Collaborative Networked Communication: MUDs as Systems Tools

Rémy Evard – Northeastern University

ABSTRACT

A systems administration group is only as effective as its internal communication mechanisms. On-line communication has traditionally been managed via electronic mail or news, which are neither real-time nor truly collaborative. Communication tools which let multiple parties work together in real-time have become widespread on the Internet in the last several years.

In an effort to keep a physically disjoint systems staff working together effectively, we have explored the use of MUDs as communications tools. By allowing many people to interact in an extensible environment, MUDs have solved many of the problems that we had with on-line communication, and provided many unexpected benefits as well.

Introduction

Multi User Dungeons, or MUDs, are widely used on the Internet as interactive role-playing games. They use valuable network resources, attract unruly users, and are often run by students who didn't quite bother to ask the permission of the local authorities. As such, they are considered one of the banes of systems administrators, and perhaps rightly so.

Recently MUDs have been seen in a new light by some. While they are used most often as gaming environments, the software is in no way constrained to just that purpose. Instead, it is possible to program an environment in the MUD that is suitable for socializing and communicating. The MUD becomes a virtual ''place'' on the network where people can meet and collaborate on various projects.

In this paper, we present our experiences in using a MUD as a tool for improving the communications of our systems administration group.

Site Information

The Experimental Systems Group manages the computing environment within the College of Computer Science at Northeastern University, which is located in Boston, Massachusetts. The College operates approximately 300 computers of various types, including Macs, PCs, and a set of UNIX workstations consisting primarily of SPARCs and DECstations. About 800 users keep the computers, the network, and the systems group very busy.

The core group currently consists of five full-time staff members (two of whom are students enrolled in a cooperative education program) and several over-active student volunteers. In addition, eight to twelve students participate in systems-related projects each term of the school year as part of a volunteer program, and may put in as much time as the full-time staff (or perhaps more).

With this many people involved in systems projects, coordination and communication become essential to making effective progress. The work places are scattered around the building, many people work from home, and the students and hackers on the group tend to keep unconventional hours, so meeting physically in order to coordinate is not always practical.

Communication Needs of a Systems Group

In any organization, communication between its members dictates how well the group functions. This is true of a community of any size. Consider the havoc created when a telephone network ceases functioning, the problems attributed to lack of communication in today's families, or (sigh) what happens when electronic mail quits flowing.

Different communications tools are appropriate for different types of communication. The users on our network notify us of problems and make requests by sending electronic mail to ''systems''. We work with them individually either through email or in person. We use newsgroups for announcing changes which will affect everyone, such as impending downtime or new software installations. We also have newsgroups for open discussion of systems-related issues.

Internally, much of the systems group's coordination is done via electronic mail. (We have a separate alias that we don't share with the user community. This is enormously useful when combined with judicious use of mail filters.) Email within the group is nice for announcing future plans, sending notices about changes, or communicating nearly any non time-critical type of information. It provides a handy way to log changes, is extremely convenient, and is largely non-interruptive, so you may choose when to read your new email. (If you're on the group, you always have new mail.)

Despite its flexibility, we discovered that electronic mail was not appropriate for all of the types of communication that we needed. Time-dependent information is one example: sending email saying ''Is anyone in the machine room right now?'' just doesn't work if the person in the machine room doesn't read their mail while there. Further, it's not very effective for round-table discussions. While email can be used this way, we have noticed that the quoting mechanism commonly used in electronic mail can turn a potential brainstorming session into a nitpicking mechanism. Worse, it's not quite real-time, so the discussion will tend to die out, or become multi-threaded.

Because of the distributed nature of our group, we needed to be able to have on-line discussions and to communicate about real-time issues. Email wasn't working out well, and news clearly wasn't the right direction in which to move. Many programs available on the Internet provide interactive networked communication, and presented interesting possibilities. After some exploration, we decided to test a MUD for a few weeks.

MUDs

MUD stands for Multi User Dungeon (or, pretentiously, Multi User Dimension). The original MUD was written in 1979 by Roy Trubshaw and Richard Bartle on a DECsystem-10 at Essex University [Bart90]. It was a game similar to the classic Colossal Cave adventure, except that it allowed multiple people to play at the same time and interact with each other. Reportedly, this was enormously popular, and the idea caught on.

