Avoiding Overload in Multiuser Online Applications

A Thesis Submitted to the
College of Graduate Studies and Research
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Roger Blum

Permission to Use

Abstract

One way to strengthen the bond between popular applications and their online user communities is to integrate the applications with their communities, so users are able to observe and communicate with other users. The result of this integration is a Multiuser Online Application (MOA). The problem studied in this thesis is that MOA users and systems will be overloaded with information generated by large communities and complex applications. The solution investigated was to filter the amount of information delivered to users while attempting to preserve the benefits of dwelling in a MOA environment. This strategy was evaluated according to the amount of information it was capable of reducing and the effects as seen by MOA users. It was found that filtering could be used to substantially reduce the information exchanged by users while still providing users with the benefits of integrating application and community.

Acknowledgements

First and foremost, I would like express my gratitude to Carl Gutwin for his extraordinary guidance and patience throughout my studies. Second, I wish to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) and TRLabs for their generous financial support. Finally, I wish to thank the faculty, staff, and students at the Department of Computer Science for their help and friendship, as my association with them was one of the most enriching aspects of my time as a student.

It takes a whole village to raise a child.
— possibly old, possibly African, proverb

In the spirit of this quotation, this thesis is dedicated to those who guided and supported me through the years leading up to its completion. First, heartfelt thanks go to my wife, Susan for ultimately providing the motivation to complete my work. Second, I thank my two daughters, Erica and Eva, for being the greatest teachers I have ever known. Third, I thank their Auntie Nadine for the lavish spoiling she provided them during my period of glorified unemployment. Finally, I thank my parents, Mary and Edwin, for their *unconditional* support for everything I have ever done, regardless of their own good judgement.

Contents

Permission to Use i

Abstract ii

Acknowledgements iii

Contents v

List of Tables viii

List of Figures ix

List of Abbreviations xi

1 Introduction 1

1.1 Problem . . . 3
1.2 Motivation . . . 4
1.3 Solution . . . 5
1.4 Steps in the Solution . . . 7
1.5 Evaluation . . . 9
1.6 Contributions . . . 9
1.7 Thesis Outline . . . 9

2 Related Work 11

2.1 Online Communities . . . 11
2.1.1 Representations of Community . . . 13
2.2 Human Limitations in Processing Visual Displays . . . 21
2.3 Distributed Groupware . . . 25
2.3.1 Groupware Classifications . . . 25
2.3.2 Groupware Architectures . . . 26
2.3.3 Real-Time Multiplayer Networked Games . . . 28
2.4 Filtering . . . 29

3 MOAs 36

3.1 ChessWorld, a Visualization of an Online Chess Community . . . 37
3.2 A MOA From a User Perspective . . . 39
3.2.1 MOA Goals . . . 39
3.2.2 User Interface . . . 40
3.3 A MOA From a System Perspective . . . 41
3.3.1 MOA Data Types . . . 42
3.3.2 Effects of Application and Community on MOA Data . . . 43

3.3.3 Filtering 44
3.4 MOA Architecture 45
3.4.1 Community Component 45
3.4.2 Application Component 47
3.5 SketchWorld, a MOA Prototype 47
3.5.1 Community Component 48
3.5.2 Application Component 54

4 Filtering **60**
4.1 Filtering in a MOA 60
4.1.1 How Filtering Could Affect MOA Data 61
4.1.2 Where Filtering Could Take Place 62
4.1.3 When to Filter 64
4.1.4 User- and System-Controlled Filtering 64
4.1.5 Configurability 65
4.2 Filtering in SketchWorld 66
4.2.1 Design Decisions 66
4.2.2 Filters and Filter Manager 67
4.2.3 Filter Implementation Details 69
4.2.4 Filter Manager Implementation Details 70
4.2.5 Filters Implemented 71
4.2.6 Configurability 76

5 Evaluation **78**
5.1 Phase 1: Preliminary Evaluation 78
5.1.1 Methodology 78
5.1.2 Findings and Interpretation 79
5.2 Phase 2: Qualitative User Study 82
5.2.1 Methodology 82
5.2.2 Findings and Interpretation - MOA Functionality Questions . 85
5.2.3 Findings and Interpretation - Task Load Questions 88
5.2.4 Findings and Interpretation - Comments, Suggestions 90
5.2.5 Observations 91
5.2.6 Summary of Results 92
5.3 Phase 3: Trace Log Examination 93
5.3.1 Goals 93
5.3.2 Methodology 94
5.3.3 Findings and Interpretations 95
5.3.4 Summary of Results 98
5.4 Phase 4: Traffic Simulation 99
5.4.1 Goal 99
5.4.2 Methodology 99
5.4.3 Findings and Interpretation 106
5.5 Overall Evaluation Summary 107

6 Discussion **109**
6.1 Findings 109
6.1.1 Filtering Can Reduce Burdens on Users and Systems 109
6.1.2 Users Accept MOAs 110
6.2 Lessons Learned 110
6.2.1 User Perspective 110
6.2.2 Systems Perspective 111
6.3 Future Directions 113
6.3.1 Longer Term Study 113
6.3.2 Other Applications as MOAs 114
6.3.3 MOAs and Large Communities 115

7 Conclusion **116**
7.1 Summary 116
7.2 Contributions 116
7.3 Future Work 118
7.3.1 Future Work Derived From the Evaluation of SketchWorld . . 118
7.3.2 Other Future Work 119

References **125**

A Qualitative Experiment Questionnaires and Responses **126**
A.1 Consent Form 126
A.2 Participant Profiles 129
A.2.1 Profile Form 129
A.2.2 Profile Responses 130
A.3 Task Responses 131
A.3.1 Task Response Form 131
A.3.2 MOA Functionality Questions 133
A.3.3 Task Load Questions 133
A.3.4 Participants' Filter Configurations 133

B Quantitative Analysis - Station Graphs **135**
B.1 Data Received by User 4 135
B.2 Data Received by User 5 139
B.3 Data Received by User 6 143
B.4 Data Received by User 7 147
B.5 Data Received by User 8 151

List of Tables

A.1 Participant Profile Responses 130
A.2 Functionality Question Responses 133
A.3 Task Load Question Responses 133

List of Figures

2.1 Nethack, a dungeon exploration game that uses text characters to represent a 2D space. 15
2.2 Buddies list of a chat program showing the presence and status of colleagues. 16
2.3 The Palace chat room. Participants are represented by avatars, and speech is shown in cartoon bubbles. 17
2.4 Chat Circles. Text is displayed in coloured circles to show the flow and rhythm of the conversation. Circle colour identifies the contributor of the text, and circle size indicates recency, with larger circles representing more recent contributions. 19
2.5 Visual Who. The location of people's names indicates their areas of focus. 31
2.6 Quake, a popular multiplayer game 32
2.7 Croquet space presenting a drawing application 33
2.8 Junk mail, or spam filters identify and isolate emails that contain questionable content, or originate from unreputable sources. 34
2.9 MySpace members promote and grant special privileges to other members. 35

3.1 ChessWorld, a visualization of an online chess community. 38
3.2 SlideWorld, illustrating one possible MOA user interface where users do their work within the community view. 41
3.3 SketchWorld system architecture. 49
3.4 Community View, consisting of the main view, radar view, list view, and chat pane. 51
3.5 The application component of SketchWorld, a drawing application. The user selects the tool to use in the palette window and uses it in the drawing window. The drawing contains samples of all possible element types. 55

4.1 The exchange of quality requests and application state updates in a MOA. 63
4.2 SketchWorld's filtering subcomponent. Three filters are shown and the rest hidden for clarity. 68
4.3 The Recent Visit Filter in action. 73
4.4 SketchWorld's filter control panel. 77

5.1 Traffic received by User 8 from User 5. User 8 refrains from manually adjusting observation quality. 87
5.2 Traffic received by User 5 from User 4. User 5 increases the quality for User 4 briefly, then decreases it back to its original level. 88
5.3 Traffic received by User 6 from User 7. 96

5.4 SketchWorld drawing being observed in the community view during traffic simulation. 101
5.5 Traffic received by 20% quality observer during traffic simulation. . . 102
5.6 Traffic received by 40% quality observer during traffic simulation. . . 103
5.7 Traffic received by 80% quality observer during traffic simulation. . . 104
5.8 Traffic received by 100% quality observer during traffic simulation. . . 105

B.1 Traffic received by User 4 from User 5. 135
B.2 Traffic received by User 4 from User 6. 136
B.3 Traffic received by User 4 from User 7. 137
B.4 Traffic received by User 4 from User 8. 138
B.5 Traffic received by User 5 from User 4. 139
B.6 Traffic received by User 5 from User 6. 140
B.7 Traffic received by User 5 from User 7. 141
B.8 Traffic received by User 5 from User 8. 142
B.9 Traffic received by User 6 from User 4. 143
B.10 Traffic received by User 6 from User 5. 144
B.11 Traffic received by User 6 from User 7. 145
B.12 Traffic received by User 6 from User 8. 146
B.13 Traffic received by User 7 from User 4. 147
B.14 Traffic received by User 7 from User 5. 148
B.15 Traffic received by User 7 from User 6. 149
B.16 Traffic received by User 7 from User 8. 150
B.17 Traffic received by User 8 from User 4. 151
B.18 Traffic received by User 8 from User 5. 152
B.19 Traffic received by User 8 from User 6. 153
B.20 Traffic received by User 8 from User 7. 154

List of Abbreviations

1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional
CSCL	Computer Supported Cooperative Learning
CSCW	Computer Supported Cooperative Work
CVE	Collaborative Virtual Environment
FICS	Free Internet Chess Server
GIF	Graphics Interchange Format
GIMP	Gnu Image Manipulation Program
IRC	Internet Relay Chat
ITS	Intelligent Tutoring Systems
JPEG	Joint Photographic Experts Group
MOA	Multiuser Online Application
MUD	Multiuser Domain
STSS	Short Term Sensory Store
UDP	User Datagram Protocol
VE	Virtual Environment

Chapter 1

Introduction

Online communities are global gathering places for people with common interests. They exist in the form of chat rooms, message boards, and email lists, and while there are differences in their dynamics, they all serve as places for people to carry on discussions, express opinions, or help others. Many online communities are devoted to a particular application. For instance, the Gnu Image Manipulation Program (GIMP)[1] is supported by web sites containing resources contributed by users, several mailing lists, a Usenet news group, and an IRC chat channel. At these venues, participants provide assistance, discuss bugs, future features, or other graphics programs, as well as engage in social, nontopical conversations.

Application and community are not connected, however—to access the community, application users must use a web browser, mail client, or chat client alongside the application. This disconnection hinders the effectiveness of the community in two ways. First, community participation is limited to users who make the conscious effort to take part in it, since joining the community does not occur automatically when they use the application. Many users may not realize that some or any communities exist until they experience problems and actively seek assistance. The people who are isolated from community will not be aware of the activities of others, others will not be aware of theirs, and the community is smaller and weaker than it could be. Second, the application itself is not available for use as a communications tool by its community. The community is limited to rudimentary communications such as text messages or sharing screen shots, poor substitutes for the interaction possible

[1]The Gimp Home Page `http://www.gimp.org`

when users are able to observe each other's actions.

A response to the shortcomings just detailed is to integrate community and application, resulting in a Multiuser Online Application (MOA). A MOA, by definition, has the following features:

- facilities for a user to perform individual work, such as create a drawing, write a letter, or study molecular structures.
- facilities for a user to observe other users in the community performing their own work.
- facilities for a user to communicate with other users in the community, for instance by text, audio, or video chat.

When a user starts up a MOA, he/she will be presented with a space in which to perform work and a space in which he/she can observe representations of other MOA users. These two spaces could be combined into a single space where both work and observation are performed. Not only will this user be able to observe others, but others will also be able to observe him/her. When two or more users choose to converse, they may do so using the MOA. The conversation may be limited to just the chat window or whatever facility is provided, or it could be augmented by using the observation space to point and gesture at aspects of each others' work. A single-user application is the virtual equivalent of sitting in a solitary cubicle; a MOA is the virtual equivalent of sitting in an open-plan office filled with colleagues.

A MOA designer will face the challenge of determining how the volume of information generated by users and their activities will be transported and presented to other users, as this volume could potentially be taxing on both users and systems. This information may be broken into three categories. First, there is information about people in the MOA, such as their location, recent history, notable abilities, and preferences. Second, there is information about the interaction between these people and the application, such as commands and gestures, and the resulting products of these interactions. Third, there are the communications between people, such as chat, audio, or video.

When considering architectural requirements for a MOA, community size must be taken into account, as it is conceivable that several thousand people would be brought together by a popular MOA. In a commercial environment, membership might be restricted to protect privacy and trade secrets, or some people may choose to opt out for personal reasons, but there could still be many people in attendance. Each member will contribute and receive a significant amount of data, so as community size increases, so will demands on systems and users.

Application complexity and application data size must also be considered. Modern single-user applications are complex and taxing to both their users and the systems they run on; adding the ability to watch and communicate with other users will only make them more so.

1.1 Problem

The problem examined in this thesis is: due to community size and application complexity, both users and systems may be overwhelmed by the information generated within a MOA.

From a user's standpoint, the power to observe and interact with fellow application users can be a curse as well as a blessing. When confronted with a landscape dotted with several hundred users, will it be possible to search for a select few who can help solve a problem or offer advice? Will this landscape be so busy that peripheral activities make it impossible to focus on a single activity? Will the burden of digesting the sheer volume of information be so great that users ignore or discard MOAs altogether?

From a systems standpoint, information overload will affect the senders and receivers of that information, as well as the network infrastructure connecting them. Each running instance of the application will be a source and destination for data. The sender, in addition to bearing the burden of the application's processing demands, will be burdened with the task of transmitting updates to other users, informing them of actions taken by the user or other events of significance related to the user's work. In addition to task-related traffic, users will also send communications such as chat messages to other users. There will also be management data to mark the arrival and departure of users, and any other events regarding the MOA's state. At some point, the task of generating and sending MOA data will cause the sending system to bog down and interfere with its primary job of executing its user's commands. The network that connects all users of a MOA will receive a stream of data from every user, and through a central server or other mechanism route it to other users. When the network becomes saturated with MOA traffic, throughput will decrease due to packet collisions, overflowing queues, and other maladies associated with excess traffic. The receiver of MOA data will be tasked with processing, storing, and displaying the information sent to it in some meaningful way. Since the receiver is also a sender, these burdens will add to those previously described.

1.2 Motivation

Confronting the problem of overload is necessary to make MOAs feasible. If suitable methods are not devised to reduce the information directed at MOA participants, MOAs will be sluggish and unusable and overwhelm both humans and systems. It is not enough to rely on faster hardware and increased network capacity to deal with this problem, as community size and application complexity will likely also increase with time, so it is important to make the best use of available resources. Furthermore, human capacity will likely remain constant and will require assistance to accommodate the demands of taking part in an increasingly large community.

If MOAs can be made feasible, they would offer a number of benefits over an application that is disconnected from its community:

- *A larger community.* Application users would be more likely to participate in the community, since little or no effort would be required to join it. A larger community implies more people able and willing to help others, and an effectively greater cumulative expertise.

- *A stronger work group.* Physically distributed members of a working group, for instance, would feel less isolated, be more prone to chance interactions with other community members, and more likely to engage in social conversations, all essential to improving group effectiveness. They would be more aware of each others' presence.

- *Enhanced communication.* In a MOA, the application itself can be used for communication. Distributed group members would be able to watch other members. In a teaching setting, the instructor would use the application while students watched. Chat would be used for narration, as well as interaction among students and instructor. Conversely, the instructor would be able to monitor the progress of students and offer feedback.

- *Partner finding.* In an effort to find a potential collaborator or to solve a problem, a user would be able to "browse" other users to find someone with the necessary abilities to help, or perhaps the system would recommend potential partners to the user.

- *A mechanism for seamless shifts between single-user work and shared work, in a MOA that allows users to work collaboratively.* A typical scenario might be the following: a user browses through the community, finds someone who is doing interesting work, offers to join in, and after receiving permission to do so begins to work collaboratively with the other user.

1.3 Solution

The solution investigated in this thesis is to restrict the amount of information transmitted and presented to the user by introducing a filtering system to block information about "uninteresting" participants, the premise being that a MOA participant will have varying degrees of interest in others, with only a select few being interesting. As a participant becomes more and more interesting to his/her observers, the system allows more and more information to be sent to the observers.

The filtering system consists of several individual filters and a manager to coordinate their activities. Each individual filter acts in response to user activities: some will monitor the activities of the observing user, deducing which users are interesting and which are uninteresting; other filters monitor the activities of the observed user, deducing from those activities if the user is interesting to others. The filters are also user-configurable: a user can create a configuration where several filters act simultaneously, with the user specifying which filters figure highly in the calculation of interest levels and which filters play a less significant role. Filtering reduces strain on the user by withholding information that is not important to him/her, while at the same time reducing the strain on systems, as data that does not reach the user can be throttled at some point during its journey from sender to receiver. The filtering system, to be successful, must achieve a balance where it still lets enough information through to allow the MOA to remain a useful environment where users are able to observe in sufficient detail all interesting activities and maintain awareness of other users.

The design of a filter specifies the answers to two questions: *how* to reduce data traffic, and *when* to reduce data traffic. There are two ways to reduce data traffic:

- *Reduce transmission frequency.* Send periodic messages at a reduced rate, or aggregate these messages and send only their cumulative effect. In the extreme case, block messages completely. Transmit messages regarding interesting people frequently and immediately, and transmit messages regarding uninteresting people less frequently, after some delay, or not at all.

- *Reduce message size.* Send smaller and less detailed versions of large, fully detailed messages. Messages about interesting people convey most or all information about their activities with the greatest detail possible, while messages about uninteresting people are pared down to convey only the most essential details.

The question of *when* to reduce data traffic is more problematic since the number of possible answers is nearly limitless. The choice of a MOA's visual representation is a strong influence as it immediately leads to a number of well-known solutions which are associated with the representation and provides boundaries to potential novel solutions. For instance, if the MOA uses a 3D spatial representation, proximity would be a natural solution, as is done in multiplayer online games—objects close to the observer would be presented with high fidelity, and objects far away with low fidelity. Object detail is also used to create the illusion of a 3D space on a 2D computer monitor, however, so any filters that manipulate object detail could potentially destroy the illusion as nearby objects could be presented with the same low fidelity allocated for distant objects.

In this project, filters were devised to explore the use of the following factors as a means of deciding when to reduce data traffic:

- *Recency.* Recently observed inhabitants were shown with greater fidelity.
- *Familiarity.* Frequently observed inhabitants were shown with greater fidelity.
- *Popularity.* Three filters were built to study the effectiveness of popularity-based schemes: social networking, where interesting users would report the users that they found interesting; community rating, where users who achieved increased levels of popularity were reported to all other users; mutual interest, where users who expressed interest in each other were additionally boosted.
- *Activity level.* Busy users automatically became more interesting to others.
- *Common activity.* Inhabitants who were engaged in a similar activity to the observer were shown with greater fidelity.
- *At random.* Users who were otherwise uninteresting were picked at random and made interesting for a brief period.

1.4 Steps in the Solution

The steps in the solution investigated in this thesis were as follows:

- *Choose the type of application to study.* Candidate types, for instance, were text-based applications such as word processors and spreadsheets, and graphics-based applications such as drawing and painting programs. A simple drawing application was chosen as it was felt that as a graphical application it would be an effective and compelling showcase as a MOA prototype, plus the data structure used to represent a drawing could have diverse characteristics and thus lead to a thorough examination of the effects of filtering in a MOA.

- *Choose a way to represent a community of MOA users.* The representation was required to show both the presence and activities of each user. A 2D, or overhead view representation was chosen as it would be compatible with the 2D nature of the application, and this presentation would be readily navigable, a property thought to be relevant in a setting where one user would wish to "browse" in search of interesting subjects to observe.

- *Identify the types of data that could be sent to a MOA user.* Each type has different bandwidth, reliability, frequency, and delay characteristics. For instance, one type was the family of messages sent when users joined or left the MOA. These messages were small and infrequent relative to other types, and the benefits of sending them immediately and reliably outweighed any savings that could be derived by filtering them in some manner.

- *Devise filters to control data flow.* These filters would decide when to limit each type of data, and by how much. For instance, a filter based on popularity would limit the data sent about unpopular users, while allowing data about popular users to circulate freely.

- *Build a MOA prototype featuring a filtering system that implements the filters devised in the previous step.* To facilitate the study of these filters, it was possible to enable each one individually, or enable several at once to work in unison. The resulting prototype was studied to evaluate both the concepts of MOAs and of the use of filtering to reduce the possibility of overload.

1.5 Evaluation

The research was oriented around a detailed case study of a particular application type; therefore, the evaluation was oriented towards the experiences that arose from building and testing the case study prototype. Individual filters and several combinations of filters were evaluated quantitatively to see how they affected the flow of data and qualitatively to see how they affected the view of community, whether users accepted and understood them or if they found them distracting or confusing, and if they still allowed the MOA to remain functional and effective.

1.6 Contributions

The two primary contributions of this research are the demonstration of the concept of a MOA and evidence that suggests filtering can be employed to reduce the possibility of user and system overload while still allowing the MOA to be beneficial to users. The two secondary contributions are SketchWorld, the reference implementation of a MOA developed for study, and design guidelines for future MOA development based on the study of SketchWorld.

1.7 Thesis Outline

The following outlines the structure of the remainder of this thesis:

Chapter 2 presents a survey of related work that provides the foundation for MOA research. First is a discussion of the nature and purpose of online communities, and the ways in which they may be represented, and the importance of awareness. Second is a discussion of human limitations in dealing with visual displays and how avoiding these limitations is necessary to avoid conditions of user overload. Third is a discussion of distributed groupware, since a MOA is an instance of this class of computer applications. Fourth is the topic of filtering and how it and related techniques are currently employed in distributed groupware, as well as online message boards and social networking web sites.

Chapter 3 presents the ideas inherent in a MOA. First is a description of what a MOA would provide to its users, and how it might appear to them. Second is a description of a MOA from a system's perspective—the data that MOA users will exchange, how the application portion of a MOA and the user community itself will affect the MOA, and how filters could be used to alleviate the stresses caused by a demanding application and a large, active user community. Third is a general architecture that could support the functionality of a MOA, and finally the chapter concludes by presenting SketchWorld, the MOA prototype that was developed and studied.

Chapter 4 discusses filtering, first by presenting a general framework capable of meeting the needs of a MOA, then by presenting the filtering system implemented in SketchWorld and the behaviour of the filtering techniques investigated.

