Developing Ideas Using Personal and Large Screen Displays

Shahtab Wahid
Department of Computer Science
Virginia Tech
Blacksburg, VA 24060
swahid@vt.edu

ABSTRACT
Notification systems are designed to inform users when certain events occur. The design of a notification system depends on the information it must keep track of and the manner used to alert a user. The challenge in this field of research is to determine what notification method is appropriate for any given situation.

BrainStorm is a piece of software that facilitates the development of projects by letting people develop ideas. It consists of a large screen display running a server and clients that work as notification systems. Users can post ideas and receive updates using the clients. The large screen display shows all the information that is posted by all the users. Its purpose is to initiate casual interaction among users and to be a reference during group meetings. BrainStorm combines characteristics of face-to-face communities with asynchronous distributed communities by using the large screen display.

INTRODUCTION
Over time people have developed a need to keep track of information that one may need. A notification system keeps track of changing information and informs the user of changes. The type of information needed and the notification method determines how a notification system must be designed. This is what researchers strive to find.

The range of information and notification methods is great. It is really hard to find a perfect notification system that can be the answer to such a broad range. When designing a notification system, one must first decide what data to notify.

Information needed by users can range from meeting times to the weather. The design of a notification system will greatly depend on the data being dealt with. Once this is determined, one must find the right way of notifying the user. The ways to notify a user are endless. One goal in this area is to be able to choose the right method.

Notification Methods
Many notification systems exist within our environment. The alarm clock has always been a simple notification system. It alerts the user at a certain time in the day. This simple device has become a useful system that has worked for many. It is designed to alert a user by trying to interrupt the task he/she may be doing.

A common example of a notification system used on a computer is the ticker. This is a system that is used to stream data such as news headlines. Users may leave such a ticker running so that they may check for changes in data while they work on their primary task. The ticker may be checking for updates after a certain period of time so that the user will be notified of changes. While the ticker may not directly inhibit one’s task, it may attract attention depending on the nature of its visual aspects. In most cases, the speed of the movement may cause a user to be distracted and keep looking at the ticker.

An example of a more intrusive notification system is one that alerts a user of incoming mail by popping up a message on the center of the screen. This quick and obtrusive method alerts the user immediately of the incoming mail by diverting their attention. Such a system prevents one from continuing their primary task. Whether a user will appreciate this or not will depend on the user priorities. The issue of priorities will be discussed later.

The notification systems described so far are usually used for a single user. Large screen displays can be used to show information that may apply to multiple users. Displays may be located in a common area where people are located. As people pass by the display, they can look at the information that is shown and even interact with others around them based on the information. An issue with large screen displays is the actual relevance of information shown. All the people looking at the display may not find the information useful.

Primary Task
The definition of a primary task can be debated. One can argue that a primary task is any task that is productive. Others can say that whatever one happens to be doing is their primary task. While this argument can go on forever, what one must really understand is that the
primary task may illustrate how effective a notification system really is.

To illustrate this concept, let us consider the following example. A user’s primary task is to write a paper using a word processor. During this time a notification system is running. The user is extremely focused on their task and does not wish to be interrupted by anything at all. A few hours later, the user’s primary task changes. While performing this new task he/she may not mind if the notification system interrupts their work. The challenge in such a situation is to find a notification system that will not take actions that are not desired.

It is hard to know how the user wants to be notified based on this primary task, but in such a case a system that does not make sudden visual changes may work well. An example may be a system that uses a fade whose rate of change can be controlled to show changes in data. A slow fade is advantageous because it does not attract as much attention as notification techniques involving rapid changes. The user may continue to write their paper using a slow fade without being attracted to it. When they change their primary task, the speed of the fade can be increased since the user doesn’t mind if their second task is interrupted. The system can be manually adjusted for changing priorities.

Such a design understands that a user’s notification needs change over time. The only problem is that the notification system must be adjusted by the user to conform to changes in the primary task. Much research is needed to develop a notification system that can interpret changes in the primary task. There may be a way to track changes in the primary task by monitoring the usage of input devices and applications running, but it will be hard to track changes in mental states. Whether a person is deeply focused on a task or not can’t be measured.

**BRAINSTORM**

There has always been a need for sharing information among people. Throughout time, there have been many developments that aim to facilitate this need; especially ones that have been designed for distributed communities. Many of these developments allow users to share a variety of information. Most notably, Saul Greenberg’s Notification Collage (NC) allows users share notes, pictures, and video. The NC’s goal is to promote general awareness for users. While the system has its advantages, one of the concerns of the users was the value of information [1]. The awareness provided contains a lot of information that is not needed by the user. BrainStorm tries to increase the value of information by decreasing the amount of information provided.