The original idea has evolved over the years into a client/server architecture. The MUD server manipulates the database of objects in the virtual world, is programmable in some sort of language that allows one to extend the set of objects, and accepts network connections from clients. The client's primary task is to send and receive I/O between the server and the user. The MUD server exists on one machine on the network, while the client is typically run by the users on their own machines.

Today, there are perhaps a dozen popular types of MUDs available on the Internet. They vary in many details, such as their embedded programming language and their storage methods for the objects they manipulate, but all have the capacity to allow multiple users to interact within some shared context.

Interacting with a MUD

The majority of existing MUDs are text-based. One types commands to them using a primitive English-like set of commands, and sees the results in a like manner. They are very reminiscent of the original Colossal Cave adventure.

For example, I could type:

```
look
```

and the response might be:

```
You are standing in a sunny, grassy
park-like area, under the sprawling
limbs of a weeping willow tree,
growing on the banks of a small
stream.  To the south you see a
university campus.  A trail winds
north into a forest.  Erik is here.
```

Technically, what happened is that I typed the string ''look'' to my client program. It sent that command to the MUD server, which parsed it (not very difficult in this case). The server then calculated the effects of my command on the objects in the server (which was to retrieve the description of my location), and sent the result back to my client, which printed it on my screen.

In order to establish a connection with a MUD, you must have a ''character'' on the server. You must supply a password to login to the MUD as that character, much like when you login to a UNIX workstation. Once the connection has been opened, all the commands that you type are perceived to come from your character. When you close the connection, the state of your character (location, possessions, etc) may or may not be preserved by the server.

The MUD server typically presents a virtual space organized into ''rooms''. A room, in the MUD sense, corresponds to a place where characters or objects may be located. In the example above, I was in a room described as a park-like area. If I typed ''north'', I would have been moved to a new room, presumably some ways down the windy path and into the forest.

The primary means of communication within a MUD is by talking to other people who are located in the same room as you are. An example transcript might look like:

```
Woj [to Remy]: I looked in Tom's
    office, and found that we don't
    have the right kind of scsi
    cable.
> say Hmm.
You say, "Hmm."
Ivan says, "We've got to have one
    somewhere."
```

```
Woj says, "Well, we do have one,
    but it's only a half-foot
    long."
> emote tries to think of the
    type of cable you need.
Remy tries to think of the type of
    cable you need.
Woj says, "Or, I could turn off
    alewife and steal that one..."
> say "Check my office, if there's
    not one there, bounce alewife."
You say, "Check my office, if
    there's not one there, bounce
    alewife."
Rob says, "Anyone up for some ice
    cream?"
Brian <-
Woj says, "Wait, I'm on my way!"
> emote sighs and chuckles.
Remy sighs and chuckles.
```

A character may say something using the `say` *`something`* command. They may indicate some sort of action by using the `emote` *`something`* command. I typed `emote sighs and chuckles.` in the example above. All of these common commands have short cuts to make them easy to type (`say` can be shortened to a quotation mark, while `emote` can be shorted to a colon) and come naturally after a few minutes of trying them.

It is possible to talk privately with a person using some form of a whisper command, or to talk to someone who is not in the same room by using a paging command. Many other commands exist, and may be created as needed.

One interesting aspect of MUDs is that they impose a spatial metaphor on the participants. One may talk and interact easily with people in the same virtual room, and may use other means to communicate with people in other locations. In this way, the MUD becomes a virtual space reachable via the network.

