Broadcast Messaging

Messaging to the Masses

This powerful form of communication has social implications as well as technical challenges.

FRANK JANIA, IBM

"We want this and that. We demand a share in that and most of that. Some of this and - - - in' all of that. And their demands will all be changed then, so - - - in' stay awake."¹ Comedian Billy Connolly wasn't talking about messaging when he said this, but I don't think there is a more appropriate quote for the voracious hunger we have for information as a result of the messaging-enabled, network-connected world that we take for granted every day.

We have instantaneous access to petabytes of stored data through Web searches. With respect to messaging, we have an unprecedented number of communication tools that provide both synchronous and asynchronous access to people. E-mail, message boards, newsgroups, IRC (Internet relay chat), and IM (instant messaging) are just a few examples. These tools are all particularly significant because they have become essential productivity entitlements. They have caused a fundamental shift in the way

Broadcast Messaging

Messaging to the Masses

we communicate. Many readers can attest to feeling disconnected when a mail server goes down or when access to IM is unavailable. For some of us, network outages are now as inconvenient as a blackout.

These tools are also significant because they represent technologies that provide a means for enhanced interaction. On one end, in the case of e-mail, the technology provides increased delivery speed over that of standard post. At the other extreme, in the case of IM, the ability to advertise awareness information and have a realtime text conversation comprises a new form of communication. Broadcast messaging is a technology that falls somewhere in between, and has several use-cases that highlight its efficacy and indicate that it also will someday enjoy the ubiquity of IM. There are, however, social implications to providing broadcast messaging to a large audience, as well as challenges in building broadcast messaging tools for such an audience.

BROADCAST MESSAGING: A DEFINITION

We can run through a real-life scenario that is an analogy for broadcast messaging. You've flown to a conference in a city that you have never been to before, where you are presenting a paper to an audience of 2,000 people. You plan to stay an extra day after the conference and would like to know what that city's "must-see" attractions are. Go ahead and ask the members of the audience. A few people start shouting out responses; you get the idea that this city has more than a few interesting sites to see. There is a group of attendees in the front who seem fairly insistent that you go to one area of the city because that is the best. You walk over there and engage them in a discussion to get some more insight. You do the same with another group that offers up its own compelling argument to go someplace else. With your options narrowed to two places, you ask the audience to vote for one place or the other by a show of hands.

The above example is fairly trite, but we should consider the possibilities if this same situation is available to you and the query is something along the lines of debugging a server issue, finding a person with a special skill set for a customer presentation, or discussing possible marketing campaigns. At first glance this may sound like another application of IM, but there is an important

distinction. In the case of broadcast messaging the audience is undefined.

In broadcast messaging, users broadcast messages to topics. Other users listen on those topics and can choose to act on messages or not. The message is essentially a request for interaction from some or all of the recipients. That request for interaction can be something like a request to chat or answer a poll. The implementation of broadcast messaging we will look at is ICT (IBM Community Tools), where the topics are called "communities" and the request for interaction takes one of five forms.

ICT: A REAL-WORLD IMPLEMENTATION

ICT is a suite of applications that incorporates broadcast messaging and IM. The most prolific use-case of ICT is the IBM internal deployment, with an average of 18,000 users per month. There are five applications for broadcast messaging: w3alert, TeamRing, SkillTap, FreeJam, and PollCast. Users broadcast many types of requests to one of many communities, but the most active is the "everyone" community. This is the community that everyone listens to by default. The novel feature of communicating to "everyone" is circumventing the need to categorize your request while getting it out to a large audience of potential responders. The ability to broadcast to everyone can be very powerful, but it also has social implications and technological challenges.

I'll first discuss the specifics of ICT's broadcast applications and then their social implications and technological challenges.

w3alert is the simplest case application. It allows users to broadcast informational messages to a community, with an optional URL for more information.

TeamRing is an application for sharing Web presentations. It allows users to send out invitations to a Web-based presentation, and then broadcast the URL of the current page to all users viewing the presentation.

SkillTap is by far the most utilized application. A user broadcasts a request for help or information to the community. A responder will start an instant messaging session with the requester and provide the requested information. An additional feature of SkillTap allows the user who responded to submit his response to a searchable FAQ database.

FreeJam allows users to create a just-in-time chat room where the topic of discussion is based on the user's broadcast. For example, a user may broadcast a request to discuss problems installing a software package.

PollCast is a polling application. A user broadcasts a multiple-choice poll to a community of users who can

respond to the poll while the results are tabulated and graphed in realtime.

FreeJam and PollCast are potential candidates for a form of persistence similar to the SkillTap FAQ database. While an individual can store the results of a poll or FreeJam transcript locally, there has been no global-persistence model implemented for either due to some social and technological concerns. Of course, storage is a concern. As the community grows, provisioning and providing storage becomes less than trivial. Questions come up such as: How long do you store the data? Who has access to it? Does everyone have access to a FreeJam transcript or just the people who participated in the discussion? Privacy questions arise as well: Do you allow someone to opt-out their portion of a FreeJam discussion? In the model used by SkillTap for FAQ entries, submission is at the discretion of the responder. So how does that work when multiple people contribute to a discussion? These issues are easily resolved compared to those that crop up when deploying a broadcast messaging system.