BrainStorm is a system that notifies users of progress made on projects they may be working on. Users can post information regarding their projects for others to see. A large screen display is also used so that users that are in a meeting may view the information posted. With these affordances in mind, the goals of Brainstorm are the following:

1. Ensure that the value of data is maintained for the user by letting the user choose what data he/she wants to see.
2. Notify the users when other users post new data.
3. Facilitate the development of ideas as users interact with shared data by providing an environment in which users can generate ideas.
4. Encourage users to meet regularly by using a large screen display.

The logic behind BrainStorm can also be associated with that of newsgroups or listservs. Users can post artifacts and make use of artifacts posted by others. These artifacts come in two forms: ideas and comments. An idea is usually information that is in the form of regular text. A comment is a note that is a reply to an idea. All ideas and comments go under a project making all them a subset of the project.

BrainStorm has a client-server model, which enables all the users to share their information. The administrator running the server will have all privileges. He/she will be able to manage projects and ideas on the server. While posting and deleting ideas are also allowed, he/she will be able to post, but not delete comments. An option to merge ideas will also be provided enabling the development of ideas. The server can also tag ideas as permanent so that they are not removed. An idea will automatically be removed if it has not been tagged as permanent and has been on the server for more than a period of time determined by the administrator. A client, on the other hand, will only have the ability to post ideas and comments onto projects they subscribe to. The client will be updated every hour and the user will be notified about updates.

**Visual Display**

The visual aspects of a display impact the users in numerous ways. In many cases, the interface is what makes or breaks a piece of software. At this point, there are two possible ways of displaying the information on the large screen display and clients. Testing will have to be done to determine which visualization will be appropriate for the users.

The first choice is to display the ideas in the form of a table (Figure 1). The table has columns for the ID of the person who posted the idea, time of post, permanent tag, title of the idea, and a short excerpt of the idea. Each row is dedicated to one idea. Clicking on one of these ideas...
allows them to see the full text for the idea and any comments for the idea. The advantage to using this type of a display is the efficient use of space. Users using the large screen display are able to see a preview of ideas and quickly be able to switch between ideas. On the other hand, it may be harder for users to see the full text of multiple ideas at the same time. A study on how large the excerpt should be may have to be done. By finding a correct size, we may be able to compensate for not being able to show the full text of multiple ideas at the same time. This type of a display will also work well with a search feature if it were to be provided. Search results can be easily displayed in this form as opposed to the next.

If this method is not used, it may still be combined with the previous choice. In the table form, the full text of an idea and comments are shown once a user clicks on an idea. This could open a new window with the idea, which could be designed like the Post-It notes. Combining both of these forms can combine the advantages together. Users can retain their overall view of all ideas, but be able to compare ideas by opening a few.

Besides presenting the ideas, the software must be able to handle multiple projects in some way. The large screen display will either show ideas from one project, or will have to be divided into sections to show multiple projects. If multiple projects are displayed at the same time, this means that there is decreased space for the ideas in each project. In this case, it may be better to use the table method since the Post-It note metaphor demands more space. It is better to use the Post-It notes if only one project were to be displayed at a time. To switch between projects, one can create tabs. Clicking on any tab changes the display to show the data of another project.

Underlying Design

There are two main aspects of the software one must find solutions to. They are the organization and storage of data and the design of the server.

For the storage and retrieval of data, there must be a format that is determined. For each project, there will be a separate file that contains all the ideas and comments. This is done so that the time to access data will be shorter. The filenames for the projects must be unique.

To distinguish between fields of data in a project file, there must be a set of tags that are used. One can easily parse through the files using the tags and find what is needed. The <idea> and <comment> tags will be used to show the start of an idea and comment respectively. The <id>, <time>, <title>, and <text> tags will be used for the
individual fields in an idea or comment. The following is an example of how the file will organize the data fields:

```xml
<idea>
  <id>nlobo</id>
  <time>04/12/02 13:43</time>
  <title>Graphics</title>
  <text>I've got a new layout for the graphics at my site</text>
</idea>
```

The full implementation of the server, at this point, has not been fully determined. Broadcasting information from the server can be done using an http server. Clients are able to connect to the server and parse through the files to update users. The implementation of submissions to the server is what must be worked on later. The server must be able to handle multiple users that may submit at the same time and must deal with simultaneous reads from and writes to project files. There may have to be a priority implemented between users submitting and users reading.

