment knowledge of the artifacts that other group members are working with. Such artifacts include the documents and drawings that individual group members work on over the course of a day as they pursue their collective work. Whittaker and colleagues [8] found that over half of all casual interactions in an office involved some form of document sharing, where documents were mostly used as a cue or conversational prop. As detailed in [5, 8, 7] and summarized by Tee [6], being aware of such artifacts is valuable for many reasons:

- *Monitoring* each other's progress on the document;
- *Coordinating* their joint activities;
- *Triggering interest* by seeing another person's activity, even if it is not part of a joint task;
- *Determining availability*, where knowledge of artifact usage suggests how busy people are and if they can be interrupted.
- *Creating serendipitous opportunities* for people to engage in artifact-oriented collaborations.

Artifact awareness is easy when people are co-located (primarily because of the artifact visibility), but is problematic for distributed groups. Consequently, various technical solutions providing artifact awareness have been implemented (see [7] for examples). 'Explicit push' occurs when people explicitly send artifacts to others as part of a communication dialog, such as file exchange via email or within an instant messenger chat session [7]. 'Explicit pull' occurs when people retrieve files that were somehow made available by others, e.g., with peer-to-peer file-sharing, document repositories [1,7], and version control systems. 'Artifact availability' happens when people post artifacts through some kind of awareness server that displays the artifact state, e.g., Notification Collage [4], Community Bar [6], Sharing Pallette [7] and OpenMessenger [2].

While these later systems provide a fine granularity of artifact awareness [4,6,7], they use specialized software that are yet another awareness device present on the desktop that competes for attention. In addition, many systems disassociate artifact awareness from the interpersonal awareness of actual group members interested in that artifact. This is a problem: we know that in everyday life, artifact awareness is closely associated with interpersonal awareness of the people that use them [8,2,7]. Instead, we believe we can incorporate artifact awareness within an existing highly-used awareness system: Instant Messengers (IMs). IMs let ad-hoc groups of friends (buddies) stay aware of one another and move into easy conversation and interaction. Our main idea is that artifacts can become a 1st class IM buddy and behave like other buddies within a defined group. The artifact knows what people are interested in it and notifies others about its state. Others can interact with the artifact (and the rest of the group) through IM's standard chat features.

Cite as:
Greenberg, S., Stehr, N. and Tee, K. **Artifacts as Instant Messenger Buddies.** Report 2008-896-09, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada. T2N 1N4

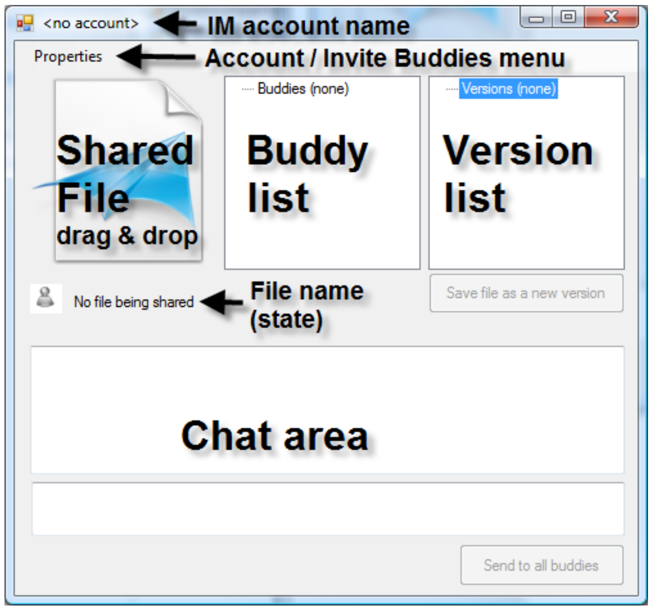


Figure 1. The (annotated) Artifact Buddy as it first appears.

