

SHOCK: Communicating with Computational Messages and Automatic Private Profiles

Rajan M. Lukose, Eytan Adar, Joshua R. Tyler

Information Dynamics Lab, HP Laboratories
1501 Page Mill Road
Palo Alto, CA 94304 USA
+1 650 857 1501

{lukose,eytan,jtyler}@hpl.hp.com

Caesar Sengupta

Encentuate Pte. Ltd.
151 North Buona Vista Road #02-45
Singapore 139347, Republic of Singapore

caesars@cs.stanford.edu

ABSTRACT

A computationally enhanced message contains some embedded programmatic components that are interpreted and executed automatically upon receipt. Unlike ordinary text email or instant messages, they make possible a number of useful applications. In this paper, we describe a general and flexible messaging system called SHOCK that extends the functionality of prior computational email systems by allowing XML-encoded SHOCK messages to interact with an automatically created profile of a user. These profiles consist of information about the most common tasks users perform, such as their Web browsing behavior, their conventional email usage, etc. Since users are sensitive about such data, the system is designed with privacy as a central design goal, and employs a distributed peer-to-peer architecture to achieve it. The system is largely implemented with commodity Web technologies and provides both a Web interface as well as one that is tightly integrated with users' ordinary email clients. With SHOCK, users can send highly targeted messages without violating others' privacy, and engage in structured conversation appropriate to the context without disrupting their existing work practices. We describe our implementation in detail, the most useful novel applications of the system, and our experiences with the system in a pilot field test.

Categories and Subject Descriptors

D.2.11 [Software Engineering]: Software Architectures – *information hiding, patterns (e.g., client/server, pipeline, blackboard), domain-specific architectures.*

General Terms

Design, Economics, Human Factors

Keywords

Web profiles, privacy, recommendation systems, knowledge management, email, peer-to-peer networks, XML, SOAP.

1. INTRODUCTION

Ubiquitous electronic messaging systems such as conventional email, instant messaging, and even voice telephones are all explicitly “addressed” systems. That is, one can only send a message to someone whose explicit email address, instant message handle, or telephone number is known beforehand.

However, one can imagine many other less explicit, but perhaps more specific, types of message “addressing”. For example, one may wish to send a message to people who have viewed a certain Web page such as newspaper story or an online product description or a scientific abstract. Perhaps it may be useful to send a message and start a conversation with others in an organization that have performed Web searches on a specific topic like “Apache security” or “Java XML parser”, or who have had experience installing a particular software package such as Freenet. At the same time, users who participate in such a messaging system may not be willing to allow their Web usage to be publicly searchable—they are likely to be concerned about their privacy. Beyond targeting, it is frequently useful to merge and analyze responses in some way (e.g. tabulating survey results, finding a time to meet, showing discussion threads, etc.). Without sophisticated natural language processing this task generally falls upon the user.

In this paper, we describe a general and flexible messaging system called SHOCK (Social Harvesting of Community Knowledge) that allows such message targeting while simultaneously maintaining user privacy. This is achieved by client software that automatically builds a user profile, which is stored locally at a user's client and is under the control of the user. The profile consists of information about the most common tasks a user performs, such as their Web usage and content, conventional email content, the software they use frequently, etc. The profile is thus simply a processed form of data already on the client.

SHOCK clients connect to each other through a peer-to-peer network or through conventional e-mail transport. SHOCK messages can be of many types, are easily defined and extensible, and are composed through a simple HTML interface. They contain an XML-encoded computational component that is executed automatically against the profile at every receiving SHOCK client. The result of that computation determines whether the message is presented to the user. The computational component also allows the messages (and responses to them) to be structured in specific ways to, for example, allow automatically tabulated polls and surveys. The system has both a Web interface as well as an interface tightly integrated with Microsoft Outlook.

The SHOCK system also has a number of features that were designed to allow the most flexible kinds of messaging and conversation. For example, users can have truly anonymous conversations over the peer-to-peer transport using a message laundering scheme and built-in transparent encryption, and groups of targeted users can participate in threaded *ad hoc* conversations easily. Additionally, the decentralized nature of the system makes the marginal cost of implementation very low since it makes use

of otherwise idle resources such as client storage for profiles, and client processing for indexing and message matching.

The resulting system provides the building blocks for a number of interesting applications beyond simple targeted messaging. For example, users spend more and more time interacting with Web-based repositories of data and information (especially within enterprises). Their behavior and usage of these resources reflect on how valuable the resources are as well as the interests and expertise of the users. Additionally, users within organizations possess a great deal of non-public valuable information contained in their email conversations, their personal collections of files, etc. A system like SHOCK can allow some of the value in these hidden resources to be unlocked through the construction of higher-level applications built with SHOCK's basic functionality. Since profiles are automatically created, never published, with access always permissioned by the owner, these hidden resources can be leveraged without violating users' privacy.

In what follows, we describe the system in detail, provide examples of useful applications, discuss our experiences with a pilot field test, and consider related work.

2. SYSTEM DESCRIPTION

The SHOCK system was designed with two primary criteria. The first, detailed and timely user profiles, was required to provide the infrastructure for the targeting of SHOCK's computational messages. The second criterion was the reduction of work required by the user in order to increase participation. These traditionally opposing demands were satisfied through the use of a peer-to-peer design. In contrast to filtering solutions that rely on central servers ([8][13][19] and [21]), SHOCK utilizes a sophisticated client that ties more closely into user's activities and messaging habitat.

Beyond providing strong privacy controls, local profile construction allows the client to monitor a larger set of user activities. While a user's Web surfing behavior can be obtained by analyzing a proxy server (or sniffing the network), and email data can be tapped by tying into the email server, other information is not available in any central location. For example, which programs are installed on a user's machine is not easily found in any central location. The SHOCK client, by running locally, is therefore able to generate a more detailed user profile, allowing for sophisticated targeting of messages.

Additionally, a centralized solution has a number of other disadvantages. User control over private data would be eroded by this transfer to a common location, which would also become a central point of failure (in terms of privacy and security as well as performance and availability). The central store would also require considerably more resources in terms of storage and processing power, whereas SHOCK's decentralized solution utilizes existing resources.