Chapter 5 presents the evaluation of SketchWorld, which consisted of four parts: a series of preliminary evaluations to search for deficiencies and garner initial feedback from users; a user study to assess acceptance of MOAs and of filtering within MOAs; an examination of trace logs taken during the user study to discover patterns in filtering and weaknesses in the system that should be confronted; a traffic simulation to weigh the effectiveness of filtering with reducing data exchanged by users.

Chapter 6 discusses the results of Chapter 5, explaining how they support the hypothesis that filtering can reduce the possibility of user and system overload while still maintaining the desired characteristics of a MOA. This chapter also presents the lessons learned for future MOA development and possible research avenues that merit exploration based on experiences during the evaluation.

Chapter 7 summarizes the research done in this thesis, presenting its contributions and highlighting topics for future research on MOAs.

Chapter 2
Related Work

Six bodies of research form the foundation of MOA development. First is the study of online or virtual communities. Second is the study of community representation, the manner in which an online community or any large body of data may be presented to a user. Third is the study of awareness and why it is important within the scope of virtual environments. Fourth is the study of human limitations in interpreting the kinds of information that will be presented in a MOA. Fifth is distributed groupware, since a MOA would be an application that brings many physically separated users together to observe and communicate with each other in regard to their work. Finally, the sixth body is the realm of real-time multiplayer networked games, more specifically the techniques that game designers employ to make the best use of available network and other system resources with the goal of bringing globally distributed game players together in a virtual environment.

2.1 Online Communities

According to The Collins English Dictionary [27], a community is

> **1. a.** the people living in one locality. **2.** a group of people having cultural, religious, ethnic, or other characteristics in common. **3.** a group of nations having certain interests in common.

Another definition of community is an "identifiable self-conscious group with shared common interest" [24]. For the research presented in this thesis, the important points to draw from these definitions are that communities are made of people with a common interest, that these people realize they are part of the community, and that physical proximity, while allowed, is not necessary. There are no set bounds for the size of a community: some communities may consist of only a few people, while others may consist of millions. For this thesis, however, the community size of interest is a working group of anywhere from a few to dozens of people.

Online or virtual communities are communities where members interact with each other by electronic means [61]. Rather than travelling to meet at a designated location, online community members travel to the nearest computer terminal with internet access and meet at a designated chat room, web site, newsgroup or email list.

Awareness is an important device in an online community: mechanisms that work to provide it indicate when others are present, who they are, and what they are doing. According to Dourish and Bellotti [16, page 107], awareness is "an understanding of the activities of others", providing "a context for your own activity". Awareness gives online community members a sense of who they are with and what they are doing; it helps define what the community offers to its members and in return what the community expects of its members.

Facilitating awareness offers the benefit of making unintended interactions possible. In an office setting, one of the advantages of placing employees in proximity to each other is that it leads to incidental contact and unintended, informal meetings in hallways, in turn leading to productive conversations and collaborations that might not have happened otherwise [33]. In a virtual environment, similar chance interactions are possible when participants are notified of the arrival and presence of others. In groupware design, awareness and social presence, the knowledge of existence and availability for interaction of others, are seen as essential requirements necessary to bring people together in a virtual space [12, 18].

The development of relationships in a group also leads to better partner finding, whether people are brought together by their direct association with one another, or indirectly by one person finding a partner through a series of intermediaries. In virtual environments, systems have been developed to enhance partner finding beyond analogs in the real world by analysing virtual personas.

Communities serve other purposes besides the common interest or attribute that defines them. For example, they satisfy social needs such as giving a feeling of belonging [29, 48]. Online communities also serve these needs, being much more than mere sources of information [42, 60]. Online community members provide social and emotional support and entertainment to one another, band together to achieve goals or solve problems, and have social structures and norms as well as means of enforcing them. Such structures are necessary to curb the dark side of human nature extending into cyberspace [14], since community members may be profoundly affected by their online interactions.

User communities are those that are devoted to a computer application or to a class of applications. In user communities, people seek and sometimes receive help using the application, provide feedback to application developers, and generally share ideas.

2.1.1 Representations of Community

The way an online community is represented to its members may influence how they will access and interact with it as the representation will govern what activities and interactions are possible. A representation of community will consist of the embodiments of community members, the objects or artifacts that are available to community members, and the interactions taking place involving members and objects. Community members will want to know who else is present and what they are doing, and have the ability to communicate or take part in an activity with them.

Text Based Representations

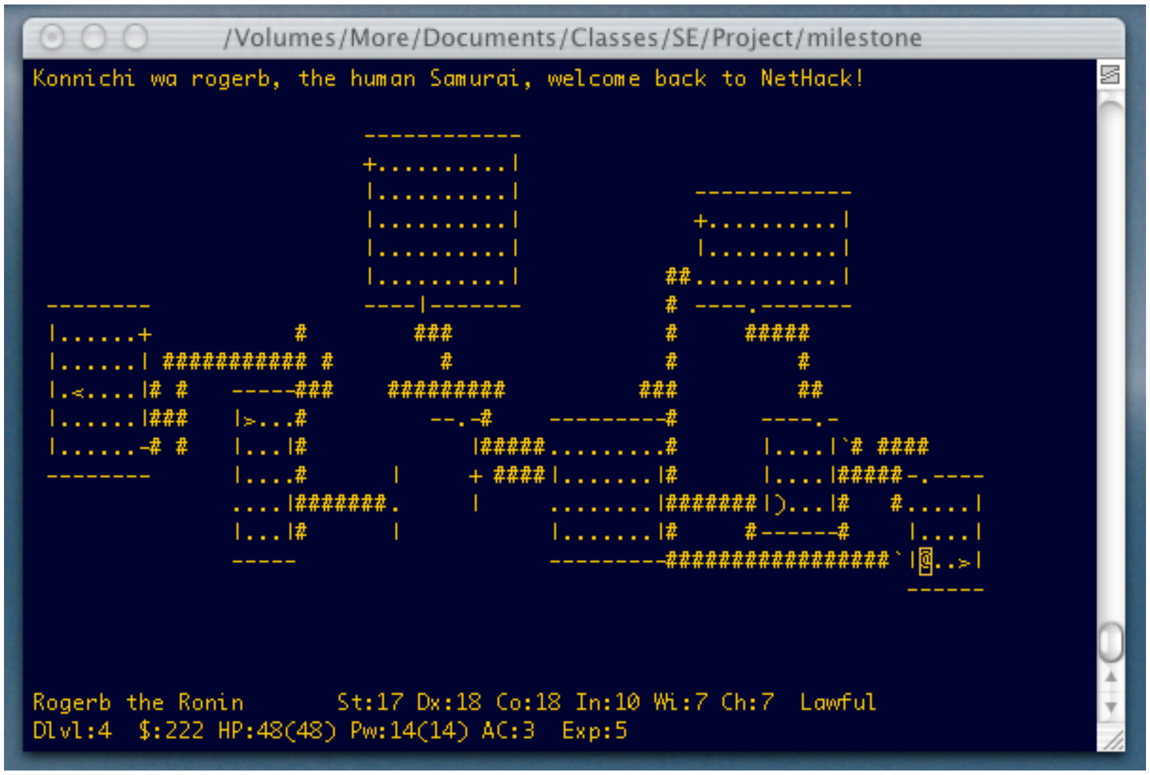
The oldest and simplest representations of online communities are text based. The first online communities were email mailing lists [20] and Usenet newsgroups [43], where members exchanged text messages. Later on, message content diversified to include graphic images, or anything else that could be transmitted digitally. More recently, a number of other delivery mechanisms such as web sites and chat [26] have found their way into everyday use. In spite of their rudimentary nature, text based communities are popular and enduring.

Multiuser Domains (MUDs), originally developed for online gaming, have also been adopted for use in the workplace as virtual office spaces [11]. They support lightweight, informal communication that may be either synchronous or asynchronous. Participants are able to leave behind artifacts such as notes so others may see them later. Since they were originally meant for gaming, they possess a number of features that are less desirable in a work environment, and lack a number of desirable features. Of particular interest was the desire for the ability to present richer content, such as a whiteboard or a spreadsheet clipping [11].

While text-based communities are popular and functional, they do have a number of limitations. Visual concepts are difficult to express in words, and providing an image or diagram instead of formulating a verbal description is an easier task for a message sender and the result is more readily comprehensible by the receiver. In spite of enhancements to allow the use of graphical images and even movie clips, these enhancements tend to be used as merely an occasional supplement to text communications. When dealing with a subject that does involve graphics, two choices that a user has are to somehow translate a graphical concept into language, or create and send a screen shot or movie and hope that his/her audience is able to receive and view it properly. While text can be used to create spatial representations (Figure 2.1), traditionally pure text-based environments such as chat do not readily offer the use of spatial concepts such as location or movement. Text allows for messages such as "John is in the room" and "Steve is running to the telephone", but these require the reader to remember that John is in the room and to imagine Steve running for the telephone. Finally, text-based environments confront a language barrier. Not only must all participants know the language or languages being spoken, they must also share common terminologies.

1D Representations

In this thesis, it is assumed that one-dimensional (1D) representations are graphical entities arranged in a linear fashion as opposed to text representations such as text on



/Volumes/More/Documents/Classes/SE/Project/milestone

```
Konnichi wa rogerb, the human Samurai, welcome back to NetHack!

                         ------------
                         +..........|
                         |..........|             ------------
                         |..........|             +..........|
                         |..........|             |..........|
                         |..........|           ##...........|
 --------                ----|-------           # ----.-------
 |......+           #        ###                #     #####
 |......| ########### #        #                #         #
 |.<....|# #   -----###    #########          ###         ##
 |......|###   |>...#         --.-#    ---------#      ----.-
 |......-# #   |...|#            |#####.........#      |....|`# ####
 --------      |....#      |     + ####|.......|#      |....|#####-.----
               ....|#######.     |     ........|#######|)...|#   #.....|
               |...|#      |           |.......|#     #------#    |....|
               -----                   ---------#################`|@..>|
                                                                  ------



Rogerb the Ronin       St:17 Dx:18 Co:18 In:10 Wi:7 Ch:7  Lawful
Dlvl:4  $:222 HP:48(48) Pw:14(14) AC:3  Exp:5
```

Figure 2.1: Nethack, a dungeon exploration game that uses text characters to represent a 2D space.

a computer console that may also be arranged in a linear fashion. 1D representations may consist of text but also may feature graphical adornments such as icons. One example of a 1D representation is the buddies list used in chat programs (Figure 2.2). A buddies list shows when friends and colleagues are logged in, and may also show status or availability (Idle, Busy, Away). Position in the list may be used to convey meaning: names may be arranged in alphabetical order, by status, importance, or by group. Iconic symbols may also be present to convey identity, status, or other meanings. The advantages of symbols over text are that they transcend language boundaries, and may be more compact and more easily recognizable than text. 1D representations are capable of being richer and more elaborate than pure text, often at the cost of increased screen real estate and graphical processing.

With the use of position as an indicator of meaning come the tasks of location and navigation, although these tasks are usually associated with locations involving two or more dimensions.

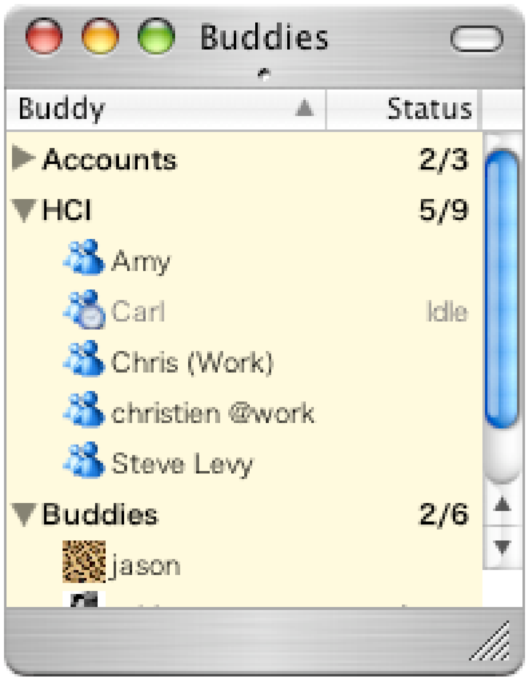


Figure 2.2: Buddies list of a chat program showing the presence and status of colleagues.

2D Representations

Two-dimensional (2D) representations of data expand on the use of position by introducing the use of a second dimension. A 2D space often is presented as an overhead or map view, which readily provides location and route knowledge allowing the navigation between locations [35]. When online communities are presented as 2D spaces, community participants are presented as avatar-like symbols, with their placement in the space being random, according to participant status, or according to participant preference. The addition of a second dimension typically implies the addition of the following abilities, although these abilities may be available with less effectiveness in 1D representations:

- *The ability to use artifacts.* Objects in the 2D space may be either the participants themselves or tools and props that the participants may use or manipulate.

- *The ability to use movement.* An object moving from one location to another may indicate change in activity or interest.

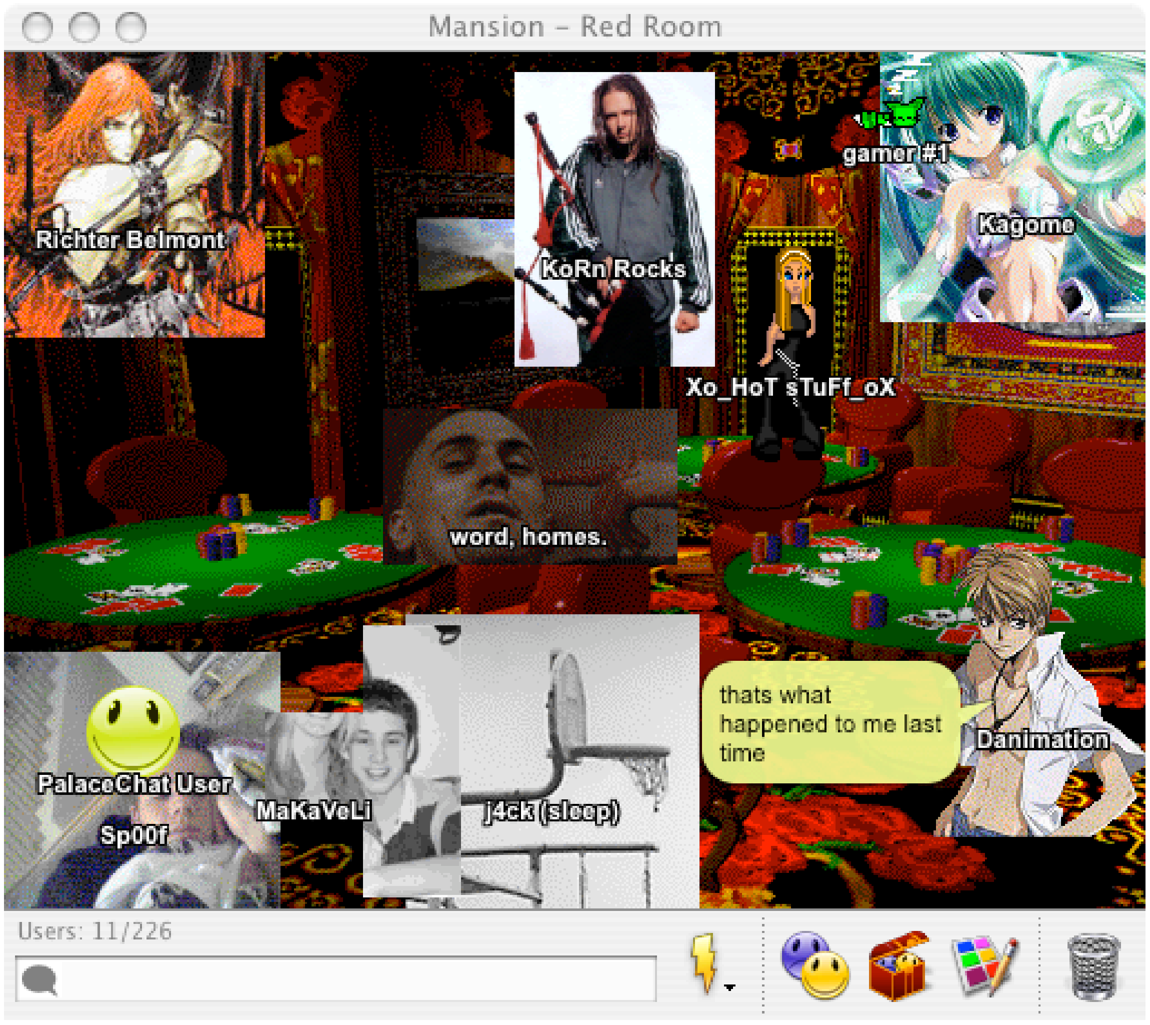


Figure 2.3: The Palace chat room. Participants are represented by avatars, and speech is shown in cartoon bubbles.

2D representations will usually consume more screen space than analogous 1D representations. To alleviate screen real estate limitations when presenting large representations, viewports and distortion effects [47] are often employed to allow the viewing of a small portion of the space in greater detail.

Graphical chat rooms are a popular example of 2D representation of community. In The Palace[1] (Figure 2.3), chat participants are represented by avatars in an attempt to convey identity or personality. Chat Circles (Figure 2.4) [59] represent the dynamics of conversations by placing text within coloured shapes. The shapes

[1]The Palace `http://www.thepalace.com`

and colours are chosen both to convey the flow and rhythm of the conversation and to identify the sender. While all participants are represented in the graphical space, a simple proximity-based filtering is implemented. Only those participants who are close to each other see each other's messages. In VChat [51], text chat is supplemented by the use of avatars in an attempt to simulate facets of face-to-face conversations such as gaze and gestures.

Donath's Visual Who [15] shows the presence and associations of virtual community members (Figure 2.5). The names of people in the community are positioned according to their relationships with various community areas of focus. The more strongly affiliated they are with a particular area, the closer they will be to its location in the space.

3D Representations

Three-dimensional (3D) representations extend spatial representations by introducing depth in addition to X and Y axes. While 2D representations provide an overhead view of data, 3D representations allow the observer to navigate around or even inside the data space and observe it from different viewpoints. The field of data mining grapples with the task of presenting very large amounts of data to help a human comprehend it. The Bead [10] and SPIRE [56] systems present large numbers of text documents, arranging them in galaxy and landscape metaphors according to similarity.

Virtual environments (VEs) go one step further and create a setting that mimics reality, presenting users with virtual rooms and landscapes. Multiplayer games such as id Software's Quake series[2] (Figure 2.6) and Sony's Everquest[3] are recognizable examples of online communities represented in 3D space; however, numerous research projects have investigated the use of VEs. In Collaborative Virtual Environments (CVEs) [5] researchers study embodiment design issues such as presence, identity, activity, and availability [4]. Avatars representing CVE inhabitants may even make

[2]id Software `http://www.idsoftware.com/`
[3]EverQuest II `http://everquest2.station.sony.com/`

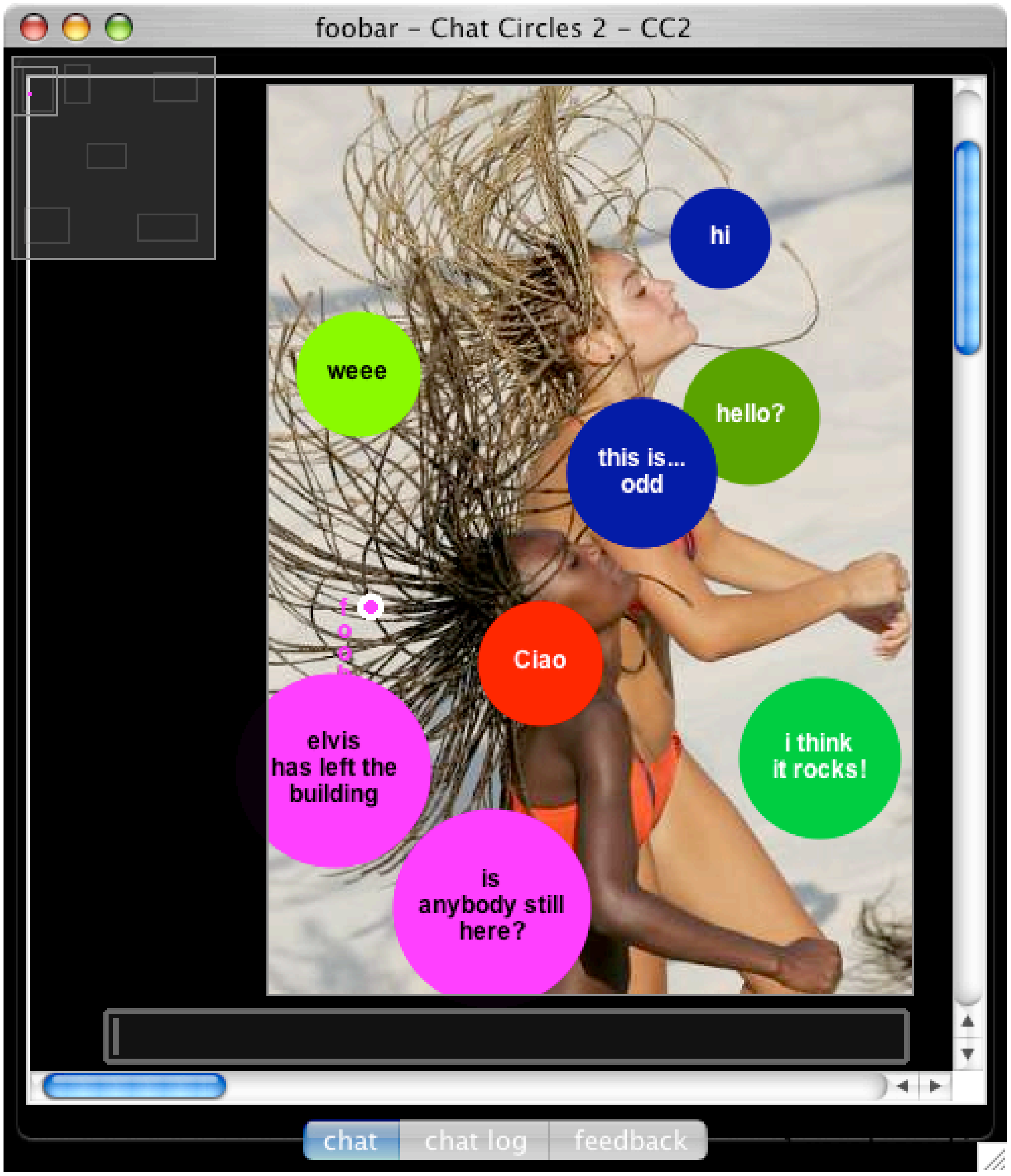


Figure 2.4: Chat Circles. Text is displayed in coloured circles to show the flow and rhythm of the conversation. Circle colour identifies the contributor of the text, and circle size indicates recency, with larger circles representing more recent contributions.

use of enhancements such as facial expressions and gestures. CVE inhabitants need not be created equal. For example, there may be roles assigned such as teacher and student in a learning environment, with each role having its own unique views and powers [53].