ARTIFACT BUDDY

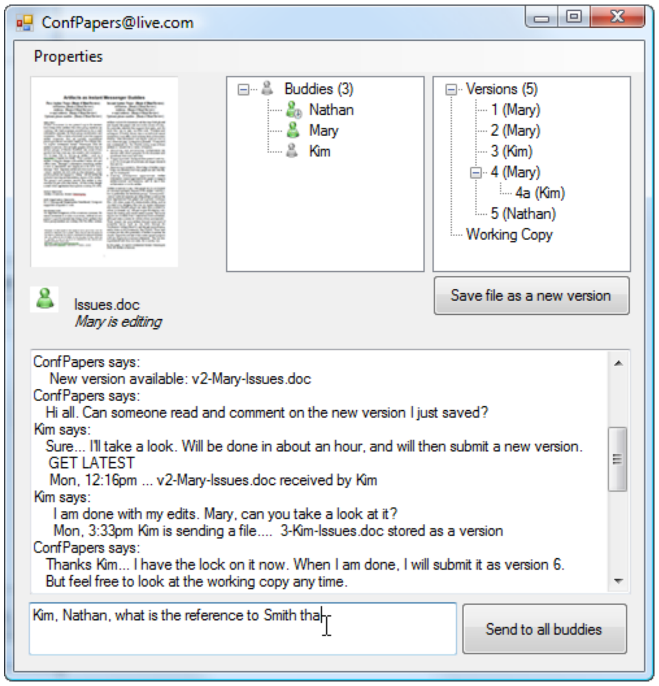
We call our experimental system Artifact Buddy. It implements a user interface (annotated in Figure 1) and a wrapper around Microsoft's Windows Messenger, chosen because it has functions typical of most IMs as well as a public API (we use the DotMSN API from http://www.xihsolutions.net/dotmsn). Through this API, Artifact Buddy programmatically invokes activities such as inviting buddies, setting and receiving state information, sending and receiving chat messages, initiating and responding to file exchanges, and so on.

We explain Artifact Buddy by scenario. Mary is working on a conference paper with her co-authors Kim and Nathan. As the primary author, Mary keeps the 'master' copy of the paper, and coordinates with Kim and Nathan over changes and updates to it. Because she holds the master, she is the only one that needs the special Artifact Buddy software.

Creating an IM account. Mary starts up Windows Messenger, and using its standard interface signs up for a new account ID on it (identified by a made-up email address) to represent document(s) relevant to her group. She calls it 'ConfPapers@live.com'.

Linking Artifact Buddy to Windows Messenger. Mary then starts Artifact Buddy, which appears on her screen as annotated in Figure 1. Using the functions accessible through the 'Properties' menu (top left), she provides the Windows Messenger account ID and password. Using this information, Artifact Buddy can now link into to Windows Messenger by logging onto that account through the Messenger API.

Inviting Buddies. She now invites the buddies to form a group, i.e., people with interested in the document to be shared. Specifically, she provides Kim's, Nathan's and her own Windows Messenger IDs (their email addresses) to



ConfPapers says:
New version available: v2-Mary-Issues.doc
ConfPapers says:
Hi all. Can someone read and comment on the new version I just saved?
Kim says:
Sure... I'll take a look. Will be done in about an hour, and will then submit a new version.
GET LATEST
Mon, 12:16pm ... v2-Mary-Issues.doc received by Kim
Kim says:
I am done with my edits. Mary, can you take a look at it?
Mon, 3:33pm Kim is sending a file.... 3-Kim-Issues.doc stored as a version
ConfPapers says:
Thanks Kim... I have the lock on it now. When I am done, I will submit it as version 6.
But feel free to look at the working copy any time.

Kim, Nathan, what is the reference to Smith that

Figure 2. Artifact Buddy after a period of use.

Artifact Buddy (again via the Properties menu). Under the covers, Artifact Buddy will use the Messenger API to send the invitation to each of them and then monitor their state. As each invitee accepts the invitation, their names will appear in the Artifact Buddy 'Buddy List' pane along with an icon indicating each buddy's on-line, idle and off-line state (Figures 1 and 2, top middle). Following the norms of Windows Messenger, the Artifact Buddy account name (shortened as a nickname) also appears as a new contact in each person's Windows Messenger buddy list, i.e., 'ConfPapers'. Figure 3 illustrates this in Kim's view, where for convenience she has created a new group within Windows Messenger called 'Conference Paper Group' and moved Mary, Nathan and the 'ConfPapers' artifact-oriented buddy into it.

As an alternative to the artifact inviting people, others could instead use the standard IM features to invite the artifact to be their buddy (i.e., by inviting ConfPapers@live.com). Currently, Artifact Buddy is configured to accept all such invitations; this access could be restricted to by comparing requests to an access control list.