BROADCAST MESSAGING IN PRACTICE

The social implications of allowing any user to broadcast to a large group of other users are powerful, but not without peril. One particular phenomenon has been "PollCast storms." The scenario plays out like this: A user will broadcast a PollCast to everyone, requesting responses to multiple-choice questions about the operating system that each user runs. The outcome of the PollCast is a dynamically updating results graph that some users, new and veteran, find really fun to watch. Consequently, another user will send out their own PollCast, followed by two or more users doing the same. There will then be a set of users who feel this is an inappropriate use of the system and, ironically enough, send out a PollCast to see if users feel that the tool should be used in that way. This is sometimes followed by a poll in response to the previous one asking about sending out polls inquiring if it is appropriate to send out certain polls. It seems silly, and overall that is the opinion. But when you have a large population, all with an equal opportunity to speak to the entire community, the communication from some careless users can become overwhelming to everyone else.

A host of potential difficulties emerge when broadcasting to a large audience. The use of lewd or foul language is often not appreciated; one person can end up offending many people. The content of the broadcast may also fall into a category that is neither foul nor lewd, but in some other way inappropriate. This is further complicated by the fact that the user base is diverse and multinational,

ICT in IBM ... and Beyond

Here are some statistics for the current ICT user base in IBM's internal deployment:

- 20,000 users per month (average)
- 6,000 peak simultaneous users
- 360 broadcasts to "everyone" per month (average)
- 1,021 communities open to every ICT user
- 437 communities with 100 or more members
- 53 different countries with ICT users
- 80 percent of the community members in technical job roles

In addition to the IBM internal deployment of ICT, we have deployed a version for use outside of IBM. That release was originally targeted at a community of A/S 400 professionals as a means of further building their community. Recently we also targeted visitors to the alphaWorks Web site (see <http://alphaworks.ibm.com/>).

There is more information about ICT, and a link to the ICT download, at <http://community.ngi.ibm.com>.

with varying opinions on the meaning of "appropriate." Real examples of inappropriate broadcasts include messages that are directed to everyone, but only meant for a few individuals; discussions of salary or other private information; or broadcasts with trivial or incomplete content. These are mostly social difficulties, but technological difficulties exist as well.

The biggest technological difficulty so far is testing the end-user software and server infrastructure as the user base grows. Sometimes the only effective test of a new feature is to roll it out to the community. As the numbers of users grow—and with a broadcast messaging system they grow virally—ensuring that users run on the same version of the software becomes a challenge. In practice, even with an auto-updating function in the client, not everyone updates to the latest version. There have been a few updates to the ICT client that require all users be running the same version of the software in order for everyone in the community to communicate. These changes have been in response to server load issues, or implementation of new features or functions.

The only true test of the infrastructure's capacity to handle daily usage as the user base grows is to monitor the system in realtime. Along with viral growth comes an increased demand on the server that is only loosely predictable. Provisioning for this demand has been an iterative process. With a technology such as broadcast messaging, which still is in developmental stages—it is

Broadcast Messaging

Messaging to the Masses

helpful to have a large user base of patient early-adopters that will help prove the technology's worth.

Within a contained environment such as a corporation, there are certain safeguards inherent to the community. All broadcasts on the system are tagged with the sender's user ID, which provides a level of accountability. None of the broadcasts is anonymous—with the exception of poll responses. Since anonymity typically breeds honesty, a user's response to the multiple-choice poll is anonymous. In the case where a user broadcasts a message that contains foul or lewd language, one possible recourse is to contact that person or the manager. But even in this case there is no automation, and the administration team may not catch every broadcast, so it is up to a concerned user to speak up.

Even within a closed environment, the ability for individual users to take action against inappropriate content is important, and in an open environment—such as the Internet in general—there must be a solution that empowers the community to moderate itself. This would be a solution whereby members of the community take realtime action against inappropriate use.

One such solution implemented in PollCast is to append a choice to all of the multiple-choice polls that says, "This poll was inappropriate." This has cut down on the frequency of "PollCast storms." It is a simple

allow members of the community to indicate when a user is abusing the system, which would in turn disable the broadcast privileges for the offending user. There are concrete implementation issues with this option. The current server infrastructure uses a broadcast engine that was never envisioned to take on the task of community discussion, but rather disseminate messages to a large number of clients quickly. The servers don't know about people or content, just connections. As a result, there is no mechanism for storing profile and privilege data for a particular user. One solution under development is for the audience to express dissatisfaction with a user via an out-of-band message, which would affect the operation of the offending user's client for some period of time.

TOO MUCH INFORMATION

Even if the problem of inappropriate broadcast content or people abusing the system is solved, there is still an issue that grows in importance with the growing size of the community. A difficulty worth specific focus is "inundation." In order to make broadcasting to "everyone" effective as the size and diverse interests of the community increase, users must not feel inundated by the increased number of broadcasts. Avoiding inundation, while still maintaining the efficacy of the broadcast mechanism, is a top concern. ICT currently has two methods for handling inundation: filters and UI (user interface) control. Rate-limiting is a possibility that is under investigation.