Applicability of MUDs to Systems Communication

We felt that MUDs had several features that would make them a useful communications tool for the systems group:

- MUDs are interactive in real-time. When one says something on the MUD, all the intended recipients see it immediately. They can answer in the amount of time it takes to type their response.
- MUDs are a networked service. Clients and servers simply need to be on the same network in order to connect to each other. Thus one need not be logged into the same computer as the people with whom one is communicating.
- MUDs are multiuser capable. A large number of people can interact with each other at once.
- MUDs are extensible. Any decent MUD server will have an embedded programming language that may be used to extend the database of server objects and to create new commands. If the tool is to be adapted to new uses, it must be flexible.
- MUDs are exclusive. Only people who have been given characters on the MUD are allowed to connect to it. This is a crucial feature. Any of the 800 users on our network can send me email, but only the systems group can talk to me on the MUD.
- MUDs, in conjunction with most clients, have a history mechanism. Even though the interaction on a MUD happens in real-time, it can be recorded by a client that is connected. Thus one may read a conversation that happened when one wasn't actively involved. Clients also have the ability to save transcripts to files, allowing for a permanent archive of important communication.

We explored other communications tools as possible alternatives. A summary of our findings and opinions is presented in the appendix.

Other Interesting Aspects of MUDs

Once we started using a MUD as a tool, we discovered several other useful features that we had not considered previously.

As mentioned above, MUDs provide a metaphor of real life (or virtual life). In a MUD, people and things exist in a place. One interacts with an object as one would in real life. This creates an interesting context for solving problems and indicating real-life situations. For example, in the systems MUD, I have an attic which is located above the cabin where we normally work. When I'm very busy in reality, I move to that attic, and leave everyone else in the cabin. In this way, I am out of the flow of conversation, but still available on the MUD if someone wishes to discuss a specific issue with me. The spatial metaphor is also used when we build systems tools into the MUD.

Because a MUD is extensible via a programming language, the objects in the MUD can be programmed to do anything that you could do with any other type of program. This is an extremely powerful tool, and we are just beginning to make use of it. For example, we have considered constructing a version of a new building that the College may be expanding into, and then simulating the physical network connectivity within the MUD. We will do this before the building is built in reality, in order to foresee problems that may crop up.

One tool that currently exists in our MUD is a Gopher client. We ported this code from another

MUD, where it was originally developed [Masi93]. Instead of simply running a command that pops a Gopher interface up on the MUD, the Gopher client was built into the spatial metaphor of the MUD, and exists in the MUD as a "Gopher slate", something like a portable computer. People in the MUD may make their own Gopher slates and manipulate them by navigating Gopher menus, much like one does with a conventional Gopher client. Gopher slates differ from other clients in that they may be shared with other participants in the MUD – one can bring up a document on a Gopher slate and then hand it to someone so that they too can read it, or they can put it on a table and jointly manipulate it. The MUD has then become a way in which we can collaborate in our use of Internet resources.

Other items in the MUD that have been programmed are a dictionary, a weather map, and various systems status monitors.

Running a MUD

We are running a type of MUD known as a MOO, which stands for MUD, Object-Oriented. A MOO keeps the database of objects in memory, which means that the process tends to be large. Other types of MUDs keep unreferenced objects on disk, and therefore take up less in-core memory. We chose to use a MOO because of possible future research directions; had we been concerned about computing resources, it would have been wiser to choose a different type of MUD server.

MUDs can be run on practically any UNIX machine; we're running our MOO on a SPARC 670/MP. It currently takes 9 megabytes of disk space to store the code and several backup copies of the object database. The process takes anywhere from 5 to 15 megabytes of memory, depending on what is happening within the MOO. We have found that the server does not create an appreciable load on the computer, and could be run unobtrusively on a slower machine without any trouble; we used the SPARC simply because it was available when we started the MUD.

The MUD client runs on whatever machine the user is using. It is possible to use **telnet** to connect to the MUD. Telnet is not the most useful client, however, because it doesn't separate input from output, and the output from the MUD tends to get confused with the command that one is trying to type. Specialized clients that are easier to use have been written for for the X Window System, emacs, and for several types of personal computers.

For the most part, the members of the systems group use a client for emacs named **mud.el**, which connects to a MUD within an emacs buffer. This allows one to edit commands before sending them to the MUD, to capture output into another buffer, and to keep a history of everything that happened in the MUD that may be viewed as "scrollback" at a later date. Using emacs also makes it convenient to edit code for objects within the MUD itself.