Sharing an artifact. Next, Mary drags and drops the artifact to be shared, in this case a word document titled 'Issues.doc', onto the Shared File icon (Figure 1, top left). Artifact Buddies immediately creates and stores a copy of this file as version. It displays it as a thumbnail (Figure 2 top left) and as an item in the 'Version List' pane, where it is identified by the version number and the person who has submitted this version (Figures 1 and 2, top right). Internally, this version copy is stored in a local directory called 'versions', where the file name is prefixed with the version number and its creator, i.e., v1-Mary-Issues.doc.

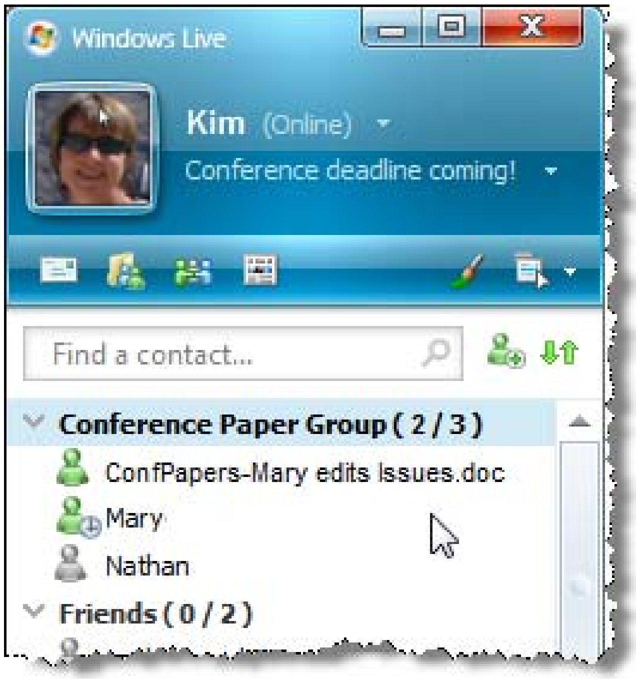

Windows Live
Kim (Online)
Conference deadline coming!
Find a contact...
Conference Paper Group (2 / 3)
ConfPapers-Mary edits Issues.doc
Mary
Nathan
Friends (0 / 2)

Figure 3. Kim's IM window showing ConfPapers

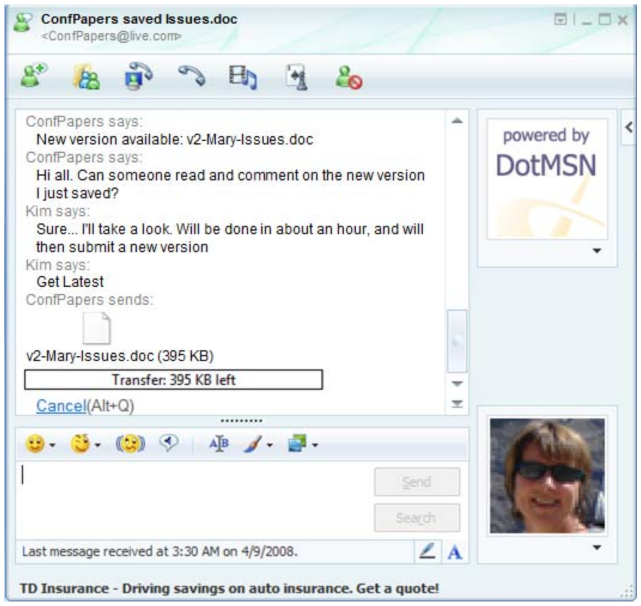
The most current copy of the file is also kept by Artifact Buddies as the 'working document' and is also listed in this pane. This working document is the one that Mary (and others) normally edit; it does not become a new version until it is submitted as such (explained shortly).

As an aside, Mary can reuse this account with different documents, although this would likely occur only after the group had completed its work on the previous one.

Basic artifact awareness. Mary now begins to work on this document in the normal way. Other people see her activity in their standard Windows Messenger client, as illustrated in Figure 3. They see that the artifact as 'buddy' is online (the green pawn). They also see additional detail in the display name field associated with the buddy. As Mary edits, the display name changes to say she is editing the file (as illustrated in Figure 3). If she pauses editing or if she closes the document, the icon changes to its 'away' state. If she shuts down Artifact Buddy or logs off her computer, it is displayed as off-line. For each case, the display name message field changes accordingly to provide detail.