In the following section we will briefly cover the design of the dedicated SHOCK network, discuss how this messaging system can be integrated with an existing email infrastructure when not all users have the SHOCK client installed, as well as how automated responses to SHOCK messages are generated from existing electronic repositories.

2.1 The SHOCK Network

While the specific SHOCK network architecture is not crucial to message targeting we briefly discuss this dedicated mode of

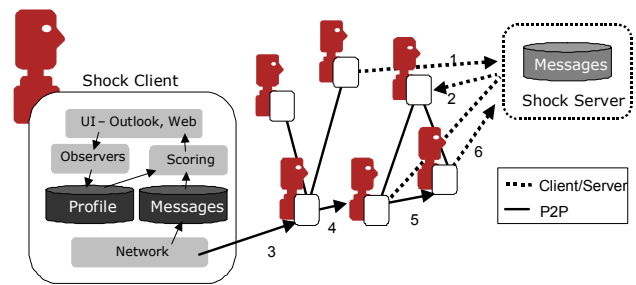


Figure 1. The SHOCK system architecture

operation. The architecture functions to connect the SHOCK clients in way that minimizes the need for centralized, costly computational resources and provide all the means for flexible messaging that includes the possibility of truly anonymous messages. A more complete description is available in [2].

The SHOCK network functions in two ways: pure peer-to-peer and hybrid peer-to-peer/server mode. The pure peer-to-peer implementation does not rely on a server for any messaging functions. When any network level message is sent it is broadcast among the peers (the SHOCK clients) until all clients have received the message. A pure peer-to-peer solution provides the benefit of not requiring dedicated server configuration and resources. However, a number of scaling and message persistence issues make the hybrid solution more attractive.

The hybrid implementation we use a simple server that buffers all messages and draws inspiration from the Crowds system [20] for message delivery. The right side of Figure 1 represents a possible network configuration. When a client sends a non-anonymous message they may simply deposit it at the server (message 1). Other clients will occasionally connect to the server and collect these messages (message 2, for example). To send an anonymous message the SHOCK client will send the message to a peer chosen at random (message 3). That peer will then randomly decide whether to pass the message to another peer or to the server. In this way, the message becomes "laundered" through the SHOCK network (messages 4 and 5), before finally arriving at the server (message 6). Through this mechanism, a SHOCK user can have the confidence that neither another SHOCK client nor a SHOCK server will know with certainty if the message originated from his or her client.

A number of SHOCK's features can be built using standard e-mail transport mechanisms. However, the use of a custom SHOCK network and server enables the provision of more efficient message propagation and scalability (through the use of buffering, intelligent routing, hierarchical network organization, etc.). Such features are difficult to build by retrofitting existing standards.

2.1.1 The SHOCK Client

The SHOCK client is implemented primarily in Java but it contains components to closely tie it into the Microsoft Outlook client, the dominant email client in our user environment. The SHOCK client locks into the Outlook application, adding folders for received and sent questions, and adding a toolbar for the most common SHOCK actions (asking a new question, replying, etc.). SHOCK messages are displayed as standard HTML format email messages. The client, functioning as a personal Web server, both generates and processes the HTML pages. Users may move, delete, and respond to these messages just as they do for ordinary

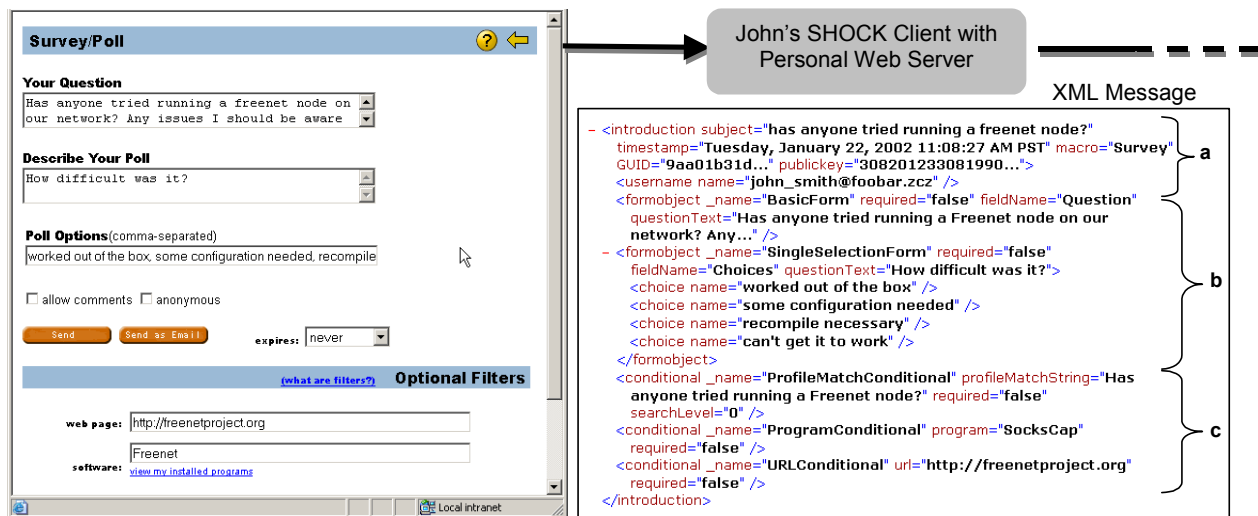


Figure 2a. A user, John, fills out a form on the SHOCK interface to find someone to help him with his Freenet installation. The form, submitted to John's SHOCK client, is transformed into an XML message and... (continued in Figure 2b)

email. Users who do not have Outlook installed or that prefer another interface are able to use a Web interface.

The left side of Figure 1 depicts the various components of the SHOCK client. Other than the interface components previously described, the client also contains a set of *observer* modules that track various user activities and store those in the profile. Currently, observers exist to store Web and email activity as well as installed programs and individual information from the enterprise directory (department, manager, location, etc.). The system was designed to allow for new observer modules to be integrated easily into the system.