Typically, a member of the group will start up a small-sized emacs on their workstation, connect to the MUD, and then leave it going for the whole time that they are logged in, occasionally glancing at it to see what's happening. Some people use the **screen** program to start an session that may be moved from one controlling terminal to another; this allows one to login to the workstation from another location (from home, for example), without disconnecting from the MUD. In this way, one may read what happened while one was absent, and keep up with any happenings. Other people prefer to disconnect completely when they're not going to be around.

Experiences and Observations

The original environment that we built was a replica of the building that we work in. Everyone built their office, we implemented a network that one could travel through, and we constructed a central meeting place. But we found that having one real-life version of our urban university was enough. Instead, we created a virtual space that is different and somewhat more pleasant, and this has seemed to change the mood of the MUD for the better.

We have named the MUD environment that we built ''The InfoPark''. The InfoPark is in an outdoor setting, with trees, streams, and animals. In the middle of a forest is a cabin, where most members of the systems group go when they're connected. We've also built a university campus for use by one of the research groups within the College, where they plan to hold on-line meetings.

We have found that the MUD is an effective way to hold pre-arranged meetings for people who can't be in the same physical location. We have had virtual systems meetings three or four times, each about various topics. We save a transcript of the meeting and email it to people who weren't present, and refer to it when trying to remember exactly what issues had been raised. Using a MUD in this way is not as time-effective as meeting in reality, but is at least as useful as having a conference telephone call.

We use the MUD as a coordination mechanism. People tend to announce on the MUD what they are doing in real life. Phrases like ''Jim heads into the machine room to check the tapes'', ''Ivan is about to reboot amber'', or ''I'm hungry, who's interested in lunch?'' are commonly seen. We have found this to be so useful that we have encouraged it by writing utilities that people can use to indicate why they are ''idling'' in the MUD. A character is called ''idle'' when they do not respond to activities within the MUD. This normally happens because the user has quit paying attention to the MUD for some reason. When a character is idling ''for office visitors'' or ''to drive home'', the other participants in the MUD can look at that character and see why it idled. In this way, we use the text-based virtual reality to reflect what is happening in real life. Before the MUD, we saw each other only at meetings, or after running all over the building trying to locate each other.

The MUD is used as a quick brain-storming or problem-solving mechanism, often in response to a recent email message. It's common for us to have a five-minute conversation on the MUD about a small systems issue, such as some detail of DNS, how to implement an extension to the file system, or how to fix some user's problem. Previously, these conversations would have happened through slower email, through office visits, or at regular systems meetings. All of these mechanisms are more cumbersome, and would have happened much less frequently. Thus the MUD has enabled new communications patterns.

The largest unexpected benefit of the MUD is that it created a social environment that didn't exist. People have real conversations on the MUD with other participants. These are typically, but not always, about systems-related issues. Because of this, members of the group find out about projects that they're not involved in. The new undergraduate volunteers come to know senior members of the staff that they would otherwise not recognize. The MUD has become a social place for the systems group that is populated even when everyone is logged in from home. It may be said that this takes away from work hours, or cuts down on effectivity while in the office. But, over time, we have seen our systems group, people who had simply worked on related jobs, grow into a real team. This change is due partly to our shared social context.

It is common for someone on the MUD to have a real-life interruption, be it a phone call, an office visitor, or simply being too busy to pay any attention to the MUD. Thus people tend to become inactive suddenly on the MUD. This doesn't create a communications problem – whenever they resume, the buffer is there, containing the old conversation, and it can be continued without difficulty. The problem it presents is when one needs to find a specific person who is on the MUD but isn't paying any attention to it. Our solution is to use a paging verb that causes the idle person's terminal to make an audible beep. If that person can, they will respond to the MUD. If they can't, they probably aren't anywhere near a computer. This is when communicator badges would come in handy (or be horribly annoying, depending on your point of view).

Many of the undergraduates who work with the group have cooperative education jobs in other parts of Boston. During the day, or even for a whole summer, they are unable to come into Northeastern. Because they stay on the mailing lists and can continue to interact with us on the MUD, they are constantly in touch, and can continue to work on projects. It is often difficult to tell whether a person is actually in the building (without asking them, that is), and has become increasingly less important over time as our network communication tools improve.