Under the covers, Artifact Buddy monitors if the file is opened by periodically examining the names of opened windows on the display. If the file is opened in a window, it tracks keyboard activity in that window to detect if the person is editing the file. It then translates this into state and display name messages transmitted through the Windows Messenger API.

Saving versions. At any point, Mary can specify that a copy of the working file should be saved as a new version (the 'Save file as new version' button shown in Figure 2, middle right). Internally, Artifact Buddy implements a


ConfPapers saved Issues.doc
<ConfPapers@live.com>

ConfPapers says:
New version available: v2-Mary-Issues.doc
ConfPapers says:
Hi all. Can someone read and comment on the new version I just saved?
Kim says:
Sure... I'll take a look. Will be done in about an hour, and will then submit a new version
Kim says:
Get Latest
ConfPapers sends:
v2-Mary-Issues.doc (395 KB)
Transfer: 395 KB left
Cancel(Alt+Q)

Last message received at 3:30 AM on 4/9/2008.

TD Insurance - Driving savings on auto insurance. Get a quote!

Figure 4. A sample of conversations, commands and events as they appear in the IM Chat Dialog window.

lightweight albeit simplistic version management system. It copies the file to the storage location mentioned previously, numbers that version, and tracks who has submitted it. That version and the name of its submitter then appear on the Version list (Figures 1 and 2, top right). At any time, Mary can open previous versions by double clicking it in the list.

As will be mentioned shortly, other buddies can get previous versions of the document via variants of the 'Get' command and submit them as well via drag and drop into the IM chat dialog box. When a previous version is resubmitted, it is saved as a new version. That is, if the version 4 was edited and then resubmitted, it is saved as version 5. However, if a new version had already been submitted by someone else, the system will save the current version as a child version. For example, the version tree shown in Figure 2 (middle right) shows that Kim has retrieved, edited and resubmitted version 4 of the document. Since Nathan had already submitted version 5, Kim's is stored and listed as version 4a.

When a version is saved, Artifact Buddy initiates a chat dialog with all other buddies. For each buddy, a standard Windows Messengers chat box appears, exemplified by the dialog in Figure 4. Artifact Buddy sends a message indicating that a new version of a file is available (for certain types of files it also summarizes the differences between the current and previous version). Dialogs are also recorded in the Artifact Buddy chat area (Figures 1 and 2).

Conversing with Artifacts. Buddies can initiate a dialog with the artifact, where any person can send it commands via the standard IM chat box. For example, if Kim types 'Get Latest' into the chat box, Artifact Buddy will respond by initiating a file transfer of the latest version of the paper, just as if 'Artifact Buddy as person' had dragged and dropped the file into the chat area (see Figures 2 & 4). Similarly, any buddy can do the following:

- ***Get #*** gets a particular version identified by its number;
- ***Get original*** gets a copy of the original document, i.e., it is the same as Get 1;
- ***Get latest*** gets the latest saved version of the document;
- ***Get working*** gets the current working copy of the file;
- ***<Drag and drop> version*** submits the dropped file version to the version control system;
- ***Versions*** lists all available versions;
- ***Get transcript*** gets a copy of the entire dialog entered by people as they converse with the artifact and with each other, as well as the events generated by the artifact.
- ***Help*** lists all commands.

All submitted commands are also displayed in the Artifact Buddy chat area (Figures 1 and 2, bottom) as well as the chat dialogs of other buddies (Figure 4). This provides both the master and others with ongoing awareness of all activity concerning these files and versions.

Conversing with the group. Any buddy can not only converse with the artifact, but with all buddies associated with that artifact. If the text entered in any IM chat dialog window is not recognized as a command, Artifact Buddy assumes it is a message intended for the group and passes it through to all participants. Similarly, the master – in this case Mary – can use the Artifact Buddy chat window to broadcast messages to all other buddies. We can now see that the (different) dialogs shown in Figures 2 and 4 contain a mix of commands, events, interpersonal dialog, and file transfers.

A transcript of all messages are retained in an XML file by Artifact Buddy. At any time, any person can request and review the transcript of events, of submitted commands, and of the group dialog around the artifact through the 'Get transcript' command. This ability to review the transcript is especially important for people who have been off-line for a while, as it helps them get up-to-date.