One of the benefits of inhabiting a CVE is that, like the real world, users are aware of each others' presence and actions and can engage in chance encounters and unintended interactions that are beneficial and may not have happened if not by accident [30].

While CVEs mostly approximate the experiences found in the real world, designers may introduce special enhancements or "magic" [52] that have no basis in reality but help deal with interaction limitations. Magical features tend to increase the power available to the user at the expense of decreased learnability, as these features may not be obvious to novice users. Learnability is not the only aspect that is sensitive to magic. In multiscale CVEs [64], users may resize themselves to work on different sizes of a structure—small users work on a small part in great detail, while large users work on the entire structure. Manipulating one's own size in such a way, however, raises issues of social presence, social distance, and proximity, as size may also convey one's importance or status, small users may wish to stay at a safe distance from large users, and large users may appear closer to the observer than they really are. When introducing features such as this, designers should be conscious of possible unintended consequences and evaluate whether the benefits outweigh the risks.

Croquet [50] is an architecture for collaboration using replicated versioned objects in a 3D medium. Participants introduce their own objects to the environment or are able to manipulate objects contributed by others, and are additionally able to take part in voice communications with others. Participants are also able to view others as they perform work. Figure 2.7 shows a user's drawing session presented in a 3D space, where it may be viewed by other users.

The mechanisms of focus and nimbus [22, 45] act to regulate the flow of information from the observed to the observer. Focus is the means by which an observer controls the view direction and how closely to look at objects in that direction. Nimbus is the means by which an object controls how much information about itself an observer will be allowed to receive. By using focus and nimbus, both the observer and the observed respectively are able to regulate the amount of information passing between them.

When deciding between 2D and 3D representations, the increased realism of 3D is offset by a number of factors. First, some users experience "cybersickness" similar to motion sickness when interacting with 3D representations [31]. Ranging from headaches and eye strain to disorientation and vertigo, symptoms may occur both during and after interacting with virtual environments. Second, interactions in a 2D representation tend to be less awkward than in a 3D representation due to limitations in supporting the 3D environment with current interface technology, although newer developments with vision and manipulation devices attempt to improve 3D interactions [1, 19]. Finally, there is debate whether 3D displays of data necessarily are better than 2D displays, or if in fact 2D displays are in some ways better. Thorndyke and Hayes-Roth suggest that 2D displays give a map-like view of data and are better for navigation[57]. The purported advantage that 3D displays utilize spatial memory better than 2D [55] has also been called into question by Cockburn [13], who suggests that 2D displays are just as effective. Of course, the third possible choice is to use a combination of 2D and 3D displays to gain the benefits of both types of display [58].

2.2 Human Limitations in Processing Visual Displays

The study of human limitations is motivated by the need to know what factors contribute to user overload when processing visual displays of information. Knowledge of these factors makes two things possible. First, designers can use this knowledge in an effort to create visual displays that do not lead to overload by staying within the boundaries of human perception. Second, designers can exploit human limitations to create simpler displays that are indistinguishable from more intricate ones, thus cutting system load. A summary of the stages in which humans process information is as follows [62]:

- *Sensory processing.* Sensory inputs (sight, sound, touch, taste, smell) are collected and temporarily held in a short term sensory store (STSS). The length of time an input is stored depends on its type: visual stimuli are held for approximately 0.5 seconds, and auditory stimuli for between 2 and 4 seconds. When these stimuli are not interpreted during these storage periods, they will be discarded.

- *Perception.* Sensory inputs are interpreted quickly and automatically using knowledge stored in long term memory. Perception is enhanced when two or more stimuli are sampled at once. When these stimuli vary in correlation (parallel) with one another and provide redundant information, reliability is increased. When these stimuli vary independently (orthogonal) of each other the effective bandwidth of this stage increases and more information may be successfully processed.

- *Cognition and memory.* This process is much like perception except it requires effort and attention, and requires more time. The result of this learning is entered into long term memory. Cognition and memory is prone to disruption and uses working memory which is limited to holding approximately seven items at a time. Chunking (combining several items together to treat them as one) reduces the problem, and improves with experience and familiarity.

- *Response selection and execution.* After understanding the current situation based on the previous three steps, an appropriate response is chosen and executed. If the situation changes too rapidly, responses will lag behind and may end up being inappropriate.

- *Feedback.* After executing the response, the result of those actions is sensed. The time difference between actions and perceived results is critical. When it is too long (long beta), the person is not able to adapt to a change in the environment.

- *Attention.* Senses are directed to areas that are judged to be interesting or important, based on the processing of earlier inputs. When many areas are simultaneously interesting, senses may be diverted from one or more important inputs.

The model just described reveals several causes of overload and hints at techniques to avoid it. A simple example is the disruption of the sensory processing stage by presenting too many stimuli in a short period, overflowing the STSS and resulting in some stimuli being completely undetected. A more complicated example is the disruption of the attention stage, where several types of failure are possible:

- *Failure due to preoccupation.* A human operator becomes overly focussed on one or more sensory inputs, and overlooks an important event outside the area of focus.

- *Failure due to selective attention.* A human operator focused on some portion of the environment fails to notice some important event elsewhere.

- *Focused attention failure.* A human operator is distracted from an important activity by the arrival of less important information, such as a car driver being distracted by his/her cell phone ringing while driving on a busy freeway.

- *Divided attention failure.* A human operator, while attempting to pay attention to two or more inputs by switching focus between them, misses an important event at an input that is not currently being attended to.

Researchers have studied a number of ways of managing perceptual and cognitive limitations. One method of increasing the flow of information to a human while avoiding overload is to use both visual and non-visual pathways. Since people are able to see and hear at the same time, sound may be used in addition to sight to deliver information [6, 7]. The use of earcons, or auditory cues, can be particularly helpful, as people need not be paying attention to a computer screen to hear them and they may convey important, timely messages that may otherwise be lost in a visual mess.

Overload can result simply by adding more users or objects to an environment. In a study of the performance of virtual helicopter pilots [28], these pilots were prone to failure in situations when their environment became so congested and they became so preoccupied with their surroundings that their performance deteriorated to the point they would crash their aircraft, just as real-life pilots might. The solution to this problem was to present other aircraft to the pilots as groups of aircraft rather than as individuals. The problem then became how to create and maintain these groups. For instance, group creation could be done automatically by grouping aircraft in proximity to one another, or manually according to aircraft purpose.

Overload can result from having too much choice. Patrons of online marketplaces [25], when faced with the task of selecting the best item from a list of several, can commit errors such as not finding the best price or not locating all available items for sale. Changing the presentation of candidate items from a breadth-oriented design that presents many items at once to a depth-oriented design that presents categories of items and forces the buyer to navigate into subcategories to view smaller lists of items appears to reduce error rates by reducing the possibility of user overload.

There are ways to exploit human limitations rather than treat them as a problem. By exploiting change blindness [8] and inattentional blindness [9], researchers were able to replace portions of a rendered scene with lower fidelity portions without human observers noticing. The result was a reduction of system overload, as lower fidelity components were less expensive to create, transport, and present to the observer.

In this thesis, the approach to reduce the likelihood of human overload is to filter extraneous information and allow users to receive and focus on information that is relevant to them.

2.3 Distributed Groupware

Since a MOA is an instance of distributed groupware, it may borrow ideas from and be designed similarly to other groupware systems. The following is a brief survey of the principal classifications of groupware, and an examination of some of the possible architectures that may be used to develop groupware.

2.3.1 Groupware Classifications

MOAs may be classified as distributed groupware. Groupware is a term used for software applications that allow two or more people to work together [17], while distributed groupware allows people to work together while using different computers, possibly at different locations [44]. The field of Computer Supported Cooperative Work (CSCW) is a multidisciplinary field that explores groupware and how people use computers to work together in general [23].

Groupware applications may be broadly divided into two groups: pure communications tools such as chat [59], and shared object applications where users interact with each other and jointly manipulate artifacts. Examples of the latter group range from collaborative writing applications [3] where many users work together to create a document, to popular multiplayer games where the shared artifact is the virtual world where players hunt each other down. Another way to partition groupware is into synchronous groupware, where users are able to interact with each other or share resources in real time, and asynchronous groupware where users take turns [44]. While it may be possible to build a MOA with asynchronous properties, this thesis focusses on the qualities of a synchronous MOA.

Computer Supported Collaborative Learning (CSCL) is a research area that has explored the concept of incorporating community into specialized systems for learning. Intelligent Tutoring Systems (ITS) designers have developed systems that treat students using the system as a community. The COMET system is a collaborative ITS for medical problem-based learning [54] that provides facilities for students to communicate and work on a common problem. In COMET, students learn both from the automated tutoring module in the system and from each other, as they are able to discuss the problem with others and gesture over and annotate shared artifacts such as medical images. Another system, Livenotes [32] is a system for collaborative note-taking by students in a classroom setting, in which students with networked tablet computers use a shared virtual whiteboard to view and annotate lecture slides.

2.3.2 Groupware Architectures

Much research has been devoted to defining unique architectures to address the needs of groupware designers [2, 39]. For MOA research, architectures for distributed synchronous groupware that utilize shared artifacts are of particular interest. Such architectures will feature a network communication component to link physically separated users, and also a component that maintains the shared resources that users are able to access; this component will manage the changes that occur when users manipulate these objects. The exact nature of these two components will be determined by the features provided by the application.

Patterson presents a taxonomy of synchronous groupware architectures. He identifies ensuring state consistency as "the primary challenge for synchronous groupware applications" [37, page 27] and proposes three ways to do so. The first way is that all users share a single copy of the state (shared state architecture), the second is that each user will have their own copy of the state and keep it in agreement with other users' states (synchronized state architecture), and the third is a combination of the previous two (hybrid architecture). He then goes on to divide the application state into four levels [37, page 27]:

> The display state is the information that drives the user's display; the view state is the information that relates the user's display to the underlying information in the application; the model is the underlying information; and, the file is a persistent representation of the underlying information.

Each user will have their own copy of the display state. In a shared state architecture, users may share all three of the remaining states, or maintain their own view states and share only the model and file states, or maintain their own view and model states and share only the file state. In a synchronized state architecture, users maintain their own copies of all four states and keep synchronized with other users through at least one layer. For instance, users may keep their model states synchronized and as a result of this also keep their file, view, and display states synchronized.

Patterson's shared state architecture is essentially a client-server architecture, with the server maintaining state information, and clients accessing it. Another variation of the client-server approach is the notification server [38]. The server in this case does not maintain state information, but acts as an intermediary. When a client requests information from the notification server, the notification server forwards the request to a client that is able to provide a response, and that response is then passed back to the requesting client. Similarly, when a client modifies state information, notifications of the change are sent to all other clients through the notification server.

Lauwers and Lantz [34] classify groupware as being either collaboration-aware or collaboration-transparent. In collaboration-aware applications, users are explicitly aware of the presence and activities of other users and may have to react in response to those activities. Collaboration-transparent applications, in contrast, will look and behave much like single-user applications, as the presence and activities of other users will be less noticeable, with the application automatically responding to the actions taken by them. They go on to suggest that while many consider it desirable to produce collaboration-transparent applications, it is necessary to develop collaboration-aware systems software to do so.

Synchronous groupware that permits several users to modify the same shared objects is complicated by the need to arbitrate and resolve simultaneous changes. Munson and Dewan [36] present a concurrency control framework that provides multi-granularity locking to accommodate the varied needs of groupware applications. Prakash and Knister [40] tackle the problem of undoing actions performed by multiple users on shared objects. Their framework allows both per-user undo, where users can undo only their changes, and global undo, where users can undo any other user's changes. The framework also provides selective undo, where users can undo not only the last change made, but one or more changes previous to the last. Two problems uncovered were that sometimes undo operations were not allowed, and the effects of an undo operation did not always correspond to what the users intended to perform. Greenberg and Marwood [21] examine concurrency issues and the effects concurrency control mechanisms will have on the user interface. One of their recommendations is that human-mediated concurrency control, letting the users solve conflicts by themselves, may be considered as an alternative to system-mediated control. In the research conducted in support of this thesis, the prototype MOA does not support several users modifying shared objects. Each user is able to modify his/her own object, while others may only watch.

2.3.3 Real-Time Multiplayer Networked Games

Designers of real-time multiplayer networked games have been able to successfully create virtual worlds inhabited by players who may be separated by great physical distances, and in doing so have presented a number of lessons for MOA designers who will want to achieve similar goals. One of the most significant challenges these designers face is to compensate for network and computer system limitations to create the illusion that all players are sharing a single, immensely powerful computer rather than their individual, less capable computers connected by an often-congested and delay-ridden network.

Smed et. al. [49] present three fundamental techniques used in games:

- *Make the best use of network resources by compressing information and aggregating several messages into a single message.* Simply put, messages are made as small as possible, and many small messages are combined into one large message to reduce the relative overhead of the message's header information.

- *Send only the information the receiver is interested in.* Smed refers to this as *interest management*, which is an embodiment of *focus* and *nimbus*. The information sent to a receiver is only what the sender is willing to send and the receiver is willing to receive.

- *Predict future behaviour of the environment based on past behaviour (dead reckoning).* In first-person shooter games, for example, the future position of a projectile may be calculated based on its current trajectory and the laws of physics. As a consequence, information about the projectile may be sent less frequently, and possibly as an unreliable but more lightweight UDP datagram, as the receiver of this information will be able to compensate for less frequent updates that are sometimes lost.

2.4 Filtering

Groupware and online communities have used filtering both to emphasize and de-emphasize. Email programs incorporate "spam" filters to identify and cast aside unwanted emails from questionable sources (Figure 2.8). The buddy list commonly used in chat programs (Figure 2.2) may also be viewed as a filtering mechanism as they identify a select few friends out of a mob of millions, although the effect provided by the buddy list is opposite to that of the spam filter in that it brings forward the good rather than pushing away the bad.

Another example that is similar to the promotional characteristics of the buddy list is the social networking phenomenon characterized by the MySpace web site[4], where musicians promote themselves by creating pages featuring samples of their music, video clips, photographs, and online diaries chronicling aspects of their work. They also use their pages to promote colleagues, as evidenced by Amanda Palmer's page (Figure 2.9)[5]. In the "Artists I Support" section, she has chosen to identify and promote eight artists out of the thousands who are present in MySpace, implicitly inviting her fans to investigate them as well. The "Friend Space" section shows eight out of the 19429 members who have applied for and received special privileges such

[4]MySpace `http://www.myspace.com`

[5]Amanda Palmer's MySpace page `http://www.myspace.com/whokilledamandapalmer`

as the ability to post comments in her photo gallery. Visitors to Palmer's site are also able to investigate these friends to possibly learn of even more artists who may be interesting, yet one may wonder if a group of friends so large would be a sufficient paring down of the MySpace population to be helpful.

Online chess clubs such as the Free Internet Chess Server[6] offer both the ability to promote and to exclude. Club members have the ability to create two lists. The first list is a friends list, which may consist of other members who are good opponents and/or courteous. Members are notified when those on their friends list arrive at the club or leave, or become available to play a game. The second list is an enemies list, which may consist of suspected cheaters, or members who are simply rude and annoying. Those on the enemies list are effectively shunned: they are unable to contact the owner of the list, and are also not able to tell when the list owner is present at the club.

[6] The Free Internet Chess Server `http://www.freechess.org`

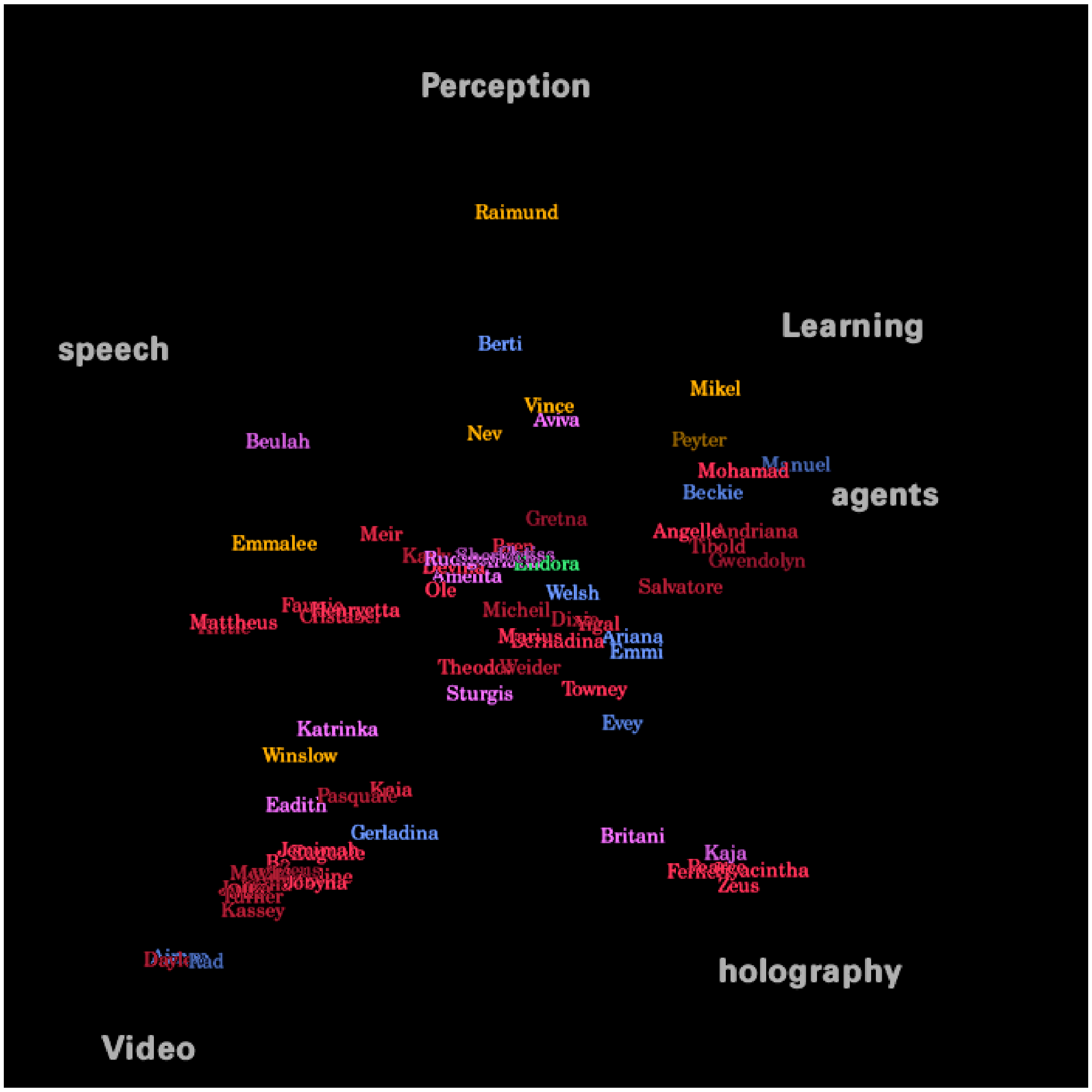


Figure 2.5: Visual Who. The location of people's names indicates their areas of focus.