Filtering broadcasts to everyone is one of the novel features of ICT. A user can specify one of two types of filters to control the broadcasts received from a particular

community for any of the broadcast applications. The keyword filter is configured by listing a set of keywords that an incoming broadcast is tested against. The filter specifies whether the broadcasts containing these keywords are to be shown to the user or discarded. The shortcomings of this type of filter are readily apparent: How do I list my interests (or disinterests) effectively? At first glance it appears too limiting to be useful—but consider the user base. The real goal of providing a broadcast function is not to reach the whole community per se, but

to reach a representative population. So if your user base grows, say, in the millions, there is a good chance that if a broadcast reading "How do I install MySQL on Redhat 8" is filtered out by a user that includes only "Linux" in his filter set, another user will receive that broadcast because his filter set includes "Redhat" and "Linux."

solution that lets users indicate dissatisfaction with other community members without being directly confrontational. This has been fairly effective for PollCast in our closed environment, but there is still the need for a more general solution. Among the options that we are considering implementing are forms of moderation that would

**Filtering
broadcasts
to everyone**
is one of the novel
features of ICT.

As we all know, users' preferences vary, and for some the prospect of missing out on any percentage of relevant broadcasts is unacceptable. Thus there is a need for more sophisticated filtering. The other type of filter currently available, an adaptive filter, is based on the Bayesian spam-filtering technique. If an adaptive filter is enabled and being trained by the user, each window that displays a broadcast includes a button that lets users classify the messages as types of messages they want to see or not.

Plans for advancements in filtering in future versions of ICT will include leveraging a number of sources of data to build filters automatically and update them dynamically according to the changing interests of the user. The goal would be to provide intelligent functionality that would act as a proxy for the user: Only show relevant messages. Resources such as chat transcripts, browsing history, e-mail, and others can be used to produce an interest profile that can be the basis for filtering or determining in what manner the broadcast is displayed. All information would be stored entirely on the client side. The user decides whether to interact with the broadcaster, so the privacy of those resources is not compromised.

Ideally, in the case where filters become highly effective, filtering could replace most static communities. Communities that represent teams of coworkers and have a defined list of members are important for communication among that team only for reasons of confidentiality or scope. For general issues such as "How do I install MySQL on Redhat 8?", letting a filter determine the interested members has the potential to reach far more people and produce deeper discussion and interaction than if the query had to be categorized a priori.

How the broadcasts are displayed can affect the perception of inundation as well. The delivery of an instant message—as a window that grabs focus and stays active until the user dismisses it—is not necessarily suited to broadcast messages. Whereas instant messages are directed to a particular user, broadcast messages are not. Without the assurance that only relevant messages will be delivered to the user, it is important that the delivery of the message be as minimally intrusive as possible, while also effectively displaying the message. Since "minimally intrusive" and "effective" are all highly subjective, the way the message is displayed must be highly configurable by the user. In the current implementation of ICT, the default window that displays a broadcast message does not grab focus; it appears in the lower right-hand corner of the user's display and automatically disappears after ten seconds. The size of the window, length of time it displays, location it displays, and a number of other

features are all configurable. These settings are all in response to a large amount of varied user feedback. In general, the user interface issue associated with displaying realtime alerts in a manner that is not annoying—especially when those alerts can occur frequently—is an interesting challenge.

"Rate limiting" is another solution to the problem of inundation. Rate limits can be applied on the sender's end or the receiver's end. On the sender's end, a user can be limited by the system or the client as to the number of broadcasts to be sent out in a period of time—or the system can limit the number of broadcasts allowed in a period of time to a particular community. These rate limits can be determined by the administration of the system, or generated dynamically based on time of day, historical usage, or in realtime by members of the community. On the receiver's end, one particularly straightforward and easy-to-implement solution would be to give the user the option to configure the client to only display broadcasts at a predefined rate. As with keyword filtering before, when the user base grows large enough—even if a fair percentage of users elect to see no more than, for example, two broadcasts every five minutes—the possibility of reaching a representative population of decent size is not precluded by this type of rate limit.

PEOPLE HELPING PEOPLE

Of all the social implications we have observed with the deployment of a broadcast messaging application to a large audience, one very positive one is the demonstration of people's willingness to help each other. This is particularly evident with SkillTap. To date there have been almost 45,000 questions asked, and almost 20,000 of them have been answered. The real power of broadcast messaging becomes very apparent when we highlight the fact those questions each got answered in about a minute, by people for whom support is not an occupation, without charge. Q

REFERENCES

1. Eircom: see <http://homepage.tinet.ie/~shaykelly6780/soundsfunny.htm>.

LOVE IT, HATE IT? LET US KNOW:

feedback@acmqueue.com or www.acmqueue.com/forums

FRANK JANIA is a software engineer with IBM's Internet Technology Team. He is the team leader from the development of the ICT client.

© 2003 ACM 1542-7730/03/1100 \$5.00.