As part of the experiment, we have invited systems administrators from another site to join us on our MUD. This has been interesting – they often have unique viewpoints on how to solve our problems. We have traded information about tools, discussed collaborative projects, and are more in touch with what is happening at their site. This has been successful largely because we had already had professional relations with them. We originally thought that they should have a separate room in the MUD to discuss their own problems, but it hasn't really been used much. We would have to say that the collaboration between the two groups has been a success, but not a huge success. Perhaps the most interesting thing to come out of it is that they are building their own MUD.

We have created a bulletin board in the MUD that has the list of current systems projects. This has been a failure because it's not really linked in with our other mechanisms for project organization. Once these are mechanisms are more clearly defined, we will undoubtedly build a MUD interface in order to see whether or not it will be useful.

Problems

While the MUD solves many problems, it also has a few of its own. First, as mentioned above, it can be easily interrupted by something happening in real life. This is something that one simply gets used to.

When many people are in a room, the conversations can get confused and intertwined. One quickly learns to pay attention to the conversation one is involved in, and to partially ignore the others. The use of recipient indications (such as ''Rémy [to Ivan]: Where are you?'') solves a lot of the problem. This is, again, something that one adapts to very quickly. On the other hand, it doesn't scale very well. We considered opening our MUD up to anyone who read this paper, but we don't yet understand how to coordinate a hundred people talking to each other.

The MUD requires a bit of time to learn. There are perhaps 10 commands that everyone must learn immediately (say, emote, page, whisper, look, and so on). Learning to administer a MUD is more difficult. This involves making sure the database is backed up, creating or disabling guest characters, and learning the MUD's programming language in order to extend the environment. This is something that can be done incrementally, but does take time. On the other hand, it can also be fun.

Perhaps the biggest problem we've discovered is that the MUD has the potential to be a big distraction. When it's constantly running somewhere on your screen, it can interrupt your concentration. Everyone learns to deal with this differently. I have simply gotten used to the flow of text across one of my windows, and tend to glance at it every few minutes to see what's happening. If I'm busy, I may not look at it for hours. If something important comes up, someone will page me. Several other people on the group iconify the window or bury the buffer below another one, or, when very busy, disconnect. This is part of a bigger problem of time management, and is related to similar interruptions caused by telephones, electronic mail, and office visitors. We have not found the MUD to be any worse than the other distractions.

Future Work

We plan on several possible avenues for future growth. One project that is currently underway are objects that represent entities on the network. We'd like a printer in the MUD that always shows the various printer queues on the network, and has the ability to restart printer daemons (with appropriate security features built in). We're working on a themely representation for computers that will notify us when they're unreachable. Perhaps:

```
A worried-looking squirrel scampers
in through the window and says,
"denali is unreachable at 3:45 pm"
```

... then again, perhaps not.

Current research at Xerox PARC involves integrating MUDs with audio, video, and shared programs [Curt93]. The MUD then becomes the perfect sequencer for these devices because of its spatial metaphor: one could actually talk with and see the people who are in the same room on the MUD, as opposed to hearing everything that is said on a simple shared audio channel (which is what primarily happens with today's multicast backbone interfaces). One could look at a white-board on the wall of the cabin and it have it pop up as an X Windows application on the screen. The application would be shared by anyone looking at the board, and changes could be made and seen by everyone. We expect these tools to be enormously useful, but are concerned that the audio and video may not be as useful as one might think, given the interruptive capability they have, and the lack of a history mechanism.

As other people on other MOOs write neat applications, we will be porting those onto our MUD. Shared World Wide Web readers are likely, as are other interfaces to the Internet's resources.

We are working on a way to access systems administration information from within the MUD, in order to make it even more of a collaborative environment where administrators can meet and discuss problems.

Because of the value of the MUD as a place for virtual meetings, we have created a conference room that we will be sharing with faculty and various groups within the University. We had thought of allowing students on the MUD in order to learn some aspects of object oriented programming, but we value our peace and quiet and will be letting them run their own MUD, where they won't be able to find us.