DISCUSSION

Artifact Buddy is a demonstration that realizes our basic idea: that artifact awareness can be managed through the standard interpersonal awareness and communication capabilities of Instant Messengers and others of its ilk. The idea can have many variations that go far beyond what we have detailed here. Some suggestions follow.

Document-level locking can be managed by having the Artifact Buddy track the state of who 'owns' a particular version. It can enforce locks, where only that person can submit the next version. It can also allow for greater control of branching versions, as is done in many version control systems. Indeed, it should be straightforward for a system like Artifact Buddy to use an existing version control system as its underlying document management engine.

Multiple artifacts. Our implementation only manages one document at a time. It could be extended to handle multiple documents. For example, a variant of Artifact Buddy could track multiple files, where it provides awareness of their global state both as a collection, and as details of particular files as they are being used (e.g., in the display name and/or the chat dialog). People can submit and received files individually, or as a collection (e.g., as a zip file). Of course, this will add interface complexity.

Artifact awareness as streamed window snapshots. Tee et. al. [6] advocated screen sharing as a method to display on-going changes to a file. Their idea was to transmit a miniature of a screen or window's image to other people, which can then be visually tracked to see changes as they occur. This too can be implemented in an IM-based system. Many Instant Messengers now have the capabilities to open a video channel, e.g., for web cam conversations. This can be exploited by having a system like Artifact Buddy take periodic window snapshots of the artifact as it is being edited, and repackage and stream them on the fly as video frames over the IM video channel. It would use the standard IM API and video codec protocol to transmit these frames.

Linking into other awareness services. There is a multitude of other awareness servers now available. These include web-based social networks (e.g., Facebook, MySpace), sidebar utilities (e.g., Microsoft Sideshow [1], micro-blogging systems (e.g., Twitter http://twitter.com/) and so on. Each has affordances built for interpersonal awareness that lend themselves to artifact awareness in ways somewhat similar to what we have done with Instant Messengers. For example, an artifact can register as a 'person' on a social network, and update others by posting events or micro-blogs. More detailed summaries can be posted as blogs. Depending on the social network, images of the changing artifact over time can be posted as pictures or as shared files. All that is needed to implement this is a public API to that awareness service.

In summary, we believe that artifact awareness can exploit the standard interpersonal awareness and communication capabilities of social systems such as Instant Messengers. Artifact Buddy is just one working example to illustrate how this can be done; many other variations are possible.

ACKNOWLEDGEMENTS

This work was partially funded by the NSERC/iCORE/SMART Industrial Research Chair in Interactive Technologies and by NSERC through its NECTAR Research Network and Discovery Grant programs.

REFERENCES

1. Bentley, R., Horstmann, T. and Trevor, J. The World Wide Web as Enabling Technology for CSCW: The Case of BSCW. *Computer Supported Cooperative Work*, 6, 1997, 111-134.

2. Birnholtz, J., Gutwin, C., Ramos, G., and Watson, M. (2008) OpenMessenger: Gradual Initiation of Interaction for Distributed Workgroups, *Proc ACM CHI 2008*.
3. Cadiz, JJ, Venolia, G.D., Jancke, G., and Gupta, A. Designing and Deploying an Information Awareness Interface. *Proc ACM CSCW*, 2002, 314-323.
4. Greenberg, S. and Rounding, M. The Notification Collage: Posting Information to Public and Personal Displays. *Proc ACM CHI*, 2001, 515-521.
5. Kraut, R., Egidio, C., and Galegher, J. Patterns of Contact and Communication in Scientific Research Collaboration. In *Intellectual Teamwork: Social & Technological Foundations of Cooperative Work*. LEA Press, 1990, 149-181.
6. Tee, K., Greenberg, S. and Gutwin, C. Providing Artifact Awareness to a Distributed Group through Screen Sharing. *Proc ACM CSCW*, 2006.
7. Voida, S., Edwards, K., Newman, M., Grinter, R. and Ducheneaut, N. Share and Share Alike: Exploring the User Interface Affordances of File Sharing. *Proc. ACM* CHI, 2006, *221-230*
8. Whittaker, S., Frolich, D., and Daly-Jones, O. Informal workplace communication: What is it like and how might we support it? *Proc ACM CSCW*, 1994, 131-138.