**Using BrainStorm**

At this point, it is predicted that BrainStorm will work well with groups of people ranging from ten to twenty. One person from this group will act as the administrator of the server. This person will create a project for the work the group is working on. Once this is done, members of the group will subscribe to the project.

Members will subsequently start posting ideas for the project using the client. Every idea acts as a contribution to the project. Others may post comments on the ideas that are posted to improve the quality of ideas. This stage resembles the process of brainstorming. As users consider and investigate a variety of ideas posted, solutions to problems they work on may be derived.

As updates are made, users are notified so that they may be able to follow the progress. The client checks for updates every hour by default. When users talk to each other at their own computers, the notifications can lead to further interaction and project development among group members. The method used to notify users has to be a subtle change; preferably one that may alert the user by making an icon change in the system tray. Such a method does not intrude one’s primary task too often. At times of high traffic, users may be interrupted too often if the notification method was aggressive. If a user wishes to check for updates at any point in time, they can instruct the client to do so. Users can change the client’s default time interval to check for updates to account for changes in primary tasks. Increasing the time interval means that users are busier and do not wish to be notified as often.

When there are a significant amount of ideas posted, the group may come to the point where they need to talk about the information and make decisions. This is facilitated by the use of a large screen display strategically located where the group can meet. During a meeting, the group members can use the display to look at all the information. Ideas and comments are evaluated and decisions are made. The ideas on the server may be merged to reflect changes. Ideas that are liked can be tagged as permanent, while others left to be deleted by the server in time.

The use of the large screen display greatly helps the progress of the work. The users can decide to meet and use the display to work. At the same time, the display itself may initiate a meeting with a help of casual interaction among users. For example, a few users who may be around the display may start talking about the work as they look at the display. This interaction may then lead to the individuals calling the other members to have an actual meeting based on the information on the display.

**Advantages**

The development of BrainStorm was based on newsgroups, listservs, and the NC. BrainStorm seeks to improve these systems and provide advantages over them by understanding the problems that they bring.

In most cases, newsgroups and listserv users do not have a common goal, although they may have a common general topic. The lack of a goal makes users post messages about anything. Many post questions or answers regarding certain issues, but they are not common issues. This type of a system does not provide an opportunity to develop a single idea. BrainStorm limits the type of posts one should make. By making users only post ideas or comments, it allows users to make progress. The biggest difference is the merge feature. This is not something that can be done in a newsgroup. Every merge that is done will show that there is progress being made within that project.

The number of people using a single newsgroup or listserv can be great. It is likely that with a larger group, subscribers may not personally know the other subscribers. This situation may not allow everyone to work together effectively and can cause some people to make posts that others may not find appropriate. It may also be hard to determine who is actually controlling a newsgroup. Because the groups using BrainStorm will be smaller and from the same general area, it is highly likely that most of the people will know each other. Because of this, the users using the system will know what an acceptable post is. In an academic or corporate setting, BrainStorm users will know who the head is, letting the users know who is in control. This limits the need for moderation.
The process of brainstorming will work better when users are posting ideas as clients [3]. This eliminates factors that can block someone from telling others what they think. There is less interference between group members. During meetings a proposed idea may be quickly forgotten. Posted ideas will remain for everyone to see and act as a record.

There are also advantages that BrainStorm provides over the NC. Users using this system were concerned about the screen space and the value of the information they were receiving. BrainStorm solves this problem by designing the clients so that they do not use all the screen space. This client can also be minimized so that it can continue running without using any space. The value of the information the users receive is also high because the user gets to choose what projects they want updates on. By letting users determine what they want, they increase the value of notifications received.

CSCW AND GROUPWARE
Computer Supported Collaborative Work (CSCW) is an area of research that tries to understand the social dynamics of people working together in groups. Researchers try to find the characteristics of working in a collaborative environment so that better designs for groupware can be developed. Groupware is the technology used to enable groups of people to work together using computers.

Social Dynamics
When many people use a system there are many social issues that arise. To understand the problem, let us discuss the social dynamics of a face-to-face community and an asynchronous distributed community.

In general, groupware fails for the following three reasons:

1. The disparity between those who do work and those who get the benefit [2].
2. The threat to existing power structures.
3. The lack of a critical mass of users

Asynchronous distributed communities are groups of people that interact from different places at different times. In systems of this nature, it is hard to tell who does the most work. There may be people who post information that is good and others who may take that information and use it as their own. Some social loafers [3], ones who don’t work as hard, cause others in their group to work harder to compensate for the work not done. Because users do not see each other, they may not feel that they are obligated to contribute to the system. In this case, people who do not work as much can get away with it and benefit from others using the system.