Conclusion

Our MUD has been a successful experiment, and we plan on continuing to use it. It has solved the communications problems we had, and has provided some unexpected benefits. Among those are the creation of communication channels that had not previous existed, and the ability to work more effectively from remote locations. We recommend it to other systems administration groups and are interested in hearing about their experiences.

Availability

The MOO server is available on parcftp.xerox.com in `/pub/MOO`. Get the latest version of the LambdaMOO code. It runs on nearly any type of UNIX platform. Clients, other MOO code (such as the Gopher slates), and related papers are also at this site.

LP, another type of MUD, is available on ftp.ccs.neu.edu in `/pub/mud/drivers/lpmud`. Database libraries are available in the `/pub/mud/mudlibs` directory. Clients and papers are also on this ftp server.

Acknowledgements

Thanks to Robert Leslie for bringing up the first systems MUD; Erik Ostrom, Larry Masinter, Jay Carlson, and Michele Evard for writing a lot of the code which was ported to the InfoPark; Stephen White and Pavel Curtis for writing MOO; the Argonne National Laboratory Mathematics and Computer Science Support Staff for participating; and the whole InfoPark crew for (on-line and interactive) comments on this paper.

Author Information

Rémy Evard is the leader of the Experimental Systems Group of the College of Computer Science at Northeastern University. He has been at Northeastern for a busy year, during which he changed nearly everything about the network. He received his M.S. in Computer Science from the University of Oregon in 1992, and graduated from Andrews University with a B.S. in Mathematics and Computer Science in 1990. He has been doing UNIX systems administration for five years, but has only been MUDding for nine months. He may be reached as "remy@ccs.neu.edu", or as "r'm" on any of several MUDs.

Bibliography

[Bart90] R. Bartle, "Interactive Multi-User Computer Games", December 1990. ftp://ftp.ccs.neu.edu//pub/mud/docs/papers/mudreport.ps.gz

[Curt92a] P. Curtis, "LambdaMOO Programmer's Manual", August 1993. ftp://parcftp.xerox.com//pub/MOO/ProgrammersManual.ps

[Curt92b] P. Curtis, "Mudding: Social Phenomena in Text-Based Virtual Realities", in the Proceedings of the 1992 Conference on the Directions and Implications of Advanced Computing, Berkeley, May 1992. ftp://parcftp.xerox.com//pub/MOO/papers/DIAC92.ps

[Curt93] P. Curtis and D. Nichols, "MUDs Grow Up: Social Virtual Reality in the Real World", unpublished report, May 5, 1993. ftp://parcftp.xerox.com//pub/MOO/papers/MUDsGrowUp.ps

[Eich88] M. Eichin, R. French, D. Jedlinsky, J. Kohl, W. Sommerfeld, "The Zephyr Notification Server", in Winter 1988 Usenix Proceedings. ftp://athena-dist.mit.edu//pub/zephyr/doc/usenix.ps

[Masi93] L. Masinter and E. Ostrom, "Collaborative Information Retrieval: Gopher from MOO", in The Proceedings of INET '93, June 1993. ftp://parcftp.xerox.com//pub/MOO/papers/MOOGopher.ps

[Vers91] B. Verser, "Network utilization by MUD players (was Re: Gaming)," in Computers and Academic Freedom Newsletter, v.1 n.43, Dec 15, 1991. gopher://gopher.eff.org/1/academic/news/cafv01n43

Appendix – A Comparison of Communication Tools

Before the systems group decided to use a MUD, we considered several other tools that had various features. We present them, along with our own opinions, in the hopes that they will clarify the differences between various communications methods. A chart presenting our opinions is given in Figure 1. This is not intended to be an exhaustive summary, but more of an overview of different types of communication tools.

Asynchronous on-line communication tools

- Electronic mail is used to send a document to any number of recipients. It is, for the most part, reliable, and is used so constantly that it has become convenient through force of habit, if nothing else. It can either be interruptive or not, depending on the receiver's preference. While email communication can be so fast as to be nearly real-time, it does not normally convey the feeling of a real-life conversation. It is possible to keep a history of email messages. Email is probably the most useful communications tool a systems administrator has.