Figure 2.6: Quake, a popular multiplayer game

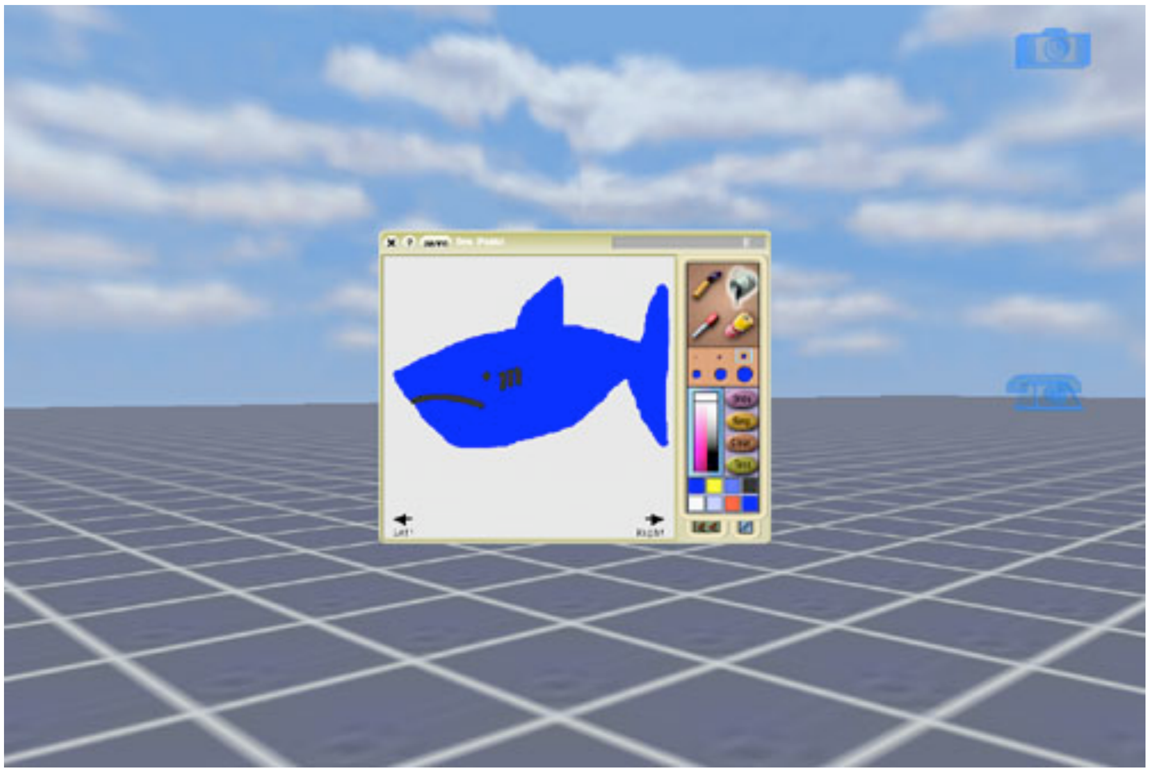
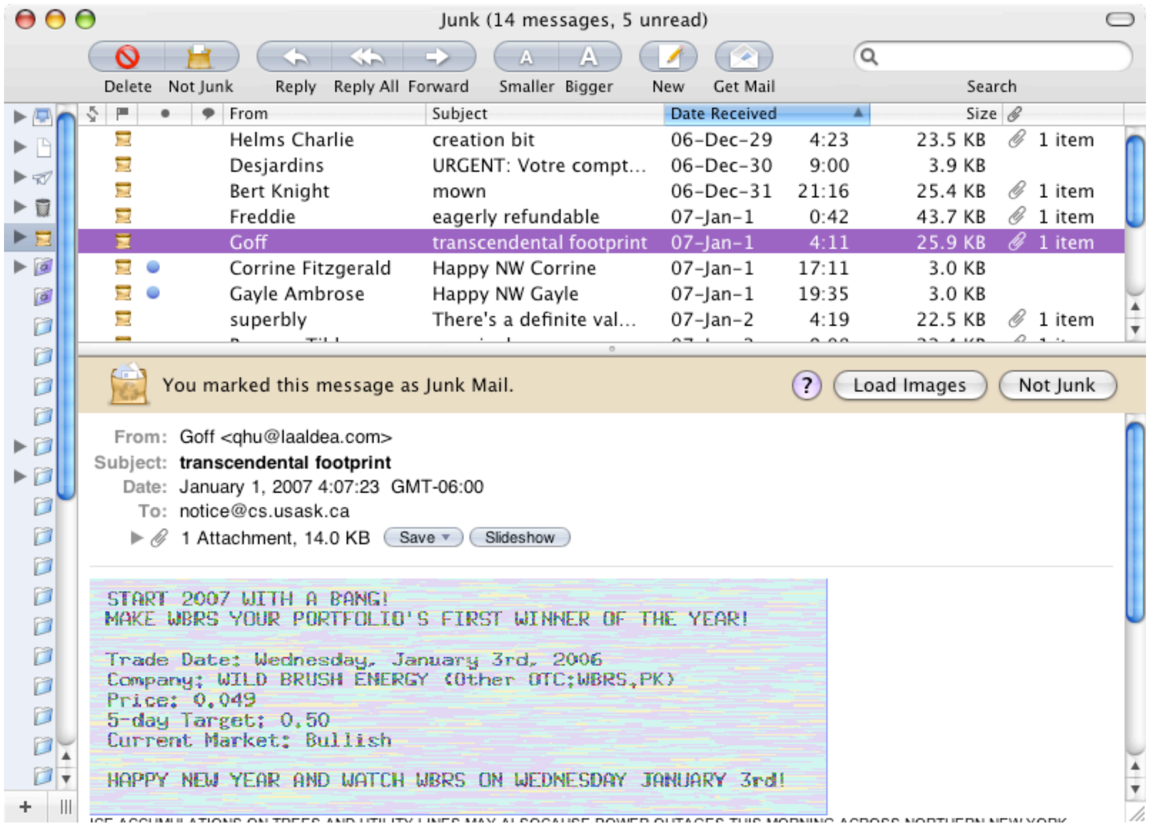


Figure 2.7: Croquet space presenting a drawing application



From	Subject	Date Received		Size	
Helms Charlie	creation bit	06-Dec-29	4:23	23.5 KB	1 item
Desjardins	URGENT: Votre compt...	06-Dec-30	9:00	3.9 KB	
Bert Knight	mown	06-Dec-31	21:16	25.4 KB	1 item
Freddie	eagerly refundable	07-Jan-1	0:42	43.7 KB	1 item
Goff	transcendental footprint	07-Jan-1	4:11	25.9 KB	1 item
Corrine Fitzgerald	Happy NW Corrine	07-Jan-1	17:11	3.0 KB	
Gayle Ambrose	Happy NW Gayle	07-Jan-1	19:35	3.0 KB	
superbly	There's a definite val...	07-Jan-2	4:19	22.5 KB	1 item

You marked this message as Junk Mail.

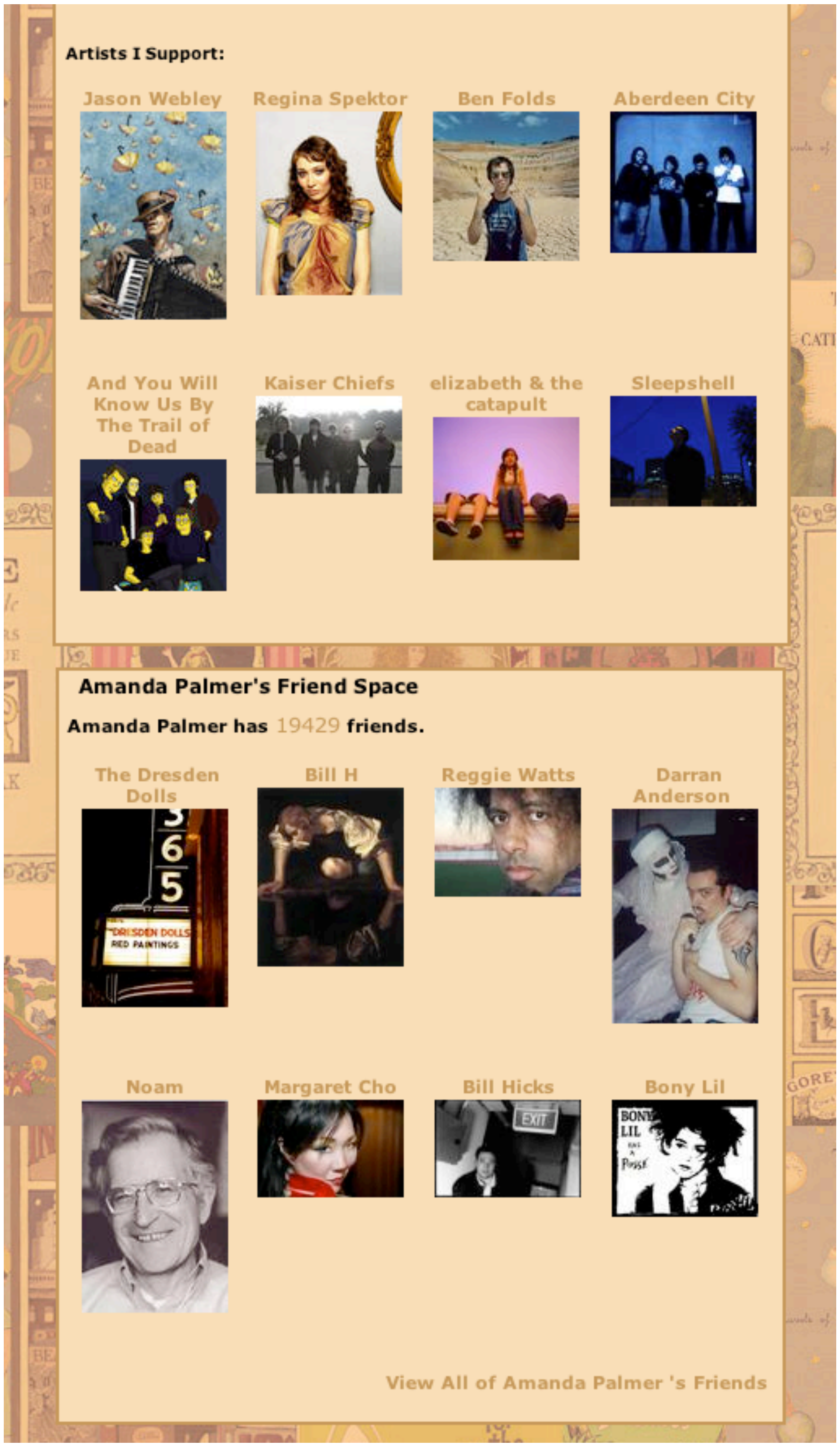
From: Goff <qhu@laaldea.com>
Subject: **transcendental footprint**
Date: January 1, 2007 4:07:23 GMT-06:00
To: notice@cs.usask.ca
1 Attachment, 14.0 KB

```
START 2007 WITH A BANG!
MAKE WBRS YOUR PORTFOLIO'S FIRST WINNER OF THE YEAR!

Trade Date: Wednesday, January 3rd, 2006
Company: WILD BRUSH ENERGY (Other OTC:WBRS.PK)
Price: 0.049
5-day Target: 0.50
Current Market: Bullish

HAPPY NEW YEAR AND WATCH WBRS ON WEDNESDAY JANUARY 3rd!
```

Figure 2.8: Junk mail, or spam filters identify and isolate emails that contain questionable content, or originate from unreputable sources.



Artists I Support:

Jason Webley
Regina Spektor
Ben Folds
Aberdeen City
And You Will Know Us By The Trail of Dead
Kaiser Chiefs
elizabeth & the catapult
Sleepshell

Amanda Palmer's Friend Space

Amanda Palmer has 19429 friends.

The Dresden Dolls
Bill H
Reggie Watts
Darran Anderson
Noam
Margaret Cho
Bill Hicks
Bony Lil

View All of Amanda Palmer 's Friends

Figure 2.9: MySpace members promote and grant special privileges to other members.

Chapter 3
MOAs

This chapter delves further into the concept of MOAs. The chapter begins with a discussion of ChessWorld, a visual display of an online chess community that was the precursor to the research undertaken for this thesis. Next is a description of a MOA from the standpoints of users and systems. To a user, a MOA is an application that enables observation and interaction with the application's user community, and the capabilities and qualities necessary to provide this functionality will be discussed. At a systems level, bringing user community to an application mandates additional obligations of both the application and the infrastructure supporting it. The chapter concludes by presenting SketchWorld, the prototype designed, built, and evaluated for this thesis. SketchWorld is a concrete example of a MOA, enabling exploration of the possibilities and challenges of incorporating user community.

The following definitions shall be used in this and subsequent chapters when discussing MOAs:

- **Application State:** an artifact that a user is working on. Depending on the application, it may be such things as a drawing, spreadsheet, or a letter.

- **Station:** representation of a user's work in the community. This representation may include user's identity, state of work, or current activity, and position within the MOA space. Many stations may be attributed for a single user. For instance, a user may be working on several drawings at once, and each drawing will have a unique station representing it.

- **Quality:** properties that describe how faithfully a station represents a user's work to another user. A high-quality representation may be large, detailed, and frequently updated, while a low-quality representation may be small, sparse, and infrequently updated.

- **Community View:** the representation of community to the user. Within this view the user is able to see the stations representing the work of all users.

3.1 ChessWorld, a Visualization of an Online Chess Community

ChessWorld (Figure 3.1) is a visual display of the games being played on the Free Internet Chess Server (FICS)[1]. FICS claims a registered membership of over 136,000 players and in addition allows unregistered players to login and play as guests. Typically between 200 and 800 players will be logged in, and between 50 and 200 games will be in progress at any time. ChessWorld may be considered as only part of a MOA, as it provides the facilities to watch others play chess, but does not provide the facilities to play games or to chat with others who are logged in.

One of the findings of the ChessWorld study was that at times it was difficult to follow a game because the activities on nearby boards were distracting; it was this finding that spurred interest in user overload, and later in developing techniques to avoid overload. It was decided not to use ChessWorld as the basis for further study, however, for three reasons:

- *Playing chess involves a limited range of actions by the user.* The sequence of actions consisting of moving a chess piece from one square to another, waiting for an opponent to move, and after some thought, moving another piece is very restrictive when compared to the actions taken by a user to create a drawing.

- *The amount of data needed to store a chess position, or a single move, is trivial with little variance.* It would have been difficult to generalize findings from the study of chess traffic to traffic for applications such as drawing programs.

[1]The Free Internet Chess Server `http://www.freechess.org`

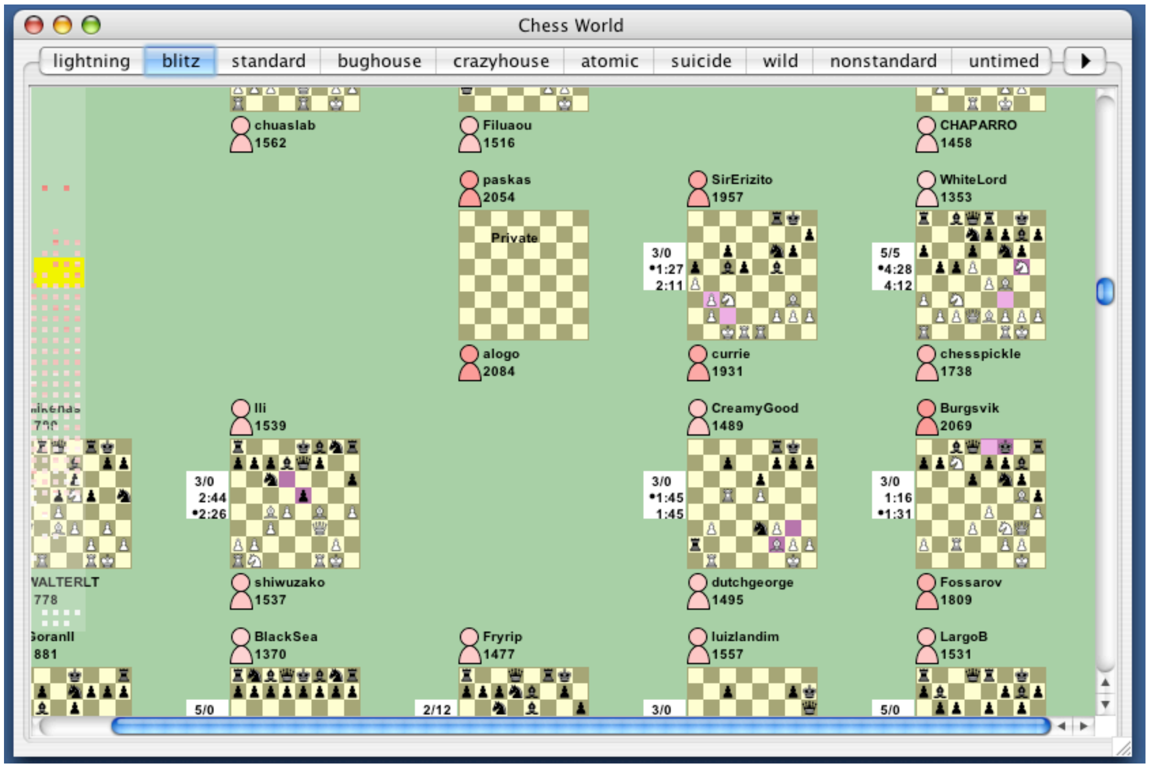


Figure 3.1: ChessWorld, a visualization of an online chess community.

- *The display of a chess position does not lend itself to being viewed at different qualities.* The range of qualities available (displaying the board as a dot, displaying an empty board, displaying a board with pieces) is very limited compared to the range available for displaying a drawing.

3.2 A MOA From a User Perspective

A MOA distinguishes itself from a traditional application by incorporating its user community. When a user launches a MOA, he/she will be able to use it to perform a task, such as drawing a picture or writing a letter, and will also be able to see other users do their own work. Users will also be able to communicate with each other with a facility as basic as text chat or as elaborate as a video or audio stream. Should the application provide the ability to work collaboratively, two or more users may agree to work on a shared artifact.

3.2.1 MOA Goals

The goals of a MOA are based on typical scenarios where it is anticipated they would be used. The community of users may number anywhere from fewer than ten to several thousand, although the community size considered for this thesis was at most a dozen members. There would be considerable variation in the relationships between users–some would know each other very well, perhaps sharing a common office space, while others may have never met either in the physical or virtual worlds. Users would have varying expertise in using the MOA's core application. The interest that users have in each others' work would likely vary according to purpose–some may be interested in others who are doing similar work, while others may search for someone doing something distinct or unique. In order to serve the needs of its users, a MOA should:

- *Instil a sense of community.* MOA users should feel they are part of a community and not isolated individuals. They should be aware of the size of the community and the nature of activities taking place within it. They should also be informed when others join or leave the community.

- *Maintain identities.* MOA users should be able to readily identify each other.
- *Enable finding.* MOA users should be able to search the community for specific users such as friends or colleagues, or users who are performing similar work.
- *Permit browsing.* A MOA should provide the user with the ability to roam through the community in search of users doing interesting or similar work. Although the community representation may arrange users in some way (i.e. according to experience or status), the arrangement may not naturally assist this search, so the user should be able to search the community in an unstructured fashion.
- *Provide focus control.* A MOA should allow the user to focus his/her attention on selected individuals and be able to disregard all others. The actions by the "unimportant" users should not distract the viewer. The user should be able to "tune in" to certain users while tuning out the rest.
- *Avoid overload.* A MOA should strive to avoid overloading the user and his/her computer with an excess of information. It should provide mechanisms to filter information that is not relevant or useful to the user. The problem of overload will increase with the number of users, as each user will be a potential source of noise and distraction.

3.2.2 User Interface

The user interface of a MOA will provide the user with a place to perform his/her own work and a place to view and interact with the community. One possible interface to satisfy these requirements consists of two separate areas, one for working, the other for community interaction. Another possibility is to offer a single area combining both work and community components. The MOA prototype SketchWorld employs the first method, while Figure 3.2 shows SlideWorld, a fictional slide show creation

MOA integrating working and community areas. In this figure, Robert is working on his presentation entitled "How I Spent My Summer" while monitoring the progress of Jacquie and Nicholas.

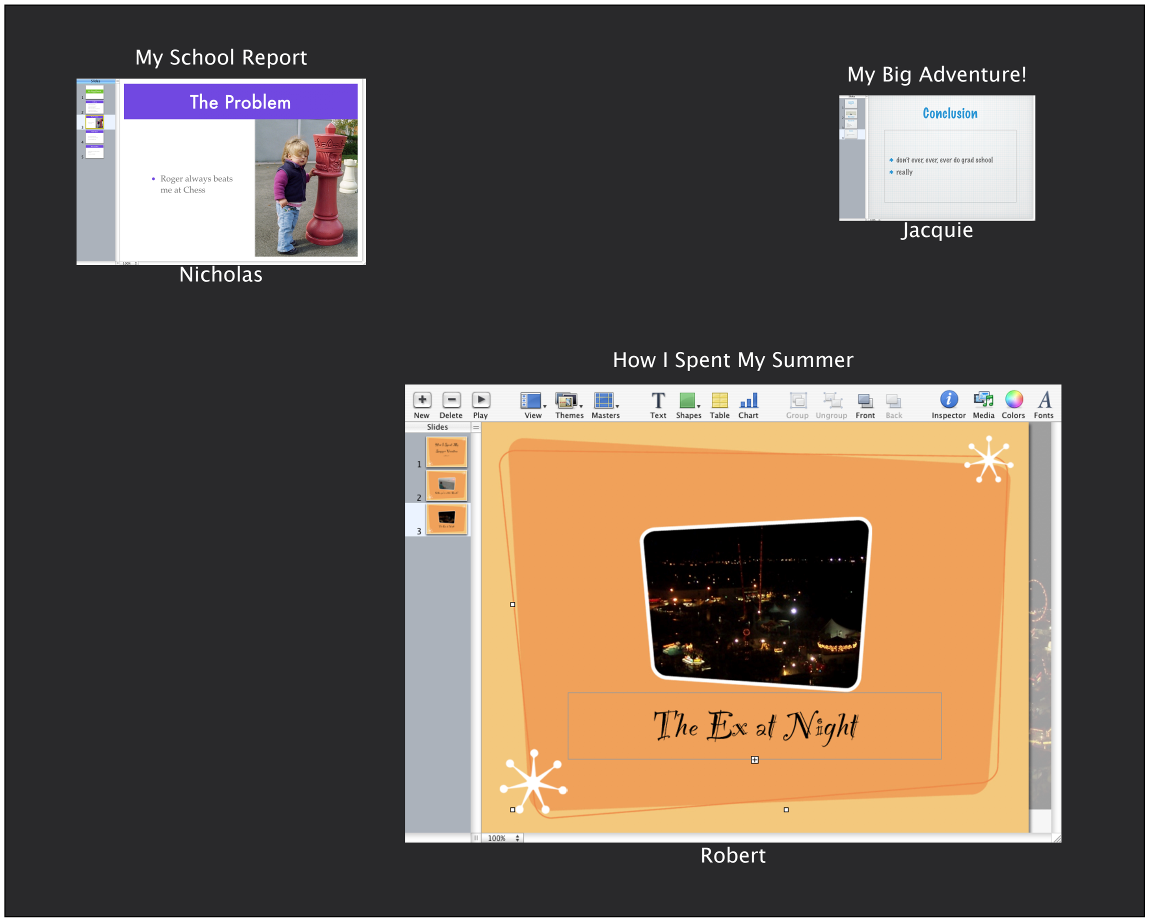


Figure 3.2: SlideWorld, illustrating one possible MOA user interface where users do their work within the community view.

3.3 A MOA From a System Perspective

The fundamental requirement for bringing community to an application is that information must be exchanged between MOA participants. This information is generated by its senders and transmitted over a network to its receivers, where it is displayed, interpreted, acted upon, and possibly stored for later reference. Closer study reveals that this information may be separated according to purpose into four classifications, each with its own distinct properties. The volume of information exchanged between MOA participants is affected by both characteristics of the application and the nature of its community. To counteract the burdens placed on systems and users by community-related information, filters may be introduced to selectively limit information volume in an effort to avoid system and/or user overload. The result of studying community-related information is a conceptual architecture identifying one possible set of components that could be combined to create a MOA.

3.3.1 MOA Data Types

The data that MOA participants will receive may be divided into the following categories:

- **Session management messages:** information about the users themselves, such as notifications when they join or leave the MOA, or change state (e.g. from "idle" to "busy"). These messages are sent by the entity that performs session management. In a client/server setup, the server would typically keep track of user logins and logouts, and broadcast notifications to all interested parties.

- **Application state messages:** complete renditions of the artifacts that users manipulate in a MOA, such as drawings, word processing documents, or spreadsheets, or incremental changes made to these artifacts. These messages make it possible for MOA users to observe each others' work.

- **Interaction with artifacts messages:** notifications of cursor movements, popup menus, and other tools and controls.

- **Interaction with people messages:** communications with other participants, such as text chat or audio or video streams.

- **MOA management messages:** messages exchanged by MOA clients. For example, the community rating filter relies on messages conveying the popularity of stations in the MOA.

3.3.2 Effects of Application and Community on MOA Data

The volume of MOA-related traffic will be influenced by both application and community characteristics:

- *Application artifacts.* Some applications such as drawing applications may manipulate large artifacts such as photographs or complicated artifacts such as drawings consisting of thousands of components, resulting in large application state messages being exchanged between MOA participants, while other applications may rely on small, simple artifacts, resulting in smaller application state messages.

- *Community size.* A large community will generate more MOA-related traffic than a small community.

- *Community activity.* A busy community will generate more MOA-related traffic than a less busy community. A busy community may be one where users frequently log in and out, causing increased session management traffic; it may be a socially active community where members frequently chat with each other; it may be an industrious community where users concentrate on their work, generating increased application state traffic.