Such systems also do not always benefit all the users. Grudin uses a meeting scheduler [2] to express this problem. In this type of system, managers and subordinates maintain a schedule. This lets the system go through all the schedules of the subordinates and find a time the manager can call for a meeting. This system clearly benefits the managers. The subordinates have to do extra work constantly to maintain their schedule since the manager can call for a meeting at any time. This shows the disparity in groupware systems.

Groupware systems also treat all the users equally, destroying the existing power structures in a community. Because there isn’t a clear leader, people may say things they would normally not say if their superiors were around. As a result, this decreases the effectiveness of groupware.

Groupware can also fail because of the lack of interest. Users do not want to have to use an extra piece of software to do things they may do by using e-mail and the telephone. Many systems require a certain amount of users to start using the software before it become effective. If the system never finds the minimum amount of people, the people already using the system will also stop using it. This also makes it hard to evaluate a groupware system.

The closer a groupware development is to the characteristics of a face-to-face community, the better it works. In a face-to-face community, people work together towards their own goals. There can be some people who do not work as hard, but this problem is usually solved by social pressure. The social pressure in a group makes one feel more committed to the others and their work. If they do not do their work, the people may look down on them during meetings. Because of this, not many can get away by using someone else’s work as their own and try to do their best.

When it comes to power structures, the extent of relationships between people becomes vivid during face-to-face meetings. While superiors are clearly identifiable, they can also exert their power upon others. This allows the exercise of control and order.

BrainStorm is a groupware system that lets a community work as both an asynchronous distributed community and a face-to-face community. When users are alone and interact through the system, they work as a distributed community. Meetings in front of the large screen display establish a face-to-face community. By combining these two types of interactions together, groupware issues in social dynamics are adjusted and corrected.

BrainStorm is used among people who are in the same general area. These people will get together often to have
meetings, which have the same characteristics as face-to-face communities. There will be a certain degree of social pressure and a clear power structure. BrainStorm allows these characteristics to be passed on to the distributed community interactions. Because face-to-face interaction will be common, the asynchronous distributed interaction becomes more like the face-to-face meetings. Members of the community will know that if any of them try to avoid work while they are alone, they will face the consequences when they actually meet with the others. Casual physical interactions throughout the day will force users to maintain their work habits while they interact as a distributed group. Thus, the meetings initiated by members or by the large screen display actually control the social dynamics of a distributed community.

As mentioned before, the logic behind BrainStorm is similar to that of newsgroups and listservs. An idea in BrainStorm is comparable to e-mail. E-mail happens to be one of the success stories in the area of CSCW. People tend to share the benefits and burdens of using e-mail equally [2]. For this reason, BrainStorm has a good chance of distributing the benefits and burdens to all group members.

The large screen display also develops the critical mass of users that is needed for BrainStorm. If a few members start using the system and constantly refer to the information on the large screen display during meetings, more users will become inclined to using the system.

**SUMMARY**
The discussion has consisted of two parts: notification systems and BrainStorm. The challenge in designing notification systems is deciding what the right way to notify a user is. This will involve an analysis of the information being monitored, notification method, and user needs.

BrainStorm is a system that uses a client notification system and a large screen display. Both these notification methods are tied together in order to promote the development of ideas that people work on. In no way is BrainStorm a system that generates ideas for the people, but instead an environment in which ideas can develop.

It is thought that BrainStorm should work effectively as a groupware system. Face-to-face interactions between members act as mechanisms that make distributed interactions similar. Asynchronous distributed involvement in the system starts to gain a sense of social pressure and a power structure. The critical mass of users can be established by using the large screen display to attract more users.

A lot of testing will be needed to understand how people will really use BrainStorm. Tests may need to last over long periods of time since time is needed to watch how ideas may evolve. It would be interesting to find out if people end up treating BrainStorm like a listserv and post anything they want. Some may refute the system as a whole and use a listserv instead. This conclusion can depend on the performance of the large screen display since it is the key to interaction between people. In summary, BrainStorm is still an idea that must be explored in greater depths.

**REFERENCES**
2. Grudin, J. Why CSCW Applications Fail: Problems in the design and evaluation of organizational interfaces, ACM/SIGCHI, 1988