	Real Time	Online	Archivable	Unobtrusive	Multiuser	Exclusive	Extensible
Electronic Mail		•	•	•	•		
News		•	•	•	•		
write	•	•					
msend	•	•					
talk	•	•			•		
IRC	•	•	•	•	•	?	
Zephyr	•	•	?	?	•		•
MUDs	•	•	•	•	•	•	•
telephones	•						
pagers							
walkie-talkies	•				•		
communicators	•				•	•	

Figure 1: Communications Tools – Criteria and Opinions

- News (or USENET news), is a system that allows a document to be spooled on a machine for some time, where many people can access it. News can be shared across many machines or isolated to one site. It is good way to communicate with many people about various topics in a non time-critical manner. News is perhaps the best way to send announcements that many people should see. Multi-party discussions can work relatively well in news, but can take time, and have a tendency to diverge into irrelevant topics. News is normally accessed by the readers if and when they choose to read it, not when the news is first available.

Real-time on-line communication tools

- The UNIX **write** command is used to send a message to a user's terminal. It can be disabled by the receiver with the use of the **mesg** command. **Write** is real-time, but is inconvenient for all but the shortest of messages, because it only reads standard input and tends to create a mess on the receiver's screen.
- The **msend** utility is the next step up from **write**, allowing one to send messages to users of another computer. The input stream is more flexible, but the output still tends to clutter up the receiver's terminal.

Interactive on-line communications

- The UNIX **talk** command is useful for short conversations between two people. Enhanced versions of it allow conversations between any number of people. It is as real-time as it is possible to get over an ASCII connection, allowing one to see the typos made by the other person. We found talk to be annoying primarily because it is extremely interruptive (drat, which window is that beep coming from?), there is a confusion between talk protocols on various operating systems, and the interface is inflexible and primitive. Nonetheless we do use it occasionally.
- Internet Relay Chat, or **irc**, is a global chat program. It is useful for interactive real-time conversations between multiple people. While it would have solved most of our initial needs, we chose not to use it because of code stability problems, and because we weren't sure that it would truly be exclusive to the systems group.
- Zephyr, the communications server from Project Athena, provides a networked, scalable messaging service that is accessible via many types of clients. It allows real-time messages to be sent from one person to any number of people who are on the local network (for various definitions of local). With the right client it's possible to use Zephyr to have an ongoing conversation. More importantly, it can locate a user anywhere on the network and deliver the message (a feature that a MUD can not easily replicate). We think Zephyr may be as useful as a MUD and will be testing it in the future, perhaps in conjunction with MUDs. We did not initially try it because of the difficulty of bringing it up in a non-Athena environment.
- MUDs provide a multiuser extensible environment that can be used as a communications tool. A user connects to it by using telnet or a more sophisticated client. The primary disadvantages of a MUD are that it must be actively administered and that the user must initiate the connection. This is as opposed to Zephyr or email, where the user is located wherever they are logged in. These negatives did not significantly hinder us, so we elected to use a MUD.

Interactive off-line communications

- Telephones, pagers, cellular phones, and walkie-talkies are all solutions we've seen used. They have their uses when one is away from a computer, but are prohibitively interruptive. While they do solve the real-time communications problem, we didn't feel that provided the type of solution that we were looking for, largely because of the inconveniences that come with them. For example, telephones are only effective in real-time if one is near them when they ring, while walkie-talkies must be carried everywhere, and are constantly making noise. We originally thought that Star Trek communications badges would be exactly what we wanted but have since decided that they would simply be more advanced forms of annoyance devices. Perhaps the ideal solution for all of these will be the light-weight, mobile, internetworked, portable communications tool that may be turned off and ignored when necessary. A Powerbook running PPP over a cellular modem could provide interesting possibilities, for a price.

Looking at all the options, we felt that the combination of electronic mail and MUDs solved nearly all of our internal communications needs. Neither completely solves the problem, but they work as nice complements to each other. We use email primarily for internal announcements and instructions, and the MUD primarily for on-line coordination and planning.