SHOCK does not suffer from security concerns in the same way that other computational mail and mobile code systems do. This is due to several features of the system and the environment in which it operates (corporate intranets). Computational elements in SHOCK are designed to never automatically release information to the network. Once a message arrives at a client it is filtered and rendered it is up to the user to determine if and how to respond.

2.2 Messaging

The SHOCK network allows for two types of messages: *Introduction* and *Response*. Both messages are represented in an XML format. In order to start a "conversation" or broadcast an announcement a user will send an Introduction message. Any future responses to that Introduction will be through a Response message. As illustrated in the example Introduction in right hand side of Figure 2a, Introductions contains three types of fields: *general headers*, *conditionals*, and *form objects*. Response messages follow the same general structure but in the current system do not contain conditionals. General headers (labeled *a* in the figure) specify the message's subject, time stamp, a globally unique identifier (GUID), and an identifier for the user who sent the message (this may be "anonymous"). Messages contain a public key that may be used to encrypt the response. Introductions may also include an expiration date that indicates when a message no longer needs to be processed. In the particular example above, the user *john_smith@foobar.zcz* wants to know about running a *Freenet* node.

In order to group responses and threaded discussions, Response messages contain two additional fields indicating the GUID of the message to which this is a response (responses to responses as well as introductions is supported) as well as the GUID for the general thread (this must be the GUID of an Introduction message).

The elements in the second section (labeled *b*) are the form objects, which specify the fields to be filled in for a response as well as validation rules. In the example above, SHOCK interprets the user entries at the top of the form (under Survey/Poll) to determine that there are two fields that will be rendered and displayed to the user. The first, a *Question* field, allows the respondent to return a free-form answer to the question(s): "has anyone tried running a Freenet node? Any issues I should be aware of?" A response to this field is not required. The second field, a *SingleSelectionForm*, is rendered to allow a user to select one option from a list (worked out of the box, required some configuration, etc.).

Finally, the conditional section (*c*) specifies scoring and filtering rules for messages. By selecting various criteria in the "Optional Filters" portion of the interface, the asker may specify specific conditions that must be satisfied for the message to be displayed to potential respondents. Most email clients support filtering rules on the receiving end. Conditionals can be thought of as filters specified at the sending end. The Introduction above specifically requires that the user must have the program "Freenet" installed (the *ProgramConditional*) and their profile should score high enough on the question (*ProfileMatchConditional*). Realizing that not everyone was successful in installing Freenet, John also expands the message targets by asking for people who have visited the Freenet website. This is done by means of a *URLConditional*.

2.2.1 Generating Questions - Macros

In order to satisfy the SHOCK usability design criteria, we opted to build the message creation interface around specific tasks and in such a way that the details of the XML generated as output was abstracted from the user.

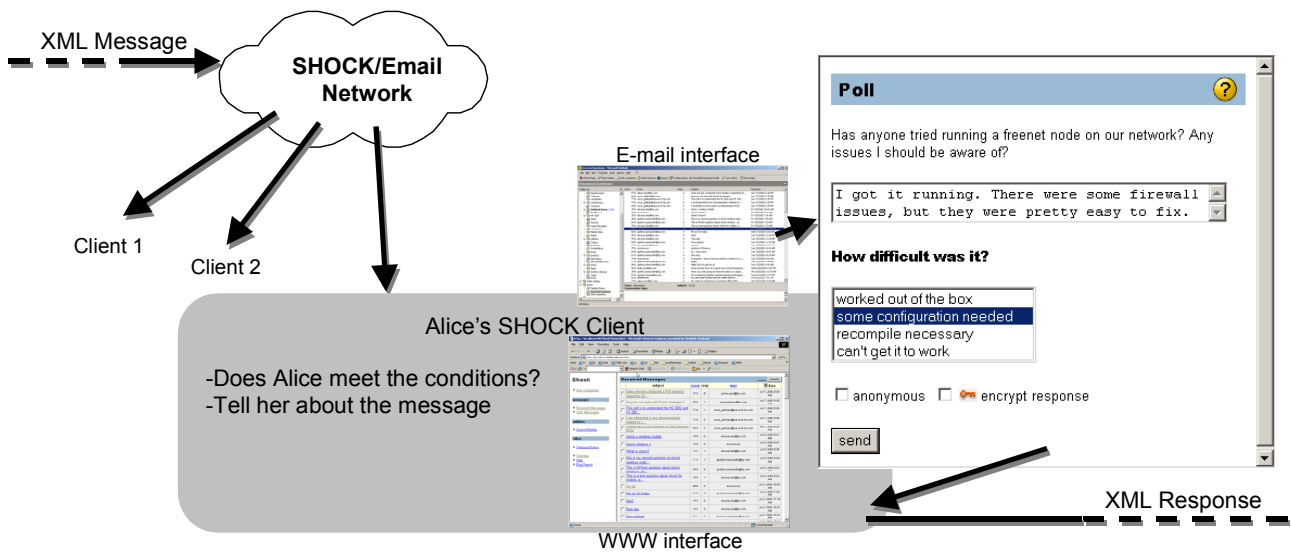


Figure 2b. The message is propagated to other SHOCK clients through the SHOCK and e-mail networks. Alice's SHOCK client receives the message and because Alice meets the conditions she is shown, and responds to the message. The validated response is sent ... (continued in Figure 2c)

Macros in SHOCK are simple HTML screens that allow users to rapidly define key fields (which will become form objects) and filtering criteria (which will become conditionals) around specific tasks. In the case of our Freenet user, John, is interested in a survey type question. The left side of Figure 2a is the macro screen he will see for asking that type of question. An alternative macro, as depicted in Figure 3, provides a quick way for users to generate a software announcement. Software announcements are intended for individuals in an Information Technology role who wish to target users by installed application. For example, a virus warning for a specific Web server can be sent through SHOCK so that only users with that Web server will receive the message. As we will later discuss in the applications section, depending on the tasks that are frequently pursued in an organization or group, different macros can be created to specifically address the needs of users trying to complete specific task and will expose different SHOCK features.

Behind the scenes, the SHOCK client will take output from the macro forms and generate the appropriate XML Introduction complete with conditionals and form objects.