As MOA-related traffic increases, systems will face increasing demands to generate, transmit, and process this traffic, and users will be increasingly burdened with interpreting the information presented to them. At some point, one or both of the following conditions may occur:

- Systems will become overloaded and application performance will suffer.

- Users will become overwhelmed with the overabundance of information presented to them.

3.3.3 Filtering

To reduce the possibility of overload, the approach studied in this thesis is to create filtering mechanisms that reduce the amount of data transferred between users, with the caveat that these mechanisms still allow the MOA to meet its goals. The central premise to filtering is that a MOA user will have varying degrees of interest in other users: most will likely be uninteresting, some will be moderately interesting, and a select few will be very interesting. The filters will reduce most data from uninteresting users, some data from moderately interesting users, and almost no data from very interesting users.

When deciding which types of messages to filter, one must consider characteristics of the messages themselves such as their size and frequency of transmission. It is also necessary to consider effects on the user receiving these messages, and the size and makeup of the community using the MOA.

Application state messages are prime candidates for filtering due to their size and frequency relative to other MOA traffic, and the relatively high costs of generating them at the sending end and processing them at the receiving end. The reception of these messages also significantly affects the user's perception of his/her community; filtering these messages would result in a noticeably less crowded and busy view of community. Filtering these messages is the focus of the thesis.

Interaction messages *may* be good candidates for filtering. While audio or video streams definitely are costly from a system standpoint and may likely be distracting to users receiving them, even chat messages can be problematic. When considering a community of several hundred or more users who tend to be highly communicative, the chat traffic they generate must be approached as a potential problem for both systems and users. For this thesis, filtering interaction messages was not studied since the target community for experimentation was a relatively small group of people who would likely be more focussed on performing their assigned tasks and less likely to engage in chat conversations.

Session management messages could be considered for filtering if they involve the exchange of rich profiling information for each user, or the community is sufficiently large. Consider the scenario of a MOA where a wealth of information such as a portfolio of a user's past work was made available to the community whenever that user logged in. In such a case, it may be more practical to send only this user's name to all others in the community when he/she logs in, and distribute the portfolio only to the few users who ask for it. For this thesis, filtering session management messages was not considered since user profile information consisted only of the user's name and identifying colour, and these messages would occur infrequently during the course of experimentation.

This section was a brief outline of filtering for the purpose of explaining its relationship with other components of a MOA. For a more detailed discussion, please refer to Chapter 4.

3.4 MOA Architecture

One conceptual MOA architecture consists of an application component providing the facilities for the user to do work and a community component responsible for the task of connecting to and presenting the community. Provided the division is suitably made, two advantages of this architecture are that the community component could be reused to develop MOAs centered around other applications, and existing "non-community" applications could be converted to MOAs with minimal intrusion.

3.4.1 Community Component

The community component brings MOA users together so they can view and join in to work with each other and to communicate. It assumes the responsibility of network management, deciding the qualities that a user will view other users, and presenting the community to the user. These responsibilities are handled by the following subcomponents:

- **Network Management.** The network management subcomponent is responsible for sending and receiving messages between MOA participants, as well as performing session management.

- **Controller.** The controller subcomponent is the manager of all other subcomponents and acts as the processing point of user inputs.

- **Database.** The database subcomponent keeps track of information necessary to observe and interact with the community. Information contained within the the database includes users who are currently logged in, data representing the work these users are performing, the qualities the local user wishes to see of other users' work, and the qualities that other users wish to see of the local user's work.

- **Community View.** The community view subcomponent is a graphical display of the activities of all MOA members, consisting of stations showing each member's work. There are three basic approaches for arranging these stations:

  - *In a one-dimensional (1D) list.* Stations are displayed in a simple list in order of a single variable, such as alphabetically according to user names.
  - *In two-dimensional (2D) space.* Stations are assigned x and y co-ordinates according to two variables and laid out as if they were laying on a flat surface.
  - *In a three-dimensional (3D) space.* In addition to x and y coordinates, stations are assigned a z, or depth coordinate according to a third variable.

  The community view need not be restricted to passive observation. The MOA designer may choose to make it a place for interaction, as was done in SketchWorld: users use the community view to change the location of stations or command them to become bigger or smaller.

- **Filtering.** The filtering subcomponent is responsible for controlling the qualities of the stations the user will view according to degree of interest or importance. Less interesting or unimportant stations will be throttled, while more interesting or important stations will be allowed to send larger, more accurate updates of their state to the viewing user. The filtering subcomponent will make these judgements based on a combination of user and system inputs.

3.4.2 Application Component

The application component's primary role is to provide the working area for the user, and can be considered roughly equivalent to a "communityless" application. For instance, it if were an image manipulation program, it would have the facilities for creating new images, modifying them, and saving them to files. The secondary role of the application component is to provide information for the community component. When one MOA user is being observed by others, the application component is responsible for generating application state information for the community component to send to the observers. In the example of the image manipulation program, the application component will generate representations of the images the user is working on so the community component can send them to observers. The makeup of these application state messages is application-dependent and may vary according to its quality. For instance, a full quality version of a drawing sent to observers may be an exact duplicate of the drawing being worked on, identical to the data structure composing it, whereas a low quality version may be a tiny bitmap image. The only requirement of what is sent is that the receiver must be able to decode and present it to the observer.

3.5 SketchWorld, a MOA Prototype

SketchWorld is the working MOA prototype developed to test the concept of MOAs and of filtering in MOAs, and was written in Java version 1.5[2]. The application component is a simple drawing program that allows the user to create drawings consisting of elements such as lines and rectangles. The community component joins the user to his/her community via a central server. Figure 3.3 shows the components

[2]Java Technology Home Page `http://java.sun.com`

that make up SketchWorld and their relationships with one another in terms of communication paths.

3.5.1 Community Component

SketchWorld's community component follows the general architecture given previously, consisting of network management, controller, database, community view, and filtering subcomponents. The following discussion deals with design decisions and implementation details of SketchWorld's community component.

Networking

SketchWorld follows a client/server network architecture. One user in the community is designated as the server, and all other users will establish a network connection to this server. The bulk of the networking subcomponent is provided by the GT Groupware Toolkit[3], a Java toolkit developed at the University of Saskatchewan Interaction Lab. GT performs most session management tasks, providing notifications when users log in and out, as well as maintaining a database of currently logged in users. GT also provides facilities for sending messages to and receiving messages from other users. The networking subcomponent acts as an intermediary between the network and the Controller subcomponent, relaying messages sent by the Controller to another Controller that is part of a remote instance of SketchWorld.

While Croquet [50] was another candidate for the basis of SketchWorld, the reasons for not choosing it are as follows. First, Croquet was in its early stages of development when the thesis research began and it was unknown whether the project would have been stable and usable at the time, much less if it were to continue to be developed. Second, Croquet's choice of a 3D representation conflicted with the direction of the thesis research, which equated filtering with altering the size of objects in the community view, a practice that would likely confuse users, who would often wonder if an object became less important according to the filtering

[3]GT Groupware Toolkit `http://hci.usask.ca/research/gt.shtml`

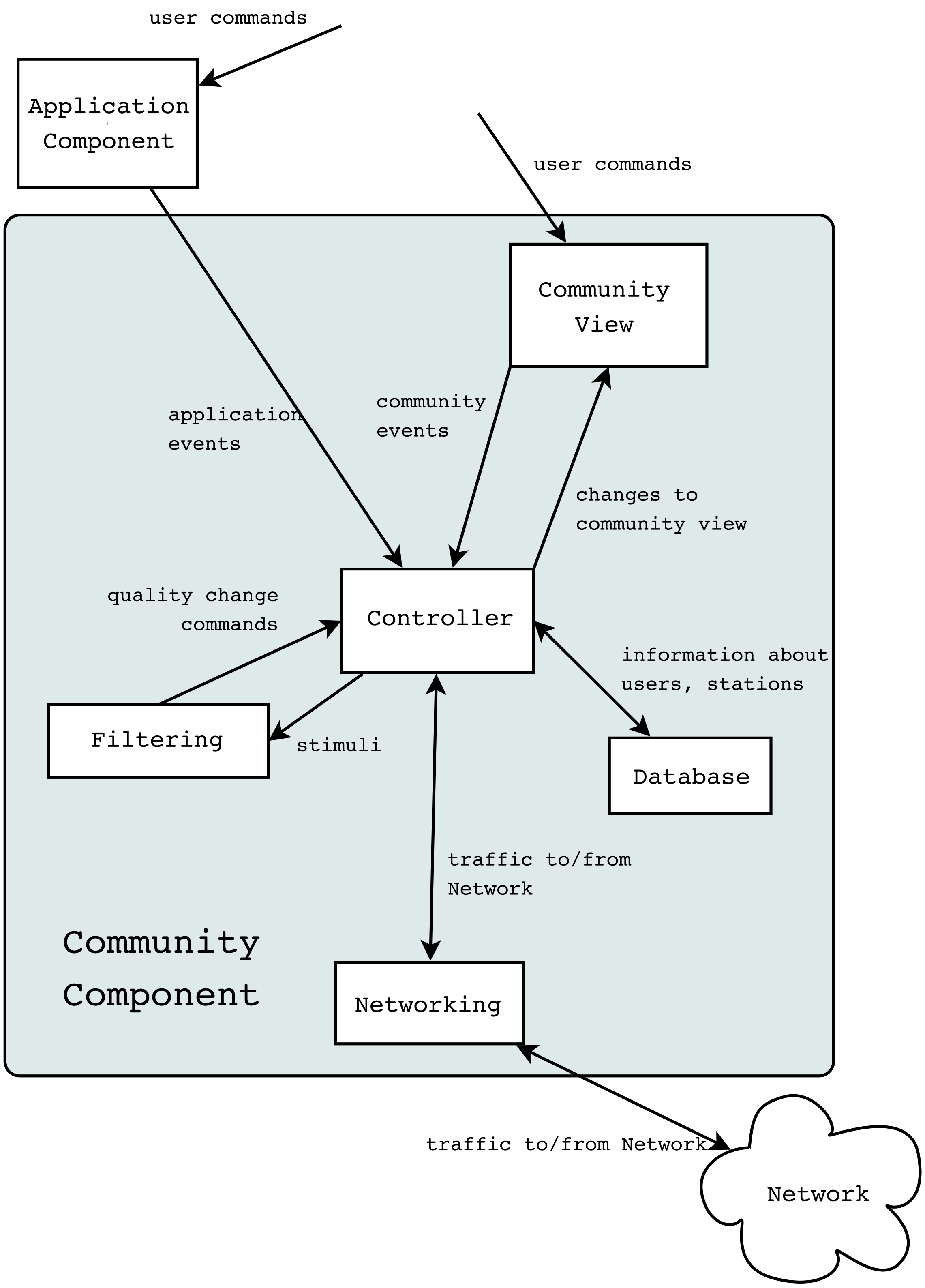


Figure 3.3: SketchWorld system architecture.

system, or if it had merely been moved to a more distant location by another user. Finally, there was a risk that developing the envisioned filtering system for use in Croquet would be intrusive and problematic, if in fact the Croquet architecture had allowed it.

Controller

SketchWorld's controller component acts as the central hub, bringing together the application's other components. The controller acts as a dispatcher for all events coming from the local user and from instances of SketchWorld running remotely.

One of the advantages of SketchWorld's design is that thanks to the interfaces provided by the controller and by other components, replacing components with newer versions would be relatively simple. For instance, only the controller deals directly with the GT toolkit communications facility, so if GT were to be replaced by another similar toolkit, the only changes required would be minimal and isolated to the controller.

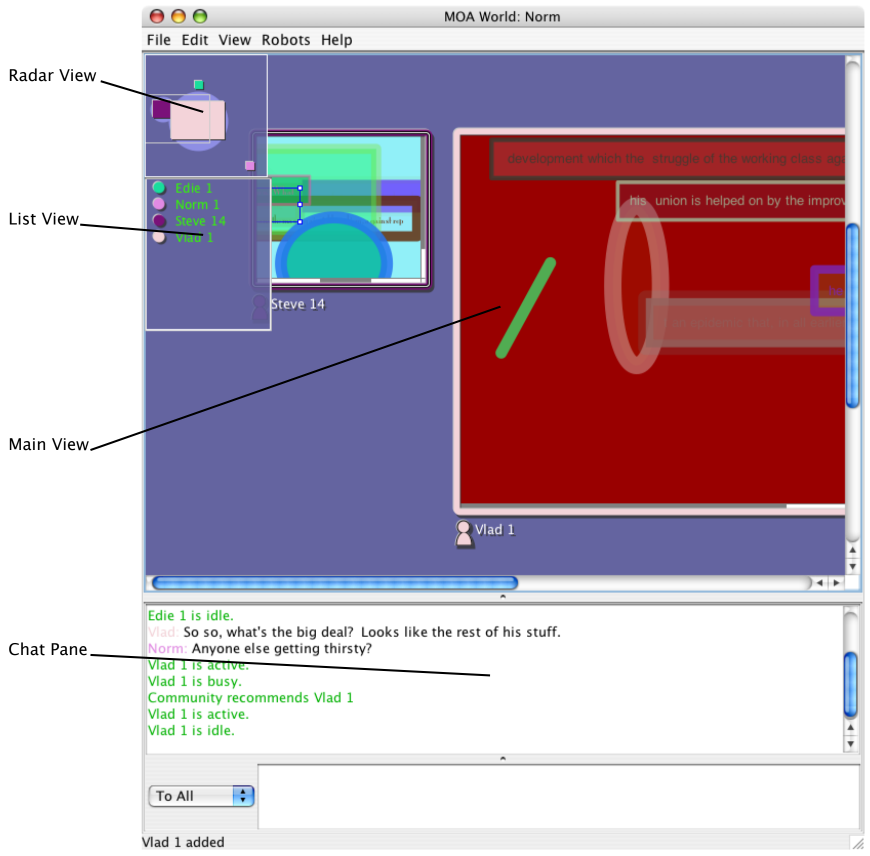
Database

The database maintains two stores, the first a collection of records of all users currently logged in and the second a collection of all stations associated with these users. Each user record consists of information such as the user's name, colour, the drawing tool currently in use, and references to stations owned by that user. Each station record consists of the following:

- the station's application state.
- the quality at which the local user wishes to observe the station.
- if the station belongs to the local user, the qualities at which other users wish to observe the station.

Community View

The community view consists of four components (Figure 3.4): the main view, where users observe each other; the radar view, providing a view of the entire MOA space with minimal detail; the station list view, providing a list of all stations in the MOA; and the chat pane, where users may converse with one another or follow others' conversations. The community view receives commands from the MOA user and passes its interpretation of those commands to the controller; in return, it receives change information from the controller and updates the user's view of the community.



Edie 1 is idle.
Vlad: So so, what's the big deal? Looks like the rest of his stuff.
Norm: Anyone else getting thirsty?
Vlad 1 is active.
Vlad 1 is busy.
Community recommends Vlad 1
Vlad 1 is active.
Vlad 1 is idle.

Vlad 1 added

Figure 3.4: Community View, consisting of the main view, radar view, list view, and chat pane.

The main view displays stations, with each station featuring the drawing being worked on, with the name of the user who owns the drawing and an avatar figure below. The drawing border and avatar figure bear the colour assigned to the user.

The main view typically will present a small region of the entire community, and the user may navigate to a different region using its scroll bars.

Within the main view, users may perform the following operations: they may change the size of a station by selecting it and pressing the "+" key to make the station larger, or the "-" key to make it smaller; they may move stations to different locations in the view by dragging them.

The radar view is a small scale representation of the entire community. It displays the location and relative size of all stations in the community with rectangles of the users' colours. It also shows the current region viewable in the main view (view rectangle). Users may change the region visible in the main view by clicking or dragging on the radar view.

The list view is a list of all stations in the community. Each station is represented with the user's colour, user's name, and station name. By clicking on a station in the list view, the main view will reorient to show the station in the center.

The chat pane consists of two text areas. The upper area displays the chat conversation underway, while the lower area is used for entering text. In SketchWorld, chat was implemented as a group conversation—all users will see what all other users have typed; future versions of SketchWorld could provide the ability for single user to single user chat.

Station Placement When a user first logs in to SketchWorld, a randomly assigned location in the MOA space is generated and recorded for that user. All stations created by the user will be located at a random distance close to that point. The user may at any time elect to move his/her stations within the MOA space by dragging them in the main view. The location that a station occupies is global–to each user, each station will appear in the same location in the MOA space. With all users presented with a common arrangement of stations, the locations of stations may be used to convey meaning such as status (e.g. experienced users are near the top of the world) or type of activity (e.g. artists specializing in black and white drawings reside in the lower left hand corner); additionally, users are able to use location of a station when conversing with others ("Have you seen the drawing near the top right hand corner of the world?").

During early development, SketchWorld would allow users to move all stations, not just their own. During preliminary testing, however, many testers complained that others were moving their stations, so the policy was narrowed to grant movement rights only to station owners.

Community View and Working View With the community view displaying not only all remote stations but all local stations as well, it was possible to have users use their local stations for doing their work, as an alternative to having working views in windows separate from the community view. The advantages of using local stations as the work spaces are as follows:

- *Users would have fewer windows on their desktop.* In the case of SketchWorld, there would be one window for both working and viewing as opposed to having one window for the community view and one window for every drawing being worked on.

- *Users would not be confused where to do work.* If work spaces were in separate windows, users might be confused and try to do work on their stations in the community view.

- *Users would not have to switch between windows when they switch between working and viewing.* They would not need to hunt for smaller windows partially or fully obscured by others.

The advantages of having work spaces in separate windows are as follows:

- *The community view would be less crowded.* Users would likely increase local stations to full or nearly full quality when they worked on them, leaving less room for remote stations.

- *Users would use the community view for the sole purpose of viewing others.* They would not need to navigate through the view from between interesting remote stations and their own local stations as they alternated between watching others and doing work.

- *There would be less confusion when performing operations.* Users would work on their drawings in the drawing windows, leaving the community view for such tasks as moving stations to different places and increasing and decreasing station qualities.

For SketchWorld, it was decided to use separate community and working views, although experimental evidence (refer to 5.1.2) suggests integrating work spaces into the community view may lead to a more natural interaction, at least under some circumstances.

Filtering

SketchWorld's filtering subcomponent receives stimuli from the controller; these stimuli are events that the individual filters within the subcomponent act upon. In return, the filtering subcomponent sends commands to change station qualities to the controller.

3.5.2 Application Component

The application component of SketchWorld (Figure 3.5) is a drawing program consisting of one or more windows to draw in and a palette window holding drawing tools. The user composes pictures consisting of elements which may be lines, simple shapes, text, and bitmap images. The user is able to move, edit, or delete these elements once they have been added to the drawing. When the user makes changes to his/her drawing or performs other actions, the application component informs the controller within the community component.

Design Goals

There were two sets of design goals for the application component. The first set was focussed on how the application component would appear from a user standpoint, while the second set was focussed on how the application component would appear from a systems standpoint.

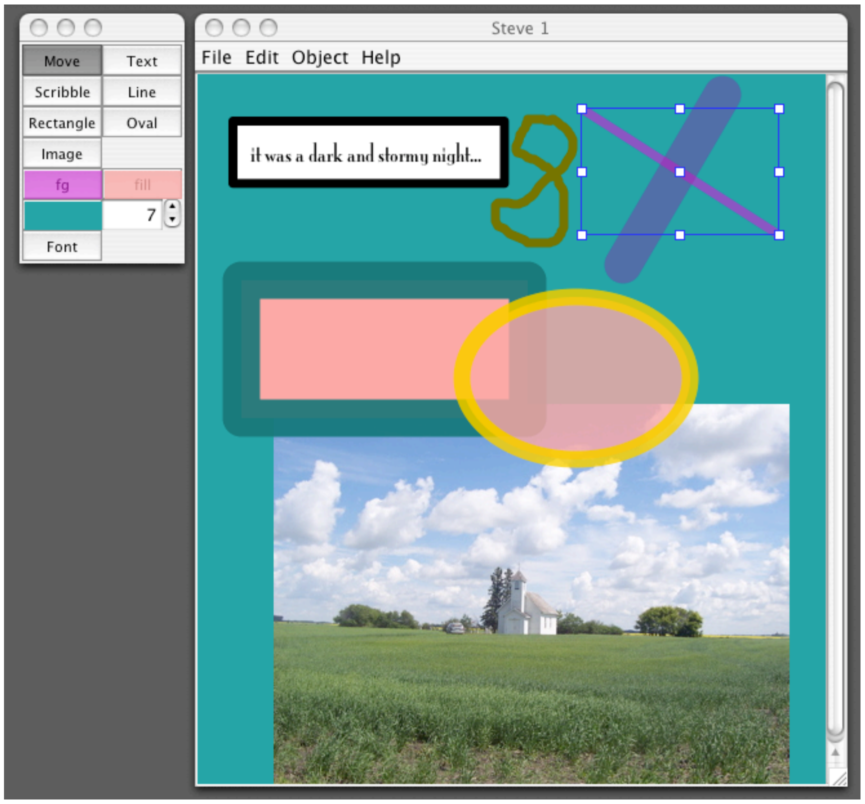


Figure 3.5: The application component of SketchWorld, a drawing application. The user selects the tool to use in the palette window and uses it in the drawing window. The drawing contains samples of all possible element types.

First and foremost, SketchWorld was to look and behave much like other drawing applications, so users would need little time or effort to learn it, and could expect that SketchWorld would offer features that were fundamental to other drawing applications. Using SketchWorld, users would be able to produce reasonably sophisticated drawings that would be easily distinguishable from drawings produced by other users. It was hoped that SketchWorld would be substantial enough to keep users busy and engaged during the experiments in which they took part, and that they would work on their drawings in a manner resembling how they would work on them using other drawing applications.

From a systems standpoint, users of SketchWorld were to be capable of producing drawings that taxed systems to various degrees—it should have been possible to produce drawings ranging from trivially small to large enough to trigger system overload conditions.

Application Component Capabilities

The application component of SketchWorld permitted the user to compose one or more drawings, each consisting of a background colour and a series of elements arranged within. The user was allowed to add, delete, move, or change element properties, as well as reorder the drawing order of elements, which determined which elements appeared on top or below others. The following element types were provided (Figure 3.5):

- *Line element,* a straight line between two points with editable thickness and colour.

- *Scribble element,* an arbitrary number of small line segments with editable thickness and colour.

- *Rectangle/oval element,* defined by a rectangular boundary, with editable border thickness and colour and optional fill colour.

- *Bitmap image element,* a JPEG or GIF image loaded from a file.

- *Text element,* a short text segment with editable border thickness and colour and optional fill colour.