2.2.2 Filtering Messages

Once introductions are broadcast through the system, each potential respondent computer must score the message for relevance. If the message's score exceeds a certain threshold set by the user, the message will be displayed in the user's interface.

Conditionals currently come in two main varieties, boolean and fuzzy. Boolean conditionals are either satisfied or not, returning a 1 or 0. A fuzzy conditional can be partially satisfied and is scored between 0 and 1 (inclusive). Furthermore, conditionals may be required or not required.

While each conditional is represented in the XML their evaluation is implemented at the client by a corresponding conditional object implemented in Java. By accessing different parts of the user's profile, the conditional object will determine if, and to what level, a conditional is satisfied. The output of each conditional object is

then combined as described below. The system was architected in this way to allow for the rapid creation of new conditional types. New conditional objects can be quickly implemented and will automatically be invoked when required through Java reflection. This allows for the dynamic updating of SHOCK clients to handle new filtering rules.

2.2.3 Fuzzy Matching

Due to its flexibility, the profile conditional is one of the most frequently used. Almost all macros automatically convert the text of the question to a profile conditional which is then compared to the recipient's full-text profile.

Documents, or specifically Web pages and emails in the current implementation, that a user accesses are automatically indexed in a full text index. The question text is then used to search the full text index for likely matches. Each matching document is then independently scored (using standard TFIDF [23] metrics) against the question text and the results are combined and normalized.

Because SHOCK clients do not have a global view of expertise (clients cannot compare one user absolutely to another), the profile conditional attempts to model the likelihood of a user's interest in a question based on the number of matching documents. SHOCK users may also declare profiles, explicitly indicating expertise and interests, and the question text can be compared to that information as well.

In the Freenet example, Alice's SHOCK client (Figure 2b) determined that based on the question and the text in Alice's profile, she may be a candidate to respond.

2.2.4 Boolean Matching

The basic SHOCK system contains three boolean conditionals that allow targeting of users who visited specific Web sites, have matching fields in the enterprise directory (department, location, etc.), and who have emailed a specific domain or user. Abstractly these are very similar, so we only discuss one in detail.

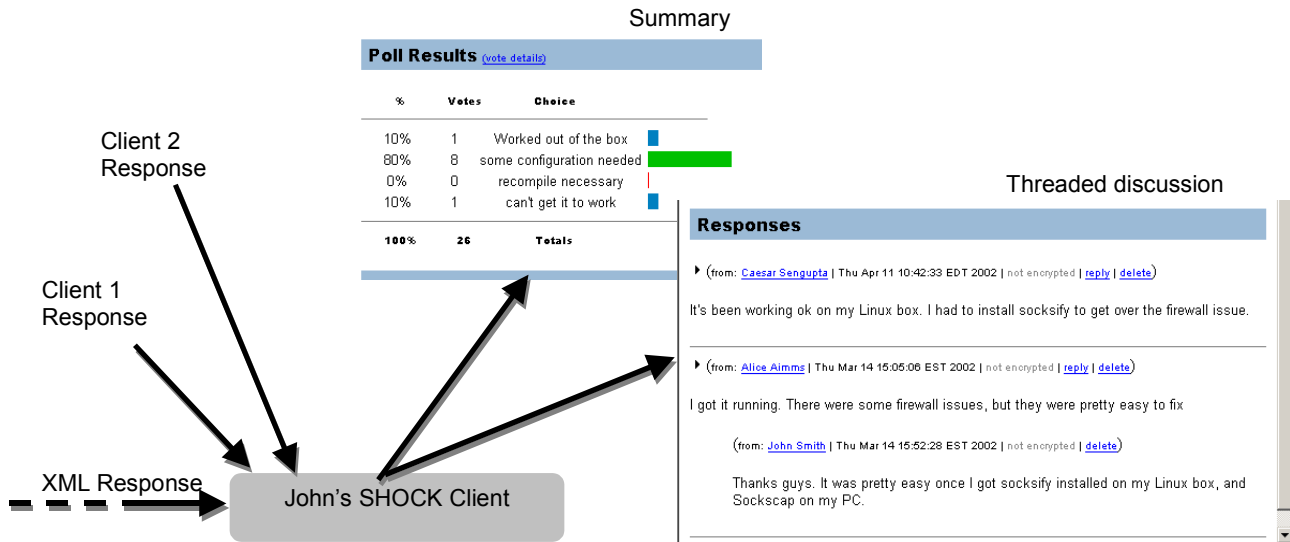


Figure 2c. ... along with other responses. John's SHOCK client (and others that are interested) can tabulate and format the results. The client generates both a summary graphic as well as a threaded discussion.

John's question contains a URLConditional which indicates that the user should have visited a website (although in this case it is not required). The module in Alice's client that evaluates URLConditionals will decide if she has visited the Freenet site, returning a 1 if she has, and a 0 otherwise.

In the future, we hope to extend boolean queries to allow users to specify that the recipient *not* have certain characteristics (e.g. "Please look at page x if you haven't seen it yet and tell me what you think."), and employing recency and frequency (e.g. "users who often visit Web site x"). The SHOCK client has easy access to all of these data.

2.2.5 Scoring

As previously discussed, a message's total score is the combination of the individual decisions of each conditional object. While in the future we may allow the macro designer or the user sending the message to specify the formula for combining scores, currently the scoring mechanism is set. The current system determines the number of satisfied boolean conditionals and the number of satisfied fuzzy conditionals and then combining the results.

The scoring mechanism also takes into account whether conditionals are required or not. If all required conditionals score above 0, the combined score of all conditionals is compared against the threshold (otherwise the message is filtered out since a requirement was not met).

We are currently experimenting with alternative scoring mechanisms including manipulation of scores not only in response to local scores but global behavior. For example, questions for which answers are observed on the network will have their score reduced (multiple users need not answer the same question). Alternately, questions that receive no answers may have their scores boosted.

2.2.6 Rendering Messages

If a message has been received and filtered, if the recipient chooses to view it, the SHOCK client will transform the XML form objects into a standard HTML form. As mentioned above, in the current system, rendered messages and responses are made available to the user through their mail client or through a Web interface. SHOCK, when installed on a machine with Outlook, automatically creates a standard mail message containing the rendered HTML inside. Both modes are illustrated in Figure 2b.