With this selection of elements, users were able to create a wide variety of drawings. Images would result in large drawings that created significant system demands. Drawing size and complexity increased with the number of elements. Text elements allowed studying the effectiveness of viewing scaled-down text. It was anticipated that while shapes and lines and images would remain discernable even when scaled down significantly, text would degrade quickly.

Interaction with Community Component

The application component has four responsibilities to the community component: to provide complete application states on request, at specified qualities; to send state update messages when changes occur in the drawing, according to the qualities specified by observers; to resolve changes made to application states; to render the contents of application states in the community view.

The application state transmitted to other users in SketchWorld consists of the following:

- *The drawing,* consisting of its dimensions, the elements contained in it, and the background colour, at the quality requested by the community component. At the minimum quality, no drawing information is conveyed, as at this level the station appears as a coloured dot.

- *The currently selected tool.* This value is common for all stations associated with the user and is broadcast to all users in the MOA regardless of their interest in receiving it.

- *The cursor position,* if the user's cursor is currently over the drawing. For a low quality station, cursor position is not shown and is also not sent.

The application component will send complete application states only when requested by the community component. This will happen when a remote user changes the quality at which he/she wishes to observe the drawing, and the response from the application component is the application state at that quality. When the local user changes the drawing, a change notification message is sent unsolicited to the community component; the change notification consists of only enough information to describe the change, which will be one of the following:

- *Element added.* The user has added a new element, such as a line or an image, to the drawing. The change notification consists of only that element.
- *Element deleted/moved.* The user has deleted or moved an element. The change notification consists of a reference, or identification number, to the element.
- *Element modified.* The user has changed the element in some way, such as changing its colour or the width of the border line. A new version of the element is sent.

The notification message sent to the community component consists of several versions of the message to be sent to observers, one for each quality that has been requested. For instance, if two observers are viewing a station at quality q_1 and one at quality q_2, the application component will send a notification to the community component consisting of two messages, one at q_1 and one at q_2.

When an update message is received, the community component on the receiver provides it and the previous application state for its station to the application component; the application component resolves the update with the previous state to produce an updated state which is returned to the community component.

When the community component decides to redraw all or part of a station, it passes the associated application state to the application component along with graphics context information, and the application renders the application state.

Variance of Application State According to Quality

The means by which an application state is translated to a quality-reduced version is application-dependent. In general, some aspects of an application state will be readily reducible, while others will be nearly impossible to reduce from either a theoretical or practical standpoint. SketchWorld takes some steps to produce quality-reduced versions of drawing states that also result in storage savings, but it was not intended to represent the best possible efforts available. The study of reducing image size is both well-explored and ongoing [46, 41, 63], and there is no doubt that both SketchWorld's drawing state and the means of obtaining reduced-quality versions of it could be greatly improved with established methods. The effectiveness of filtering was judged by the qualities at which users saw the work of others and not the quantity of data that was actually transferred to avoid confusing the effects of filtering and the efficiency of the representation of drawings.

The following summarizes the differences between full-quality and reduced-quality drawing state elements in SketchWorld:

- *Bitmap image elements:* reduced-quality versions are scaled in dimension to match the rest of the drawing, and will generally require less storage volume than their full-quality counterparts.

- *Telepointers:* telepointers are scaled in dimension to match the rest of the drawing, and are omitted at quality levels below 30%, as it was felt that below that threshold they would not yield much meaningful information for users. At quality levels above 30%, telepointer messages are identical to their full-quality counterparts.

- *Line, rectangle, oval, scribble, text elements:* reduced-quality versions of these elements are scaled in dimension to match the rest of the drawing, but are identical in storage volume to their full-quality counterparts.

Two benefits of not pursuing best practices in drawing reduction were that it was possible to demonstrate behaviours that could arise when application state data cannot be realistically be compressed and to show how cutting off some types of this data completely under certain circumstances can be effective, as was the case with telepointer data.

Chapter 4
Filtering

The previous chapter presented filtering as a means of preventing user and system overload. This chapter delves further into how filtering works within the context of a MOA. The first section is a general discussion of filtering in a MOA, outlining what a filtering system should accomplish and what characteristics it should have. A filtering system will affect the qualities of stations in a user's community view, and the questions of how and when to alter their qualities will be examined. With the basic description of filtering established, the areas of a MOA that would take part in filtering are presented. The second section of this chapter discusses the filtering system of SketchWorld, which consists of a manager component that oversees several independent filter components. This discussion presents aspects specific to SketchWorld's implementation of filtering, highlighting the design decisions that were made and the implementation details of the filtering system.

4.1 Filtering in a MOA

Within a MOA, information about users' activities will be transmitted to other users. Without filtering, this information will result in each user seeing full-quality renditions of every other user's work and activity—every minute detail of a remote user's work will be sent to observers. The problem with this scenario is that at some point, either the system will be overloaded trying to generate, send, receive, and process this information, or the resulting presentation of community at the observer's side will be excessive and unusable. The goal of filtering is to reduce the amount of information exchanged between users to reduce the possibility of system or user overload, with the caveat that users are still provided with enough information to make the MOA useful to them. To accomplish this goal, filtering systems should strive for the following characteristics:

- *Computational efficiency.* The goal of a filtering system is to reduce the amount of data traffic exchanged by MOA users, and in doing so reduce the system effort required to generate, transfer, receive and process this data. The filtering system should endeavour to not create significant system demands to meet this goal. Control messages between MOA users and memory storage requirements for the system should be minimized.

- *Effectiveness.* A filtering system should produce changes that significantly reduce system load and result in changes to the community view that are noticeable by the user. Filtering is an effort to simplify the representation of community to the user as well as reducing system burdens.

- *Adjustability.* The user should be able to experiment with various filtering parameters and see immediate results in the MOA, recognizing that the filter system should be adaptable to user preferences and community conditions.

- *Predictability.* A filtering system should act in a way that is readily interpretable by the user. The user should be burdened as little as possible by the task of understanding changes in the MOA due to filtering.

4.1.1 How Filtering Could Affect MOA Data

A filtering system will rate stations according to criteria such as user interest or community opinion, and this rating will determine the qualities at which users will see these stations. The quality at which a station is observed may be a combination of several characteristics, of which the following are but a few:

- *Size.* Highly rated stations would be larger than lower rated stations.

- *Detail.* Highly rated stations would have most or all of their details visible, while lower rated stations would omit certain details.

- *Timeliness.* Highly rated stations would be updated more frequently, while lower rated stations would be updated less frequently.

For example, station quality could be defined as a combination of size and timeliness, with a high quality consisting of a large size and/or timely updates, and a low quality consisting of a small size and/or delayed updates.

4.1.2 Where Filtering Could Take Place

Figure 4.1 depicts a drawing MOA with membership consisting of one artist who is composing a drawing, and two observers who are watching the artist. The observers send requests to the artist to receive updates of the artists work at various qualities, and the artist responds by sending updates at the requested quality. Observer 1 wishes to view the work of Artist 1 at quality m, while Observer 2 wishes to view the work of Artist 1 at quality n.

There are three places where the filtering of application state messages could take place:

- *At the receiving point of the messages.* State messages would travel freely from source to receiver, and the receiver would pare down what is presented to the user. Only the receiver would need to know what qualities it wants to receive.

- *At some intermediate point on the network.* If data travelling between source and receiver goes through a server, for instance, the server could perform the filtering. The server would keep track of the qualities that all receivers expect from all senders.

- *At the source of the data.* The MOA client at the source would know the qualities that all receivers would want, and send the appropriate quality of data to each. Receivers would be required to inform senders the qualities of data that they want sent to them.

Filtering at the receiving end has the advantage of being the simplest strategy to implement as only the receiver will take part. No communication with the source of

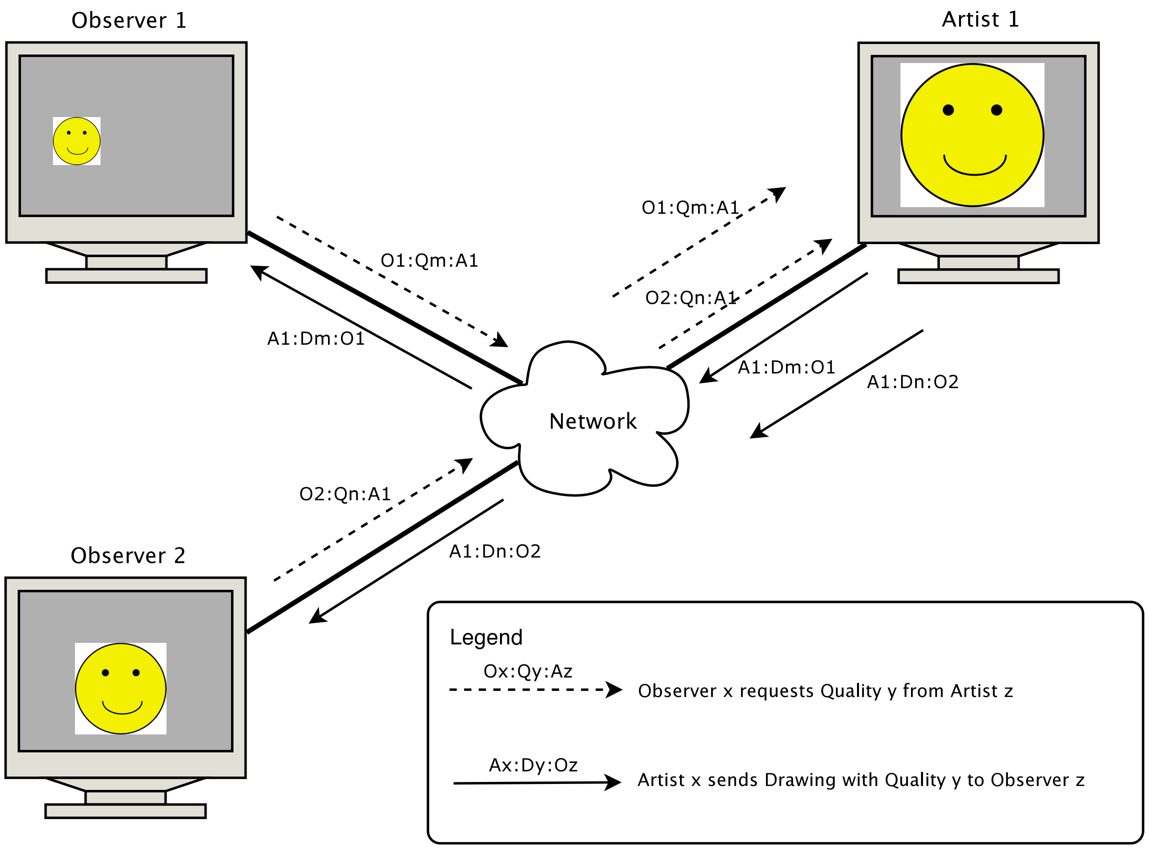


Figure 4.1: The exchange of quality requests and application state updates in a MOA.

the data is required. The drawback is this is the least effective filtering method as the sender and network are still burdened with the full amount of application state and user activity traffic. Filtering at the sending end requires that the receiver tell the sender what quality of information to send, and the sender has to keep record of this request and tailor transmissions to the receiver accordingly. The payback for this added complexity is a more effective filtering strategy since the receiver and network will be rewarded with reduced application state traffic.

4.1.3 When to Filter

There are a variety of strategies available to decide when to filter information and when to let it through unimpeded. For instance:

- *According to user interest.* This is the strategy pursued in this thesis. Filtering would result in interesting stations being shown with higher quality than uninteresting stations.

- *According to information volume.* Filtering would effectively set a maximum bound for information volume, allowing light flows (infrequent, small) through unimpeded, while throttling heavy flows (frequent, large).

- *According to system capabilities.* Filtering would access the capabilities of computers and networks along information paths, tuning information flow so that it doesn't exceed capabilities.

4.1.4 User- and System-Controlled Filtering

Filtering may be classified according to its relationship with the user. User-controlled filtering refers to a regime where the filtering system responds directly to actions by the user, resulting in an obvious cause-effect relationship: the user performs an action, the filtering system responds to the action, and hopefully the user will interpret the resulting changes made by the filtering system as the result of his/her own action. An example of user-controlled filtering is to filter according to how recently a station has been visited.

System-controlled filtering responds to actions taken by remote users and other stimuli that are not initiated by the local user, with the result that the filtering system will make changes to what the local user sees in the MOA without any apparent reason behind the changes. System-controlled filtering may involve responding to local user actions as well, but it must by definition incorporate additional remote stimuli. An example of system-controlled filtering is to filter according to community popularity. The risk with system-controlled filters is that the user will be distracted by the task of interpreting or guessing why stations in the MOA are changing in quality, so steps should be taken to provide additional feedback when changes take place. For instance, filtering based on community popularity could be augmented by displaying emblems for stations judged as popular.

4.1.5 Configurability

The filtering system will significantly influence how a MOA user will see the community, and for this reason the user should be able to configure it to suit his/her preferences. Configurability gives the user the feeling that he/she is in control of filtering and consequently how the community is presented. The user may wish for filtering to play a prominent role, or may choose to turn filtering off completely. The user will also be able to adapt filtering to the size of community, as a larger community may require a stronger filtering regimen to promote a select portion of the community. The user may wish for filtering to emphasize relationships with other users by promoting friends and co-workers while tuning out others completely. The user may wish to adjust filtering to suit the nature of his/her work as there may be times when it is important to focus on the activities of a select few, and other times when it is more important to keep a watchful eye over many. To summarize, the user will be able to use filtering to tailor his/her view of community to suit a wide variety of preferences and circumstances.

4.2 Filtering in SketchWorld

The conceptual model just presented is the foundation for SketchWorld's filtering implementation. This section discusses decisions made and details resolved to transform the abstract model to a working prototype.

4.2.1 Design Decisions

Several design decisions were made to address specific requirements or in response to characteristics inherent in the SketchWorld prototype. These decisions and the reasoning behind them are given as follows:

- *By default, a station is considered to be uninteresting.* Uninteresting stations will be shown at a minimum quality, and will increase in quality only through intervention by the filtering subsystem. This decision comes in response to the assumption that users will be interested in only a few stations at a time. The outcome of this decision is that the filtering system will need to keep track of only a fraction of the stations in the community.

- *Filtering is performed at the source.* Station owners keep track of the qualities that other users are observing the station and send them only the information relevant to those qualities. At the cost of informing data sources what they should send, application state traffic is filtered optimally, reducing the burdens placed on the network and receivers.

- *Filtering acts on direct inputs from users.* Perhaps the best way to explain this principle is with an example. Consider the design of a popularity filter that would boost the quality of stations that many users find interesting. One approach would be to calculate popularity by counting the number of times users have manually increased a station's quality by pressing the "+" key. Another approach would be to calculate popularity based on the average quality that users are observing the station. The first approach is seen to be better since the second approach suffers by being influenced by many factors such as filter settings and judgements made by filters that contribute to the resulting quality at which a station is observed, while the first approach relies solely on the explicit intentions of users.

- *User adjustments to the filtering subsystem will take immediate effect.* When a user changes some adjustable parameter, the result will be that stations will immediately change in quality. Users will be able to experiment with filtering to determine a desirable configuration.

- *Users are in charge of their own filtering configurations.* Each user will have his/her own unique view of the MOA community which will not be affected by the configurations of other users.

- *Filtering is noticeable.* Filtering will change station qualities significantly and not in ways too minute for users or systems to notice. When quality changes are caused by system-controlled filtering, notification messages alert the user to the changes and explain the reasons for them; these notification messages serve the purposes of bringing attention to changes that may otherwise be missed, and removing the need for the user to guess or interpret why they are occurring.

4.2.2 Filters and Filter Manager

SketchWorld's filtering subsystem consists of several independent filters that calculate preferred qualities for stations based on relevant inputs, and a manager that co-ordinates these filters. Figure 4.2 shows the relationship between the filter manager and the filters it supervises, and between the filter manager and the controller subcomponent. Inputs are fed from the controller through the filter manager to each individual filter. The filters respond with sets of desired view properties for stations they wish to change. The filter manager resolves these desired properties and sends requests for change to the controller.

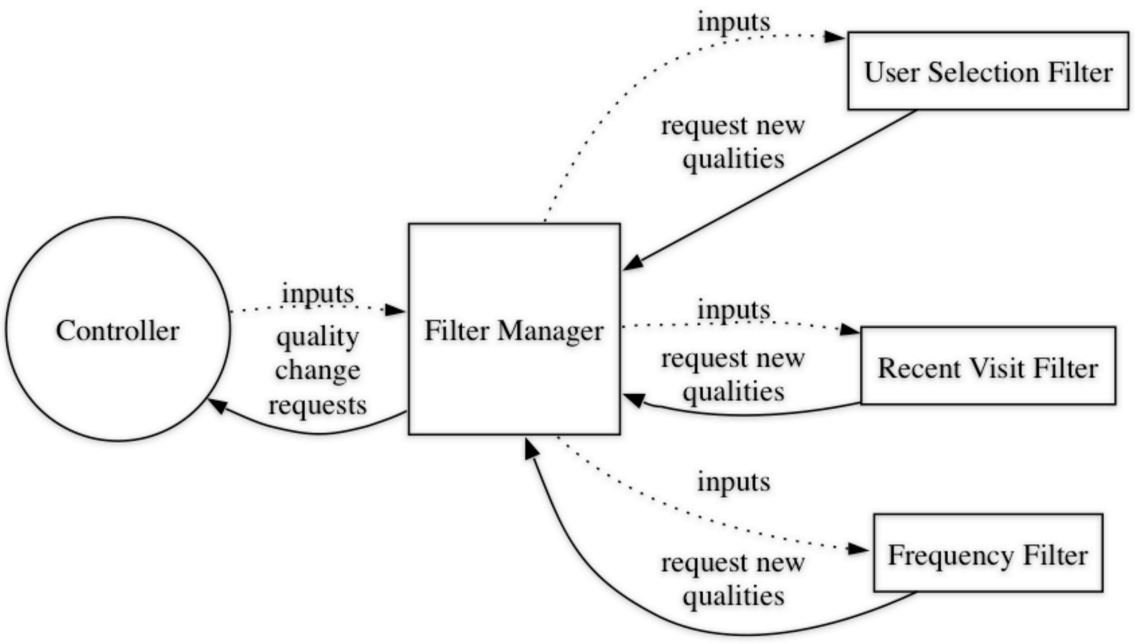


Figure 4.2: SketchWorld's filtering subcomponent. Three filters are shown and the rest hidden for clarity.

The following is a more detailed description of the inner workings of the filter subcomponent, given by listing the events that occur as stimuli to the filtering subsystem is converted to changes in station quality:

1. *The filter manager receives a stimulus,* such as a local or remote user's action or an interval timer.

2. *The manager passes the stimulus to interested filters.* The manager keeps track of the stimuli each filter wishes to receive, and passes only relevant stimuli to these filters.

3. *The filter processes the stimulus.* Based on the stimulus and other relevant information, the filter may decide to alter one or more stations' qualities. The filter will keep a record of the stations that it wishes to alter, and the qualities that these stations should have.

4. *The manager polls filters.* After the stimulus has been passed to all interested filters, the manager asks all filters to provide lists of stations they wish to alter as a result of the stimulus, and the desired qualities these stations should have.

5. *The manager calculates net station qualities.* The manager resolves all opinions from all filters, the result being a list of requested qualities for all stations to be changed.

6. *The manager sends requests for quality changes.* When the requested quality for a station differs from the quality at which it is currently being viewed, a quality change request message is sent to the station's owner.

4.2.3 Filter Implementation Details

In order to take part in the filtering model just described, a filter must satisfy the following requirements:

- *Independent.* A filter will not be influenced nor depend on other filters.

- *Agnostic of current station qualities.* A filter will not be influenced by past or current qualities of stations. The intent of this requirement is to avoid possible positive feedback loops, where an increase or decrease in station quality causes another immediate increase or decrease.

- *Memory of judgement.* A filter keeps track of stations it wishes to change in quality, and the qualities it wishes them to have.

4.2.4 Filter Manager Implementation Details

As mentioned previously, the filter manager performs an arbitration role by resolving differences of opinion between filters. The following discussion provides more details of the mechanics of the arbitration and explains how the filter manager assumes responsibility for producing noticeable changes to station qualities.

Strength and Weight Each filter has two parameters to govern its effectiveness. The first is *strength* which is how much of an effect a filter can have on a station's qualities. A strength of 100% means the filter can boost a station up to full quality, while a strength of 50% means the filter can boost a station up to a maximum of half quality. The second parameter is *weight* which is the filter's relative importance compared to other filters. If one filter has a weight of 80% and another has a weight of 40%, the first filter will be seen as twice as important or influential as the second. If a filter has a weight of 0%, it is effectively disabled.

Resolving Conflicting Opinions Between Filters Consider the situation when two filters, f_1 and f_2, would like to give a station the viewing qualities of q_1 and q_2 respectively. The filter manager will calculate the weighted average q_w based on q_1 and q_2 and the weights of the filters w_1 and w_2.

Noticeable Changes The filter manager performs a post-processing step on p_w using the current viewing properties for the station p_e and a predefined list of property levels. If the two values are equal, no further action is taken as the station is already being viewed with the desired properties. If there is a difference, the filter manager quantizes p_w into one of the predefined levels: if $p_w > p_e$, then p_w is increased to the next level above p_e; if $p_w < p_e$, then p_w is decreased to the next level below p_e. This quantizing step ensures that when a station's viewing properties change, they change significantly. The filter manager then sends out a request to change the station's viewing properties to p_w.

4.2.5 Filters Implemented

The following is a description of all filters implemented in SketchWorld, identifying their purpose, the inputs they respond to, the rules they follow when translating these inputs into requests for station viewing property changes, and any exceptions to their behaviour.

For these descriptions, a manual request to change station quality shall refer to the acts of pressing the "+" key to increase station quality, pressing the "-"' key to decrease it, or pressing the "=" key to remove any user quality preference for the station. The term "full strength quality" refers to the maximum quality a filter can request to increase a station's quality as governed by the filter's strength value. A user's "level of interest" in a station is the number of times a user has manually increased station quality minus the number of times he/she has decreased it. Also in the descriptions, visiting a station means mouse-clicking on a station to select it. Notifications of visitation are passed to the filtering system, which may choose to take some action; there is no other implied action associated with visiting a station.

User Selection Filter

The User Selection Filter allows the user to explicitly increase or decrease the quality of any station. This filter has no strength or weight associated with it and is special because it overrides the opinions of all other filters.

Inputs This filter responds to manual requests by the local user to change either a station's quality.

Rules For each manual request to increase station quality, the User Selection Filter will increase the selected station's quality by 25% of full quality. In other words, doing so four times will bring the station up to full quality. In a similar manner, a manual request to decrease station quality will do so by 25% of full quality.

The User Selection Filter incorporates a global bias adjustment that controls the default quality for all stations. By default the global bias is 0%. Increasing global bias will result in all stations in the MOA being boosted to at least the corresponding amount. When the global bias is a negative percentage, its effect will be to subtract the bias from quality calculations for all stations.

Exceptions This filter is special because it is not factored into the final calculation of station quality the same way as other filters. The filter manager will calculate the desired quality for a station based on all other filters first, then add or subtract the amount prescribed by the User Selection Filter. In effect, this makes the User Selection Filter override all other filters. The user can "mod up" a station so high that it will always appear at full quality, or so low that it will never appear more than at the lowest level.

Recent Visit Filter

The Recent Visit Filter increases the quality of the last n stations the local user visits, where n is a predefined value. The value of n used was 3.

Inputs Visitation is indicated by the local user clicking on the station.

Rules The Recent Visit Filter records the last n stations visited by the user. The last station visited is boosted to full strength quality. The second last station will be attenuated by 100/n% of full strength quality, and so on. Figure 4.3 illustrates the effects of the Recent Visit Filter as a user visits a series of stations, proceeding from left to right, with the rightmost station being the most recently visited.

Frequency Filter

The Frequency Filter increases station quality according to the number of times the local user visits a station.

Inputs Visitation is indicated by the local user clicking on the station.

Rules The Frequency Filter records the number of visits the user makes to all

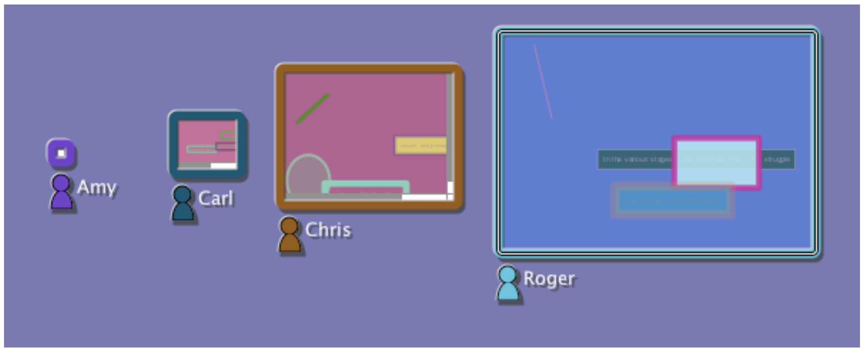


Figure 4.3: The Recent Visit Filter in action.

stations. For each visit, the filter adds (100/n)% of full strength to the station's viewing properties. The value of n used was 10. It is not possible to reverse the effects of this filter by "un-visiting" a station.

Current Tool Filter

The Current Tool Filter boosts the stations of remote users who are using the same drawing tool as the local user. This is an application-specific filter as it is based on information that is unique to the chosen application.

Inputs The Current Tool Filter keeps track of the drawing tools being used by all users. When a user changes tools, an indication of this change is sent to all other users.

Rules When a remote user changes his/her drawing tool to match the local user's, or the local user changes his/her tool to match a remote user's, all stations belonging to the remote user will be shown to the local user at greater quality. When the tools of the remote and local user no longer match, these stations will be shown at reduced quality. The boost a station will receive is 100% of full strength.

Exceptions Transitions to the "move" tool are ignored. For instance, if the local user switches from the "rectangle" tool to the "move" tool, the filter will continue to consider him/her to be using the "rectangle" tool. The rationale for this decision was the perception that the "move" tool was of secondary importance, to be used for editing existing drawing elements, and that switching to use it would not be of interest to other users.

Mutual Interest Filter

When two users express mutual interest in each others' work by manually increasing the qualities of each others' stations, the Mutual Interest Filter will provide an additional quality increase for these stations.

Inputs The Mutual Interest Filter records the levels of interest that remote users have in the local user's stations and the levels of interest that the local user has in remote users' stations.

Rules When the average level of interest the local user has in a remote user's stations and the average level of interest that the same remote user has in the local user's stations both exceed a threshold value, the filter will attempt to boost the local user's views of the remote user's stations to full strength quality; on the remote user's side, the filter will boost the local user's stations to full strength quality.

Activity Level Filter

The Activity Level Filter monitors the rate at which a local user makes changes to his/her work areas and notifies other users of activity level changes for these work spaces.

Inputs The filter receives notifications when the local user makes changes to his/her work spaces. It also receives notifications from remote versions of the filter when remote stations change activity levels.

Rules The filter tracks activity levels for each local work space. When the activity level for a work space crosses one of a predefined set of thresholds, the filter sends notifications to remote users. The filters on the receiving end of these notifications will in turn alter the qualities by which they view these stations in accordance with the new activity levels.

Community Rating Filter

The Community Rating Filter keeps track of the popularity of local stations, notifying remote users as they increase or decrease in levels of popularity. It also receives these notifications from remote users and in response alters the qualities the user will see of these stations.

Inputs The Community Rating Filter receives as inputs the manual requests to change the qualities of local stations from remote users. The filter also receives notifications from remote versions of itself when community ratings of remote stations change.

Rules The filter maintains a rating list for its local stations. The rating of a station is a function of the number of users who have an interest level greater than zero in the station, and the average interest level of those users. As a station's rating changes, it is compared to a predefined list of rating thresholds. When the new rating crosses a threshold, the filter broadcasts a notification message to all remote users. The filters on the receiving end of this broadcast will in turn adjust the quality at which they wish to view the station according to its new rating.

Social Network Filter

The Social Network Filter increases the quality of stations that a local user's favourites find interesting.

Inputs The Social Network Filter receives manual requests from the local user to change remote station qualities, and from remote users to change local station qualities. The filter also exchanges lists of favourite stations between instances of itself running on the clients of other users.

Rules Each user keeps a list of favourite stations, where a favourite station is defined as one with an interest level greater than a set threshold. Each user also keeps track of the interest levels of other users in his/her own work. When a remote user shows sufficient interest, the local user will send his/her current list of favourite stations. As the list of favourite stations changes, friends will be notified of the changes.

Random Selection Filter

The Random Selection Filter periodically picks a remote user's station at random and boosts it up to 100% full strength. The intent of the filter is to temporarily highlight stations that the local user would otherwise ignore.

Inputs The Random Selection Filter receives no external inputs. It operates on an internal timer, and at regular intervals goes through the process of selecting a station at random.

Rules This is a derivative of the Recent Visit Filter, so the second last station boosted will be attenuated, and so on.

Exceptions Stations being viewed with greater than the default quality are ignored.

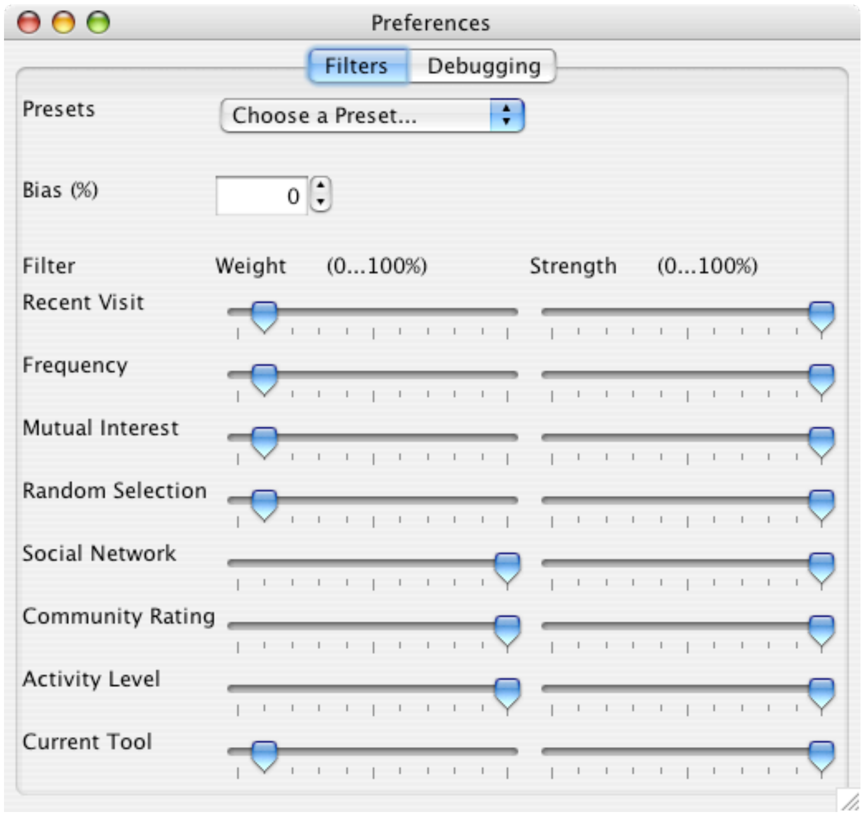
4.2.6 Configurability

Figure 4.4 shows the control panel available for changing filtering settings. Each filter has two slider controls, one for its strength value and one for its weight value. The Bias control governs the global bias value which is added to the calculation of station view properties. Valid values for bias range from -200% to 200%, effectively giving the user the ability to override any actions taken by filters. Using the global bias setting, the user is able to force all stations to be shown at full quality, or at minimum quality. Finally, the Presets control enables offers the user several preset settings:

- *All Off.* All filters are turned off.
- *One filter only.* Choose to turn one filter on, and all others off.
- *Social Butterfly.* Turn on the Social Network, Community Rating, and Activity Level filters, and turn the other filters off.

- *I'm the Boss.* Turn on the Recent Visit, Frequency, Mutual Interest, and Random Selection filters, and turn the others off.

- *Everyone is Great.* Turn all filters off and increase Bias to 100% to force all stations be shown with maximum viewing properties.



Presets Choose a Preset...

Bias (%) 0

Filter Weight (0...100%) Strength (0...100%)

Recent Visit

Frequency

Mutual Interest

Random Selection

Social Network

Community Rating

Activity Level

Current Tool

Figure 4.4: SketchWorld's filter control panel.

Chapter 5
Evaluation

This chapter presents the evaluation of SketchWorld, which consisted of four phases: a preliminary evaluation to search for obvious bugs, gather first impressions from users, and to give direction to later evaluations; a qualitative study to test the hypothesis that filters could be added to SketchWorld without detracting from its functionality; an examination of trace messages taken during the qualitative study; an experiment to measure the effectiveness of filters at reducing information flow between users. For each phase, the goals of the evaluation, the methodology of the evaluation, and the findings and the interpretations of the findings are presented.

5.1 Phase 1: Preliminary Evaluation

The goal of the preliminary evaluation was to identify obvious bugs, performance and interface issues, and obtain general impressions and suggestions from testers in preparation for later evaluations. It was anticipated that by finding and resolving these issues, testers taking part in subsequent evaluations would be less distracted by problems with the prototype and would be able to focus on the tasks given to them and to their outcomes.

5.1.1 Methodology

Three preliminary evaluations were carried out, each involving 4 to 6 testers. The first trial focused on merely attempting to run SketchWorld with several users. The second trial went one step further: testers were encouraged to actively observe others while doing their own work by assigning the task of creating a drawing and then copying someone else's drawing. The third trial shifted to the use and adjustment of filters: testers were instructed to experiment with and comment on different filter settings.

5.1.2 Findings and Interpretation

Aside from the usual defects and shortcomings expected of a prototype, testers generally seemed happy with using it. Sessions tended to devolve into caricature-drawing contests, with testers creating unflattering portraits of one another. The community view was useful for monitoring the progress of other users and probably fuelled the game of one-upmanship as each tester tried to outdo the others. The observations and suggestions fell into the categories of: context switching problems, community view problems, and filter problems.

Context Switching Problems

Using SketchWorld consisted of two separate activities, one being creating a drawing in one window, the other being monitoring and manipulating the community view in another window. Several testers commented that they had problems switching between the two activities during the course of the evaluation. One factor that may have contributed to this problem was that users tended to resize both their drawings and their MOA windows so they were large and overlapped one another, so it was impossible to see both windows at once, and switching between the two was cumbersome. One tester had a two monitor setup and opted to place the drawing window in one monitor and the MOA view in the other, and this somewhat alleviated the problem but there was still the issue of switching focus between the two monitors. Another aspect was that some users would have preferred to dispense with the drawing window altogether and work on their drawings in the MOA window. In fact, they often would forget to switch back to their drawing windows and would attempt to draw in the MOA window.

Interpretation Users face difficulties as a result of the decision to maintain drawing windows separate from the community view. One possible solution to this problem is to offer the user the ability to perform work in the community view or in separate windows and let the the user choose which method works best.

Community View Problems

While testers had little difficulty learning and understanding the community view, they expressed dissatisfaction with certain aspects of it. One problem they exposed was that it was difficult to keep track of several users at once, particularly if these users were separated by great distances in the view. They disliked navigating among interesting users whether they scrolled in the main or radar views or if they clicked on the interesting users in the station list. With these criticisms, however, came suggestions for improvement, including the use of thumbnail images of drawings in the station list and the radar view and ordering the station list by level of interest.

The decision to enforce common positioning also came under scrutiny, as several users indicated they would prefer to arrange stations in the MOA to their liking, while permitting others to do the same. If this capability were provided, users could position their favourite stations close together to reduce or eliminate the chore of navigating between them

Another point of contention was the positioning and movement of stations in the MOA view. During the first trial, users were able to move any station, even those belonging to other users. Some users did not like being moved by others, so for the second and third trials users were allowed to move only their stations, but this led to difficulties with resolving overlapping stations.

Interpretation There may not be one rendition of a community view that satisfies the needs and wants of all users and all working contexts. In a production version of SketchWorld, it will likely be necessary to allow the community view to be tailored to suit both individuals and the nature of their involvement with each other.

Filter Problems

Users had considerable difficulty understanding filtering and filter configuration. The question "why did this happen?" perhaps best illustrates the problem: users had trouble reconciling a filter configuration with the changes taking place in the community view as a result of that configuration. Similarly, they had trouble adjusting their configurations to suit their wishes.

Probable causes for these difficulties were:

- while users found that some filters such as the Recent Visit Filter exhibited obvious, predictable behaviour, others such as the Social Network Filter were less obvious.
- users could not readily explain the effects of combining the results of several filters.
- users found it difficult to distinguish between filter strengths and weights.
- adjustments to filter configurations did not always result in immediate changes to the community view. For instance, if a user decided to increase the strength and weight of the Community Rating filter but that filter had to reason to boost any station due to universally low ratings, the user would get the sense that the system was malfunctioning.

One suggestion made during this phase was to offer "preset" filter configurations that would offer complete sets of filter settings according to themes. As a result, presets were implemented in SketchWorld prior to subsequent stages in the evaluation.

Interpretation SketchWorld's filtering system is difficult to understand, at least for new users. Two factors that contribute to this difficulty are the number of filters offered and the lack of feedback explaining the effects of filtering as they take place. It is possible that users would develop sufficient understanding over a longer learning period; however, a comprehensive review and redesign of the filtering system may be in order.

5.2 Phase 2: Qualitative User Study

There were two goals for the qualitative user study. The primary goal was to test the hypothesis that *filtering could be introduced into a MOA to effectively reduce information exchanged by users, while still allowing the users to perform tasks they would normally do in a MOA*, as this was the focus of the thesis. The secondary goal was to gather observations of and opinions from participants of SketchWorld in anticipation of obtaining direction for future MOA designs.

5.2.1 Methodology

Setting

The experiment took place in a computer lab operated by the Department of Computer Science. Each participant was assigned a Dell Optiplex GX270 computer with 17 inch monitor, running Mandriva Linux. SketchWorld ran on version 1.5.0 of the Java Virtual Machine.

Participants

Participants for the experiment were recruited by an advertisement emailed to all graduate students in the Department of Computer Science. The stated requirements for participants in the advertisement were for people who were experienced computer users, and preferably those who had some experience using drawing programs. Receivers of this advertisement were encouraged to forward it to any friends or colleagues who may have been interested in taking part.

Eight participants took part in the experiment. From the profiling information forms they filled in (Appendix A.2.1), the following was known about them. The group consisted of three female and five male students from the departments of Computer Science, Nursing, and Psychology, with ages ranging from 22 to 50 years, with an mean age of 28.5 years and a median age of 25.5 years. All participants used computers at least 20 hours per week on average. While the stated expertise using drawing programs ranged from none to extensive, none of the participants appeared to have any difficulty using SketchWorld to draw pictures, as they all rapidly learned and utilized the tools made available to them.

Procedure

After participants entered the lab and were assigned their computer, they filled out consent and profiling forms (Appendix A.1, A.2.1). Once these forms were filled out and returned, participants as a group were given a tour of SketchWorld. They were told that in addition to being a drawing program, SketchWorld gave them the opportunity to observe the drawing activities of others. The facilities for changing the size of the representations of others in the community view were then described. Participants were told they could increase or decrease the size of these representations manually, and in addition could set up one or more filters to perform this resizing automatically. Each filter was described in terms of the actions it would take based on inputs it received. The means under which two or more filters could operate in parallel was also explained. Participants were then told that the experiment they were about to perform consisted of two parts, the first being a practice session, the second being a "working" session. Participants were encouraged to ask questions throughout the introduction and during the practice session.

Participants used the practice session to become accustomed to creating a drawing using SketchWorld, use the community view to observe the work of other participants, experiment with manually adjusting the view of others in the community view, and test the various filters that could do the adjustments for them. They were instructed to attempt to find a filter configuration that they believed would best suit them for the working session which followed. The practice session lasted for approximately 20 minutes.

Following the completion of the practice session, participants 1, 2 and 3 withdrew due to time constraints. They did, however, fill out task response forms (Appendix A.3.1) that were to be filled out at the end of the working session to record their experiences during the practice session. In the reporting of findings to follow, the responses given by these three have been included with those given by other five participants who did complete the second part of the experiment, as the activities performed during practice session approximated those of the working session, and responses given by the first three participants appear to be consistent with those of the remaining five.

During the working session, participants used SketchWorld to create a drawing while attempting to follow the progress of other participants. At some point they were to choose the work of another participant and attempt to replicate all or part of it in their own drawing. During the working session, event logs detailing the actions taken by participants as well as the data they received from other participants were recorded for later analysis. Participants were instructed to start with the filter configuration they had chosen during the practice session and use it for the entirety of the working session. At the end of the working session, participants filled out task response forms (Appendix A.3.1) to assess their performance and work load and provide impressions of the experiment. The working session lasted for approximately 20 minutes.

One anticipated problem was that an experiment involving relatively few participants would not be representative of the conditions of a MOA with dozens or hundreds of participants. In an attempt to simulate the conditions of a larger population, the overall world size was shrunk to 1200 by 1200 pixels and drawing size constrained to 600 by 600 pixels. With such a small world size relative to the number of participants, it would be impossible to view full quality representations of all other participants at once due to their representations overlapping. It was hoped that such a restriction would induce participants to be more selective of their viewing habits and view only a few representations at a time at higher quality and the rest at substantially reduced quality, either by manually growing and shrinking the representations or by using combinations of filters to do it for them.

5.2.2 Findings and Interpretation - MOA Functionality Questions

The first part of the task response form consisted of a series of questions focussed on whether MOA functionality was preserved when filters were actively reducing the data that participants received from others. Appendix A.3.2 contains their responses to these questions. The following is a summary of responses to the MOA functionality questions in the task response form.

Did the configuration behave as expected? (6 yes, 2 abstain) Judging from their responses, participants felt the filter configurations they used behaved as expected—stations increased and decreased in quality predictably according to filter settings and to manual quality change requests. In the words of one participant, their configuration "highlighted my interests and other things in common without much distraction". Given the short duration of the experiment and brief exposure to the filtering system, however, another participant remarked: "how would I know? (I need more practice with different settings)".

Did the configuration help you complete the task? (2 yes, 5 no, 1 abstain) Participants for the most part did not seem distracted by changes instigated by the filters. While participants didn't think the filters helped them perform their assigned task, they didn't appear to be bothered by them either: "I was able to work without any trouble". One participant implied the filter settings acted as a base, to be supplemented by his/her adjustments: "although my configuration tended to show the most popular users, I used the + and - keys to zoom in on users I couldn't see".

Were you able to maintain awareness of other users? (6 yes, 2 no) Participants apparently were able to maintain awareness of other users during the course of their task. One participant felt that filtering assisted maintaining awareness: "the filters made it possible to quickly analyse and view others". Another participant, however, noted "Sometimes the moving users makes it difficult to maintain awareness. I prefer to be able to move user's icons where I want them to be", so it appears that at least one person, awareness is related to maintaining a constant arrangement of that person's design in the community view.

Were you able to determine the identity of other users? (7 yes, 1 no) Determining the identity of others did not appear to be problematic. One participant explained how he/she did so changed during the course of the experiment, "first by using their name label, then by observing their drawing".

Were you able to maintain focus on interesting users? (5 yes, 3 no) Although the yes/no responses indicated that participants had some problems maintaining focus, the majority claimed they were able to do so. According to one participant, maintaining focus on another participant was facilitated by the fact that "their position and background colour were almost always the same".

Were you able to control the level of detail you saw of other users? (6 yes, 2 no) Participants apparently were satisfied with their ability to control the viewing qualities of stations based on their responses. The traffic graphs in Appendix B further strengthen this claim, as they show that some participants refrained from performing manual adjustments to station qualities (Figure 5.1), letting their filter setups do the work, while other participants would sometimes intervene by increasing a station's quality for a short time, then decreasing it to its previous quality (Figure 5.2). None of the participants appeared to disagree with their filter setups—no case, for instance, was observed where filters would change a station's quality, only to have the user immediately change the station back to its original quality. Participants appreciated their ability to resize others: "I liked being able to minimize the work that I found disgusting or irritating". Another participant, however, expressed disappointment with how the controls behaved: "the response was not quick enough, and it was difficult to control the level of detail in small amounts".