There are various form objects which map to different types of HTML form elements. For example, a *BasicForm* is a simple text

Figure 3. A software announcement macro.

entry box (TEXTAREA), whereas a SelectionForm is rendered as a SELECT/OPTION list. Because messages are tagged with the macro type that generated them, developers can also define specific views for different message types. Currently, this is done by programming a Java Servlet. We hope in the future to make this possible through standard mechanisms such as XSL.

John's question, which Alice has chosen to respond to, contains both a free text entry and a selection. Alice fills in the details as appropriate, and ships them back to John. Notice that Alice is able to encrypt her response so that only the sending user (in this case John) will be able to read the response. Just as Introductions can be sent anonymously, so too can responses.

Responses are also rendered intelligently. In Figure 2c, John's client collects the various responses. Because the client is aware that the question was a poll, it will automatically (and dynamically) generate a summary screen with a bar graph representing the various results. Additionally, all text responses are rendered in a threaded discussion allowing John and others to follow the conversation.

2.3 Email Integration

Despite the rich features of SHOCK we fully anticipated that not all users would initially choose to install the client. To address this, SHOCK was designed to function over standard e-mail. Next to the "send" button in the interface is a "Send as Email" button (see any of the macro Figures – 2a, 3, or 5).

A SHOCK message sent this way is embedded into a standard email message that can be sent to any user. Specifically, the XML is hidden as a comment tag in the HTML message. When the message arrives at a user's mail client, the SHOCK client recognizes the presence of the hidden XML instructions, extracts those, and then proceeds as if the message arrived over the SHOCK network.

Those users who do not have the SHOCK client will see this email as a regular mail message that contains the question/announcement and a small icon indicating that the message is SHOCK enabled with a link to the SHOCK website. Figure 4 illustrates such a message for the software announcement example (Figure 3).

This feature allows SHOCK to function as a way to send computational email over the ordinary email transport. Additionally, it encourages the growth of the user base, and can also facilitate the explicit creation of groups since only users who received the first message will be aware of and be able to participate in that message thread. The SHOCK client can also fold in standard e-mail responses to SHOCK messages automatically.

Although at present there is no way for users to send messages anonymously over e-mail we envision allowing users to regain this ability in the future. For example, the system may launder the SHOCK message over the SHOCK peer-to-peer network (as previously described) and having the last recipient e-mail the message to the destination on behalf of the original sender and broadcast responses back through the network.

2.4 Automatic Answers: "Robots"

The computational messaging abilities of SHOCK extend beyond the client software. We also designed and developed a set of automatic answer-generating services linked to corporate resources and databases, which we call *Robots*. These services listened for incoming messages just like any other SHOCK client,

and compared the message body to its own repository of information; if it could generate a meaningful response, it would send it out, just as would a SHOCK respondent. Such responses were marked as coming from a Robot.

One Robot was linked to an existing corporate expert-finding system. Incoming SHOCK messages were fed into the system, and the Robot would generate a message containing information about any matching experts that were found. A second Robot provided an interface to the internal company Web search engine, creating a SHOCK response with the results of a search on the message's contents. In these ways, we were able to combine SHOCK with pre-existing company resources without requiring any additional work on the part of the user. Augmenting the knowledge base covered by these services requires only the implementation of additional Robots.

3. APPLICATIONS

The capabilities of the SHOCK system allow a number of interesting applications. As described above, new types of SHOCK messages can be easily built out of simpler components to form macros, whose HTML rendering can be controlled appropriately. Specific macros are thus defined for specific usage scenarios that depend (typically) on the characteristics of users, their work practice needs, and their organizational context. In what follows, we break down the types of applications in increasing order of complexity and specificity to suggest the wide range of usage scenarios that the SHOCK system can be useful for.

3.1 Simple Computational Email

The simplest form of a computationally enhanced message does not require interaction with a recipient's profile. For example, in the context of distributed decision-making, a SHOCK message containing a poll or survey macro can be sent to a pre-selected group of target recipients directly addressed by email and sent over standard email transport (or if sender anonymity is required, over the SHOCK network). Users who receive such a message will have it automatically interpreted by their SHOCK client and rendered appropriately. More specifically, suppose a user wishes others to vote on their favorite Java compiler, or as in the case of our example, the ease of installing the Freenet application. Such a message would be rendered as a bar graph and users can easily select their choice, resulting in a dynamic update of the graph of vote counts. Other examples are even simpler. Users can send out general topics for discussion, and recipients can have threaded

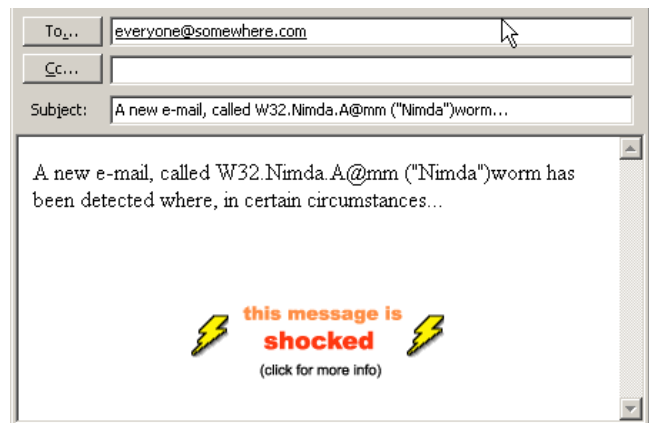


Figure 4. The e-mail message

group conversations easily (see Figure 2c).

The Active Mail system [18] provides a very basic mechanism for generating a message with a URL pointing at an automatically constructed Web page where an answer may be selected. The system provided encouraging evidence that simple computational email is useful by demonstrating that 55% of messages in a user study were transactional (asked a question), and 44% of those could be structured. Outlook itself provides a simple voting mechanism (yes/no button in e-mail). However, neither of these systems has the flexibility of SHOCK in designing forms and collecting data.

3.2 Simple Profile Targeting

These types of messages rely on computations performed against user profiles. For example, to continue the poll example from the previous section, rather than sending the message to a pre-selected and explicitly addressed group, the message could be sent to all users who have visited the Web page <http://java.sun.com>. The poll would only be offered to those who have, and no user's privacy is violated since the sender does not know who has visited the Web page unless a response is explicitly made.

Because the SHOCK profiler is located on the client, almost any potential user behavior can be targeted. The current implementation, in addition to Web usage, profiles sent and received email, software installed, as well as corporate directory information that includes geographical and organizational location. Thus, examples of other types of simple targeting include sending a message to users who send email to person X, or who work on floor N of building Y in country Z, or people who work in organization X, etc.

In large organizations where information overload through email can be a problem, simple profile targeting can be used to avoid broadcasting messages by choosing the right targeting criteria. For example, announcements related to certain Web-based database services or scientific software packages can be sent only to people who use them. In addition, the threshold variable set by the user can help tune the amount of information received this way.

3.3 Knowledge Management Applications

One of the motivations for developing the SHOCK system was to build a simple but general and flexible messaging system (analogous to email) that could support a variety of knowledge management applications. The ubiquity and resilience (in functionality) of basic email suggested that the SHOCK system would benefit by tight integration with existing email clients—email is many users' habitat [9]. Users spend more and more time using email and interacting with Web-based tools and repositories of information. Creating a messaging system with low participation costs for users can help unlock the value of user's electronic trails, while preserving privacy, for knowledge management applications in large, distributed organizations.

Since knowledge management is a very broad term, we here sketch several important application areas (which are not necessarily independent) and provide concrete applications within them.

3.3.1 Expert-finding

In large organizations of knowledge workers, finding experts in specific topic areas is an important problem. Various centrally

controlled solutions have been proposed in the past (see [1][12][17][22], for example, and [2] for others). The SHOCK system can be used to help solve this problem in several ways. We defined a SHOCK "Find An Expert" macro (see Figure 5) in which the asker is required to enter some descriptive question text about what they seek expertise in. Based solely on this question text, receiving clients can score the message against their profiles using IR techniques mentioned earlier, and only present the message if the score exceeds a user set threshold score. We denote this method of scoring as a "soft" method. Users can increase their threshold in order to reduce the number of messages they receive, allowing only the most relevant messages to be presented to them.

Alternatively (or additionally) such SHOCK messages can specify additional "hard" targeting criteria that must be met. Essentially, this allows those who are seeking expertise to define exactly what it means for someone to be an expert. As a concrete example, suppose a consultant in a large technology company seeks to communicate with people who have worked in the area of electronic banking, an area of new interest for one of her clients. The consultant may have tried first the online Web-based repository of supporting materials on electronic banking but seeks more. The user could send a SHOCK message with the targeting criteria that would only present the message to other users who have viewed the same Web documents on electronic banking as identified by the intranet URL. This can be accomplished trivially in the SHOCK system, whereas the alternative way would require modifying the database itself.

3.3.2 Knowledge reuse

Another important problem in large organizations is encouraging knowledge reuse. Knowledge workers [4] often find themselves "reinventing the wheel". For example, in investigating the deployment of SHOCK in a global pharmaceutical company, we learned that researchers often begin studying chemical compounds that have already been studied without their knowledge. This

Figure 5. A generic expert finding macro with all conditionals available.

pharmaceutical company had a naming system for compounds, and thus it would be relatively easy to define a SHOCK macro that would seek out others in the organization who had been emailing or reading Web pages about compound X. This example also illustrates the potential value of developing custom profilers, depending on the context.

3.3.3 Relationship management

Still another important problem is relationship management. Consider again a consultant who wishes to find others in the organization who have communicated with people at a potential customer in the past. The consultant can send a SHOCK message that seeks out others who send email to the domain (e.g., “ibm.com”) of that customer.

4. EXPERIENCES WITH SHOCK

We have described some of the applications made possible by the SHOCK system. In this section, we will discuss some actual examples of its use in a pilot deployment within our company, as well as the results of a user study conducted on the participants in our pilot.

4.1 The SHOCK Pilot

To learn about user perceptions of SHOCK, we ran a pilot study within our company. 47 people became “active” SHOCK users, sending or responding to at least one message. To augment our findings from the pilot, we conducted surveys of SHOCK users and non-users before and after the pilot, as well as one-hour interviews with 15 users.

As described previously, we envisioned several types of scenarios in which SHOCK would be a useful expert-finding tool. Some of these scenarios materialized in our pilot study, some did not, and we also saw some uses we did not expect.

One of the more popular features of SHOCK is its ability to support real-time, expertise-specific polls. We saw polls sent on a variety of topics. Polls ranged from very simple (one user targeted a message to colleagues on the same building floor about their opinions on the temperature in the area), to complex polls about sensitive company topics (controversial business decisions), and technical issues (product definition).

SHOCK users also described its value in situations where they had a question on a specific topic, but did not know who to ask. For example, a support technician used it to connect a client with a relevant expert: “I had asked about a need to find a firewall for a team that I’m working with... I was able to actually get the team that needed this, that had questioned me for help, I was able to hook them up with this expert. That was helpful.”

The initial pilot was partially intended to inform us on how best to design macros and streamline SHOCK into the work practice of users. Additionally, it served to educate us on how users interact with and what they expect from a system providing privacy and anonymity features.

4.2 Responses to Privacy

Recent systems [15] and studies [3] have described the benefits of local profile storage, a sentiment echoed by potential users in our early discussions with them. Local storage provides an added level of privacy, security, and control of a user’s profile.

SHOCK targets its messages using detailed (and potentially sensitive) profiles of users’ knowledge. Our privacy model is that

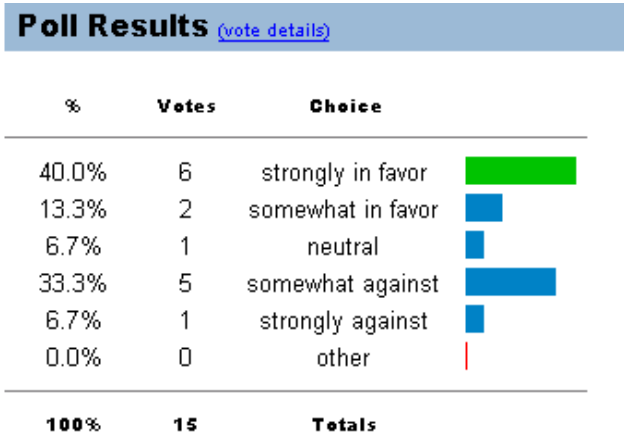


Figure 6. Results of poll on a controversial topic.

▶ (from: [anonymous](#) | Tue Jan 29 12:48:07 PST 2002 | not encrypted | [reply](#) | [delete](#))

I have concerns about our ability to execute this very ambitious plan.

Figure 7. An anonymous controversial opinion.

a user’s control is in the decision to respond, rather than in deciding which data was collected, as the statistical complexities of an indexed profile are hard to represent clearly to a typical user. To protect user privacy, SHOCK stores its profiles locally to a user’s computer, and provides the ability to delete or manipulate a profile.

In our study, however, we anecdotally observed that local profile storage was not important to our users. Despite providing profile management features, we found that users rarely stopped the automatic profile building, deleted their profile, or removed specific elements from a profile. To test this observation further, we devised a survey that simulated a request to participate in a knowledge-sharing system, asking the participants what information they would be willing to provide. Each participant randomly received one of six scenarios describing a SHOCK-like system. There were two variables: Who built the system (your employer, your coworkers, or a contracted 3rd party), and where the profile information would be stored (centrally or locally). From the 298 company-wide responses, a t-test showed that the difference between local or central storage was not significant. Free-form responses confirmed that the location of a profile is not an important consideration for most users.

On the other hand we also noticed that whenever Shock was explained to users, almost always the first question would be about the privacy of their information. It may be that users desire privacy controls, but having them is not critical. Additionally, these findings may be specific to the particular environment in our company. It is possible that in a less trusted environment, such as two partner companies sharing a SHOCK network or the Internet at large, users would exhibit a stronger preference for local profile storage, and may show more enthusiasm for the ability to control their profiles.

4.3 Responses to Anonymity

We also wanted to observe the ways in which people made use of SHOCK’s facility for anonymity. In a trusted corporate setting,

there are sometimes sensitive topics or embarrassing questions that people may wish to bring out anonymously. For example, one user posted an opinion poll on a controversial decision being made by the company (results shown in Figure 6). Not only did many users vote on the poll anonymously, but a few also made anonymous comments on the topic (one example shown in Figure 7). Interestingly, of the six (out of 15 total) negative poll responses (either “somewhat against” or “strongly against” the proposed company action), all were made anonymously. Six of the nine positive/neutral responses were anonymous. By contrast, overall the vast majority of SHOCK messages were posted non-anonymously. In this case, the ability to respond anonymously was important to many people, especially those taking a more controversial position.

However, we also observed ways in which people were wary of anonymous messages as being potentially untrustworthy. One user said, “Personally, I don't like anonymous [messages] because I think if you're going to post something you need to be accountable for it.” In addition, many of our respondents believed that within the company, personal relationships are important enough that anonymous questions are not useful.

5. RELATED WORK

Earlier work in the area of computational email, such as the Andrew Message System (AMS) [6], Information Lens [16] and its successor ObjectLens [14], Active Mail [18], and AtomicMail [7], focused on adding useful structure to ordinary text email messages. For example, in AtomicMail, users might receive a message that would present a scrollable list of document titles the sender was trying to distribute. A recipient who selected a title would be presented with abstract of the document, and asked whether she wanted to order it. Orders were sent to a special email address. Generally though, these systems tended to use computational elements to facilitate a variety of tasks such as voting messages, return receipts, group management, and meeting scheduling. Many of these functions are now part of standard email clients such as Microsoft Outlook, and HTML messages have made email a richer medium.

The FLANNEL [5] system allows for messages to be tagged with computational rules that cause the modification of e-mail *in transit*. For example, a “translate” rule would cause the server to transform the e-mail from English (the language of the mailer) to French (the language of the recipient).

The SHOCK system differs from much of this prior work by allowing computationally enhanced messages to interact with detailed user profiles at the client. This capability extends the range of applications significantly beyond increasing the structure in messages to address creeping problems of information overload and knowledge sharing. The concept of message targeting requires careful attention to the privacy issue which the SHOCK system treats as a primary design constraint. Another important design issue was making the participation cost for SHOCK users simultaneously almost zero. Therefore, integration into standard email clients to minimize disruption is an important design goal as well, unlike systems such as ObjectLens.

SHOCK's collaborative messages, which allow threaded conversations in a space shared effectively by *ad hoc* groups are also related to systems such as Active Mail [11] and the commercial system Zaplets [24]. For example, Zaplets are essentially HTML forms that groups of users can receive as email.

Since the email is really a Web page, users can use that Web page as a shared space to schedule meetings, make group decisions, etc. However, Zaplets do not interact with information specific to any user.

In the area of knowledge management, SHOCK is concerned with many of the issues taken up by systems such as Answer Garden [1], and the ER system [17]. SHOCK differs crucially from these systems in its approach to privacy and its decentralized architecture. SHOCK shares some aspects of its architecture with Yenta [10], but differs in the types of tasks and applications it aims to support. A more detailed comparison between SHOCK and such systems is provided in [2].

6. CONCLUSION AND FUTURE WORK

In this paper, we presented a flexible, low-cost peer-to-peer messaging system that allows users to send novel kinds of targeted messages, participate in threaded conversations, and send structured messages. In contrast to traditional explicitly “addressed” electronic messaging systems such as ordinary email, instant messages, and voice telephony, the SHOCK client profiles allow messages to be sent to other users who satisfy a wide range of specific criteria including Web browsing history, ordinary email content, local software usage, etc. This is accomplished by the definition of computational components embedded in the messages and the local client profiler, which allows user privacy to be retained while making use of otherwise idle computational resources. Recognizing that email is often users' “habitat” we emphasized the ability of the SHOCK system to integrate with standard email clients, and showed how the system can use its own peer-to-peer network transport when sending messages, or use ordinary email transport when sender anonymity is irrelevant.

The SHOCK system is a general and flexible system that can be used as building blocks for novel kinds of applications that take advantage of its targeting capabilities. We sketched several example applications. Our pilot study in a large, globally distributed technology company suggested some of the ways in which the system was useful as well as some of the potential problems (many of them social and cultural) that need to be addressed in order to exploit such technologies to help solve creeping problems of information overload, expertise location, and knowledge sharing.

In future work, the SHOCK system can be used as a platform for building applications that require the solution of interesting technical problems. For example, the distributed private profiles contain data that is typically unavailable to other users but may be very valuable to them. Especially within enterprises (despite the growing reliance on internal Web pages for the dissemination and storage of information) users have little incentive to publish useful information. SHOCK provides a means of unlocking the value in that hidden information. That task would be made easier and more attractive to users by the use of sophisticated cryptographic techniques that allow information about aggregates to be computed while maintaining privacy [8] and with zero cost to users. Another important area of future work is the implementation of reputation and reward systems to help create incentives for users in more sophisticated applications. Finally, we are interested in improving SHOCK's scalability through the use of controlled network topologies and intelligent server design.

7. ACKNOWLEDGMENTS

We thank Marie-Jo Fremont, Nathan Good, Bernardo Huberman, Lada Adamic and the pilot, survey, and interview participants.

8. REFERENCES

- [1] Ackerman, M. and Malone, T. Answer Garden: a tool for growing organizational memory, *ACM SIGOIS Bulletin*, 11(2-3):31-39.
- [2] Adar, E., Lukose, R., Sengupta, C., Tyler, J., and Good, N. Shock: A Privacy-Preserving Knowledge Network, *Information Systems Frontiers*, 5(1):15-28.
- [3] Agrawal, R. and Srikant, R. Privacy Preserving Data Mining, in *Proceedings of the 2000 ACM SIGMOD Conference*, May 14-19, 2000, Dallas, Texas, pp. 439-450.
- [4] Allen, T. *Managing the Flow of Technology*, MIT Press: Cambridge, 1977.
- [5] Bellotti, V., Ducheneaut N., Howard M., Neuwirth C., Smith I., and Trevor, S. FLANNEL: Adding computation to electronic mail during transmission, *Proceedings of the 2002 UIST Conference*, October 27 - 30, Paris, France.
- [6] Borenstein, N., and Thyberg, C. Cooperative work in the Andrew message system, *Proceedings of the 1988 Conference on Computer-Supported Cooperative Work*, September 26 - 28, 1988, Portland, Oregon, pp. 306 - 323.
- [7] Borenstein, N. Computational mail as network infrastructure for computer-supported cooperative work, *Proceedings of the 1992 Conference on Computer-Supported Cooperative Work*. November 1 - 4, 1992, Toronto, Canada.
- [8] Canny, J., Collaborative Filtering with Privacy, *IEEE Conf. on Security and Privacy*, Oakland, CA, May 2002.
- [9] Ducheneaut, N. and Bellotti, V., Email as habitat: an exploration of embedded personal information management, *Interactions*, 8(5):30-38, 2001.
- [10] Foner, L. Yenta: A Multi-Agent, Referral Based Matchmaking System, *Proceedings of the First International Conference on Autonomous Agents*, p.301-307, February 5 - 8, 1997, Marina del Rey, California, United States.
- [11] Goldberg, Y., Safran, M., & Shapiro, E. Active mail - a framework for implementing groupware, *Conference Proceedings on Computer-Supported Cooperative Work*, November 1 - 4, 1992, Toronto, Canada, pp. 75 - 83.
- [12] Kautz, H., Selman, B., Shah, M., Referral Web, *Communications of the Association for Computing Machinery*, 40(3):63-65.
- [13] Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., and Riedl, J. GroupLens: Applying Collaborative Filtering to Usenet News. *Communications of the Association for Computing Machinery*, 40(3):77-87
- [14] Lai, K. Y., Malone, T. W., and Yu, K. C. Object lens: a "spreadsheet" for cooperative work. *ACM Transactions on Information Systems (TOIS)*. 6(4): 332-353.
- [15] Lau, T., Etzioni, O., and Weld, D., Privacy Interfaces for Information Management, *Communications of the Association for Computing Machinery*, 42(10): 89-94, 1999.
- [16] Malone, T.W., Grant, K.R., Turbak, F.A., The Information Lens: An intelligent system for information sharing in organizations, *Proceedings of 1986 SIGCHI Conference*, Boston, MA USA.
- [17] McDonald, D. and Ackerman, M., Expertise recommender: a flexible recommendation system and architecture, *Proceedings of the 2000 Conference on Computer-Supported Cooperative Work*, December 2 - 6, 2000, Philadelphia, Pennsylvania, pp: 231 - 240.
- [18] Milewski, A.E. and Smith, T. M., An Experimental System for Transactional Messaging, *Proceedings of the GROUP 1997 Conference*, Phoenix, AZ.
- [19] Morita, M. and Shinoda, Y. Information Filtering Based on User Behavior Analysis and Best Match Text Retrieval. *Proceedings of SIGIR Conference on Research and Development*, July 3 - 6, 1994, Dublin, Ireland, pp 272-281.
- [20] Reiter, M. K. and Rubin, A.D. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security* 1(1):66-92.
- [21] Schwartz, M.F. and D.C.M. Wood, Discovering Shared Interests Among People Using Graph Analysis of Global Electronic Mail Traffic. *Communications of the Association for Computing Machinery*, 36(8):78-89.
- [22] Shardanand, U. and Maes, P., Social information filtering, *Proceedings of the Human Factors in Computing Systems Conference*, May 7-11, 1995, Denver, Colorado, United States, pp. 210 - 217.
- [23] Salton, G. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley, Reading, MA, 1988.
- [24] Zaplets. (2001). Zaplets. Available: <http://www.zaplet.com/>.