Did you feel overloaded by the information about others shown to you? (1 yes, 7 no) The lone participant who felt overloaded during the experiment cited "too much information, but short learning curve on the software could account for that". The fact that so few claimed to have been overloaded is somewhat surprising given that during the course of the study, SketchWorld was a busy, industrious

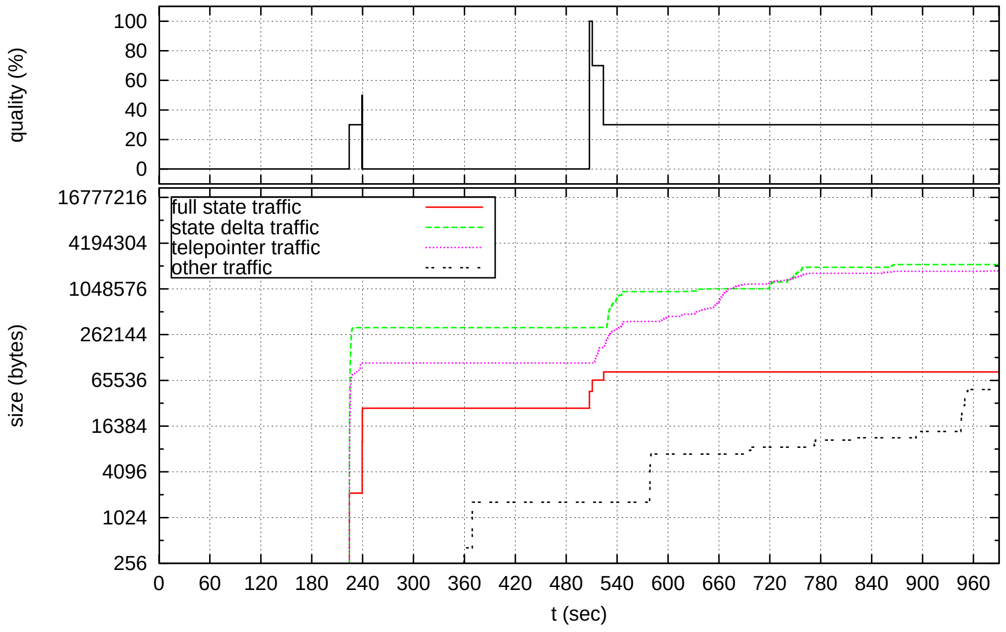


Figure 5.1: Traffic received by User 8 from User 5. User 8 refrains from manually adjusting observation quality.

environment and at times systems exhibited signs of strain.

Were you able to keep track of what others were doing? (6 yes, 2 no) Most participants were able to keep track of what other users were doing according to their Yes/No responses, but none cared to elaborate on their answer. The relatively positive response to this question is surprising since SketchWorld was deliberately hobbled in an attempt to approximate the effects of working with a much larger community, and that keeping track of others was expected to be problematic as a result.

Did you enjoy performing the task using the configuration? (5 yes, 3 no) Participants generally liked performing the experiment and using the MOA, although the strongest comment from all participants was “this could be fun”.

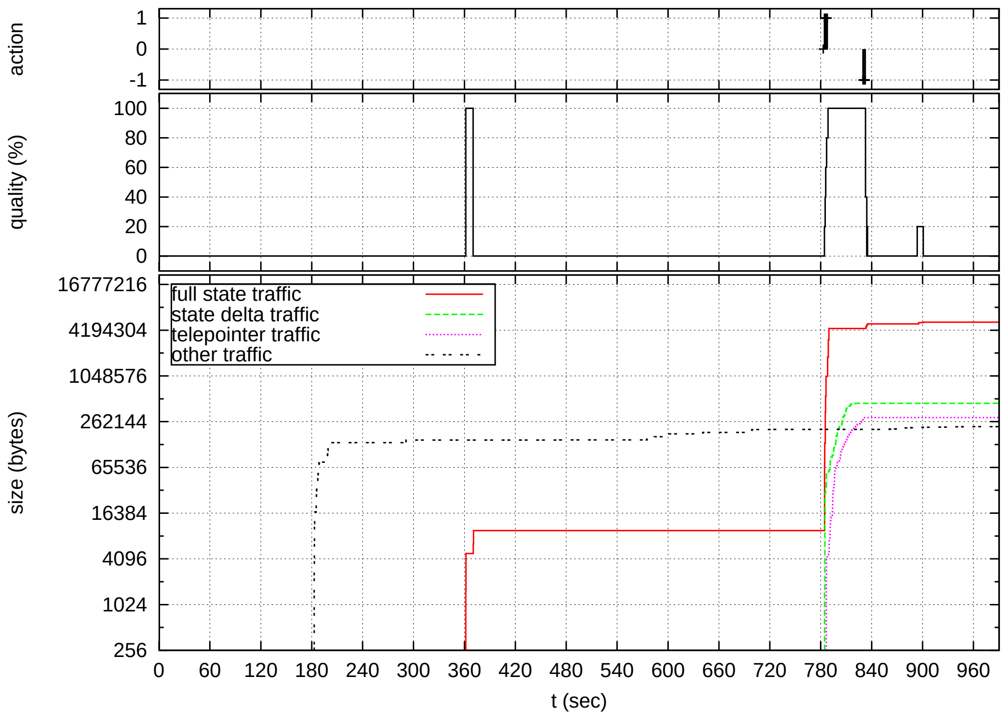


Figure 5.2: Traffic received by User 5 from User 4. User 5 increases the quality for User 4 briefly, then decreases it back to its original level.

5.2.3 Findings and Interpretation - Task Load Questions

The second part of the task response form consisted of questions to assess the physical and mental effort required to perform the experiment (Appendix A.3.3). The following is a summary of responses to the task load questions in the task response form.

Was there significant mental demand required to perform the task? (2 yes, 7 no) While the yes/no responses suggest that mental demand was not significant, participants' comments suggest the experiment was not completely effortless. According to one of participants who felt mentally taxed, this might be attributed to the nature of the experiment: "Yes, given the play and watch nature of the work in the experiment I did not really have a goal or motivation for being logged on.

The consequence being that I did not have any plan for organizing or prioritizing information from others. Also there was the component of guessing what others were doing and knowing if it was really of interest to me or not. I guess this would also be a consequence of not knowing how to use the filters (another learning curve situation)" Of the participants who didn't feel mental demands were significant, one of them still remarked "It was straightforward, except for the filtering."

Was there significant physical demand required to perform the task? (8 no) Participants did not feel taxed physically, although one participant noted "I did need to switch between mouse and keyboard a little bit".

Was there significant temporal demand required to perform the task? (8 no) Participants did not feel they were rushed while performing the experiment, with one who felt that "once I had mastered the software everything would be fine." This result should be bolstered by the fact that participants were asked to learn to use both the drawing and community aspects of SketchWorld and then perform a task that kept them busy shifting focus from drawing to observing within relatively short time period.

Was there significant effort required to perform the task? (7 no, 1 abstain) Participants did not feel they expended significant effort, although one indicated that relationships between participants would affect this result: "I am guessing the relative stability of the social group using the software would impact this aspect. I am guessing that communication patterns of the people using the software would make a big difference.".

Did you feel your performance was good? (6 yes, 2 no) Participants were generally satisfied with their own performance, with one participant remarking "generally I was able to do what I needed to". Filtering, on the other hand, appeared to make another participant less satisfied: "filter settings were causing unexpected behaviour at some times, making some users' windows too big, which cluttered the space and made it difficult to see others' work".

Did you feel frustrated while performing the task? (1 yes, 7 no) The lone user who expressed frustration while performing the experiment remarked "occasionally, the system was too slow and my drawings suffered". During periods of temporary system overload, participants reacted by pausing their activities to let the system catch up, so the apparent lack of frustration among participants may be at least partially due to their ability to readily adapt to the limits of the system.

5.2.4 Findings and Interpretation - Comments, Suggestions

Participants were given the opportunity to provide additional comments regarding any aspect of the experiment and of SketchWorld in general. Concern was expressed that the learning curve to perform the experiment was too steep, as participants had to learn the drawing program, and the MOA, and most of all the filters. Some felt that they might have understood and liked the filters more if they were to perform the experiment over a period of several days.

Interpretation a long-term experiment (several days, weeks) would likely eliminate learning curve problems, and users would be more comfortable using the MOA.

Numerous comments and suggestions were made regarding the community view. One participant expressed displeasure with the 2D presentation due to the fact that positions in the MOA were entirely by user preference and thus didn't help much. Another participant put forward the suggestion to sort the Station list according to current quality, so the largest Station would appear at the top of the list. This was in response to the problem of keeping track of who was popular in the community by using the community-based filters. Another suggestion was that the "drawing window should be set semi-transparent so I can draw and monitor the others' updates as well".

Interpretation SketchWorld's community view design was adequate for experiment, but many improvements are possible. Improvements would not be difficult to make using the current architecture - changing presentation, utilizing information that is already available.

One participant observed that while filters effectively pushed less interesting people away, there should also be mechanisms to pull more interesting people closer: "it is easier to get rid of someone than to find them. In other words, censorship is easier than discovery/search".

Interpretation The design principle behind the community view and filtering was that by default, everyone is uninteresting, and it was up to filters and user intervention to bring emphasis to interesting users, but perhaps the implementation was not effective enough.

5.2.5 Observations

During the course of the working session, participants and the system they were using were watched in order to assess the overall mood of the group and to be vigilant for any notable happenings. The following observations were recorded.

Some Participants Changed Their Filter Settings

Although participants were instructed several times not to alter their filtering configuration while the experiment was in progress, a few disregarded instructions and altered it anyway.

Interpretation Participants forgot or misunderstood their instructions, or they felt compelled to do so due to curiosity or dissatisfaction with the configuration they started with.

Participants Modified Their Behaviour to Avoid Overload

There were times during the experiment when system overload occurred, as participants noted slowdowns and intermittent freeze-ups. When these overload periods occurred, participants agreed among themselves to stop what they were doing until the system was able to "catch up". Over time, participants apparently learned to avoid behaviours that contributed to overload conditions.

Interpretation Conditions for overload were present, but overload was avoided. Part of this may be a result of participants learning what the limits of the system were, and staying below them. SketchWorld's filters aren't a guarantee that overload won't occur; however, users themselves can act as filters and regulate their activity in accordance with what the system allows them to do. Future research into MOA filtering could involve filtering that monitors system load parameters and attempts to keep them within reasonable limits.

Participants Wanted to Share

Numerous participants asked how to copy a portion of another participant's drawing from within the community view and paste it into their own drawing, and expressed disappointment when they were informed that this was not possible.

Interpretation Participants were instructed to duplicate portions of other users' work in their own drawings, and seemed to assume that if they were able to see another participant's work in the community view, they should naturally be able to copy and paste it into their own drawing rather than tediously reproducing it element by element. This may suggest that since participants were told SketchWorld was a community-oriented application, they expected that they should be able to share with one another, a fundamental trait of communities.

5.2.6 Summary of Results

The experiment confirmed the hypothesis that it was possible to introduce filtering to reduce data exchanged among users while still allowing them to use the MOA effectively. The filters consistently kept stations at reduced quality, occasionally increasing their qualities for brief periods, while participants were still able to maintain awareness of others and feel in control of their community view.

Participants did not feel they expended an inordinate amount of effort to participate in the experiment, even though it was observed that they did attempt to perform their tasks in earnest. There is still room for improvement, however. In several instances, participants would temper their response to a "yes/no" question with a comment to the contrary. For instance, one participant answered "no" to the frustration question, but commented: "Not really, but as mentioned the filters were hard to understand and interpret." While some participants attributed their problems with filters to their unfamiliarity with them and speculated that over time these problems would be diminished, the effort to improve how filters work and are controlled by the user would be well spent.

While participants' responses do indicate that they were largely satisfied with SketchWorld and its implementation of filtering, there is also evidence to suggest that their level of satisfaction would increase if they were able to it use over a longer period, particularly with respect to filtering and filter configuration: several participants stated this either orally during the experiment, or in written form in their comments.

Users expressed a desire for flexibility in the way community was presented and suggested many feasible improvements. Another indication of this desire for flexibility was that some participants seemed compelled to tinker with filter settings, even when instructed not to do so. In addition to their desire for flexibility, participants expressed their desire to share others' work, as they seemed to assume that this ability was naturally a part of a community-enhanced application.

5.3 Phase 3: Trace Log Examination

During the qualitative study, trace logs recorded actions taken by users and the data exchanged between users. Information about data received from other users was extracted from these logs and studied.

5.3.1 Goals

The primary goal of this phase was to verify that filters were actively reducing station qualities. The secondary goal was to identify characteristics and patterns of quality levels and data reception to further assess the effectiveness of filtering and possibly identify issues that should be addressed in future versions of SketchWorld or other MOAs.

The examination of station quality was chosen over a comparison of full quality data versus reduced quality data because the effects of quality on data volume were highly dependent on nature of the drawing being transmitted from originator to receiver. As an example, consider two drawings, one consisting entirely of line segments and one consisting entirely of bitmap images. The drawing consisting of line segments would result in identical data streams whether it was observed at full quality or at reduced quality, since the data structure used to store a line segment occupies the same number of bytes regardless of the quality. The drawing consisting of bitmap images would result in data streams heavily dependent on the quality, since a reduced quality bitmap image will likely occupy substantially fewer bytes than its full quality version.

5.3.2 Methodology

The trace logs taken during the Qualitative Experiment (Phase 2) contained records of each message received by participants; these records consisted of the time the message was received, its size in bytes, and its type. The logs also contained records of actions taken by users and filtering systems, such as records of when station qualities were altered.

Using a series of scripts, information was extracted from the log files and transformed into input files for Gnuplot[1], a plotting program. The resulting plots are contained in Appendix B. Figure 5.3 is an example plot showing the traffic received by User 6 from User 7 during the course of the experiment. The horizontal axis shows the time t in seconds at which events occurred. The top graph indicates times when User 6 explicitly changed User 7's quality; its vertical axis indicates whether the change was to increase quality (action = 1), decrease quality (action = –1), or "zero out" any previous manual changes (action = 0). The middle graph tracks the quality at which User 6 was observing User 7, indicating changes caused both by manual changes by User 6 and by decisions made by his/her filtering system; its vertical axis indicates quality. The bottom graph shows the cumulative traffic received by User 6 from User 7, divided into four categories: full state messages (complete drawing), state delta messages (changes made to the drawing), telepointer messages,

[1]Gnuplot Home Page `http://www.gnuplot.info/`

and “other” traffic, consisting of filter and session management messages; its vertical axis indicates the cumulative traffic received using a logarithmic scale.

The following descriptions of the some of the events portrayed by the graphs may assist the reader:

- at approximately t=330 sec, User 6 manually increases the observation quality of User 7. User 6 begins observing User 7 at 20% quality, and receives a full state message from User 7.
- between t=480 sec and t=620 sec, User 6 receives numerous state delta messages from User 7, as User 7 was making changes to his/her drawing.
- at approximately t=620 sec, User 6 manually increases the observation quality 4 times, from 20% to 100% quality. User 6 then receives 4 full state messages from User 7. User 6 also begins to receive telepointer messages from User 7 after exceeding the quality threshold of 30%.
- at approximately t=650 sec, User 6 drops the observation quality back to 20%. It appears that the first manual request to drop the quality occurred at approximately t=630 sec, but there was a delay of approximately 20 seconds before the request was acted upon.
- during the period from t=250 sec onward, User 6 receives “other” traffic from User 7. The bulk of these messages are from User 7’s filtering system to User 6’s filtering system, as various filters such as the community rating filter send messages to their remote partners.

5.3.3 Findings and Interpretations

The plots derived from user log files were studied, with particular attention given to user actions to change station qualities, changes in quality levels, and to the relative significance of the different data types received by observers.

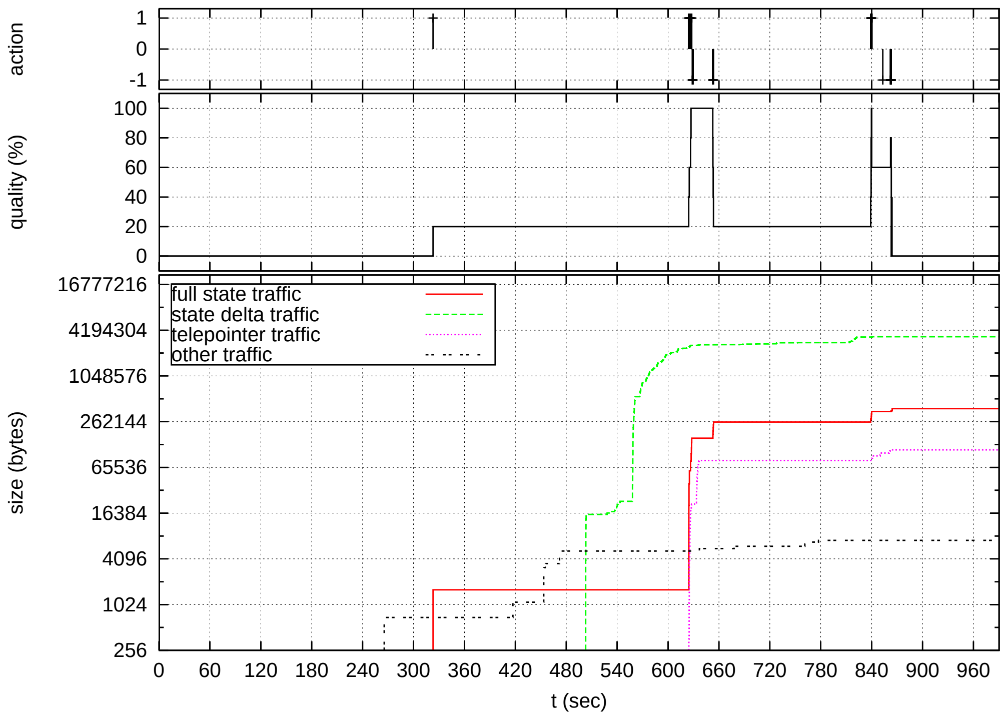


Figure 5.3: Traffic received by User 6 from User 7.

User-Invoked Quality Changes

The number of quality changes invoked by users varied from user to user. Users 5 and 6 performed frequent changes; at times, it appeared that their influence was far greater than the filters they were using. In contrast, User 8 performed no quality adjustments.

The timing of user-invoked changes was also of interest. At no time did any user contradict quality changes invoked by their filtering system. For instance, no user attempted to decrease station quality immediately after a system-invoked increase.

Interpretation Users seemed happy to let the filters do their work. When users did intervene, it was to take a closer look at another user, then move on, as witnessed by the pattern of increasing station quality, then decreasing it to its previous level after roughly one minute.

Quality Levels

Each participant had their own way of viewing the community, as evidenced by the quality level plots—no two participants viewed other participants in exactly the same manner. One pattern that did emerged, however, was that station qualities tended to remain consistently low, increasing only for brief periods due to either user intervention or due to the decisions made by the filtering system. A relationship between quality level changes and user to user data traffic that also emerged was that when a station's quality level changed, this would result in a large increase in traffic from that station due to the full state messages that were sent in response to quality level changes.

Interpretation Filters substantially reduced data traffic as indicated by the quality levels, as qualities were usually below 40 percent of full quality. This may not be indicated by the traffic levels due to the nature of the drawing elements used. Techniques to further reduce data exchange would be beneficial. For instance, when quality changes from high to low, it should be possible for the observer to derive the low-quality version of application state from the high-quality version, removing the need for the observed side to produce and send it. Techniques to limit the number of quality changes over a short period of time might help reduce the sudden bursts of data traffic such as the ones observed. In the current state of SketchWorld, it appears possible, without setting unreasonable conditions, to create a scenario under which using filters and user-invoked quality changes brings the system closer to overload than an unfiltered system would. To avoid this scenario, the aforementioned suggestions to improve SketchWorld, as well as any other future suggestions should be given consideration.

Data Type Significance

At times full state traffic exceeded delta traffic, while at other times deltas exceed full state traffic. There were instances where telepointer traffic rivalled application state traffic in magnitude. Filter management traffic was relatively low when compared to application state and telepointer traffic.

Factors that influence which data types are more prominent are found both on the originator and observer sides. On the originator side, the factors are the composition of the drawing, the rate at which elements are added, removed, or modified, and the level at which the drawing creator is moving his/her mouse over the drawing. The latter factor becomes important once the observer is viewing with sufficient quality to view telepointers. On the observer side, the factors are the quality level, which influences the size of full state and state delta messages and whether telepointer messages are sent and the frequency with which the quality level changes, which determines the number of full state messages sent.

Interpretation Conditions exist under which the dominant traffic could be full state, state delta, or telepointer; therefore, all three types should be treated as significant potential threats to create system overload. The current filtering system meets the goal of reducing application state data traffic without introducing significant data traffic to perform the filtering.

5.3.4 Summary of Results

The trace log examination revealed that during the course of the user study, filtering kept station quality levels consistently low, although there were periods when qualities did increase substantially due to user intervention or to changes invoked by the filtering system. Users largely remained in agreement with the filtering system, overriding it for brief periods to examine stations more closely. The examination also exposed a flaw in the filtering implementation which could potentially lead to circumstances where filtering actually contributes to the possibility of overload. Currently, when the observation quality for a station changes, the observer will receive a full application state message. Frequent changes to observation quality would lead to a deluge of full state messages. Fortunately, there appear to be improvements that would reduce this possibility, such as limiting the rate of quality changes. Finally, it is prudent to consider all types of data when filtering, as each could contribute significantly to overall traffic, as witnessed by this study.

5.4 Phase 4: Traffic Simulation

The fourth and final phase was a simulation study of the effectiveness of filters at reducing the exchange of application state data among users. During Phase 3, there were no controls on the drawings that users created or on the quality levels at which they were observed, while in this phase, the drawing and quality levels at which it was observed were strictly controlled.

5.4.1 Goal

It was demonstrated in Phase 3 that filters could be used to reduce the qualities at which users observed each other. Phase 4 intended to establish a relationship between observation quality and application state traffic reduction and show that reducing observation quality would result in reduced traffic.

5.4.2 Methodology

The experiment consisted of five instances of SketchWorld running on separate computers. The computers used were Dell Optiplex GX270, running Mandriva Linux, and Java 1.5.0. Clocks on the computers were approximately synchronized. One instance of SketchWorld was used by a human operator to create a drawing, while the other four were set up to observe the drawing at varying qualities, and were otherwise unmanned during the experiment. Trace logs were captured for the observing instances of SketchWorld in order to study the data they received.

In order to force the observing instances of SketchWorld to view the observed drawing sequence at a constant quality, their filters were disabled and they were manually configured to view the drawing with the qualities of 20%, 40%, 80%, and 100%.

It took approximately 5 minutes to create the drawing using the following sequence of actions:

- add a photograph to the drawing. The image used was 480 by 480 pixel jpeg image with a size of 184 kilobytes.

- resize and center the photograph.
- add a rectangle to form a border for the photograph.
- edit the border rectangle colours and stroke width.
- add a text element to serve as the caption for the photograph.
- change the text element's colours and font several times.
- move the text element above and roughly centred with respect to the photograph.

Figure 5.4 shows the drawing from the point of view of the observer watching it at 80% quality.

Following the experiment, trace logs were gathered and processed using the same techniques as in Phase 3. The resulting graphs of received data according to observed quality are Figures 5.5 to 5.8.

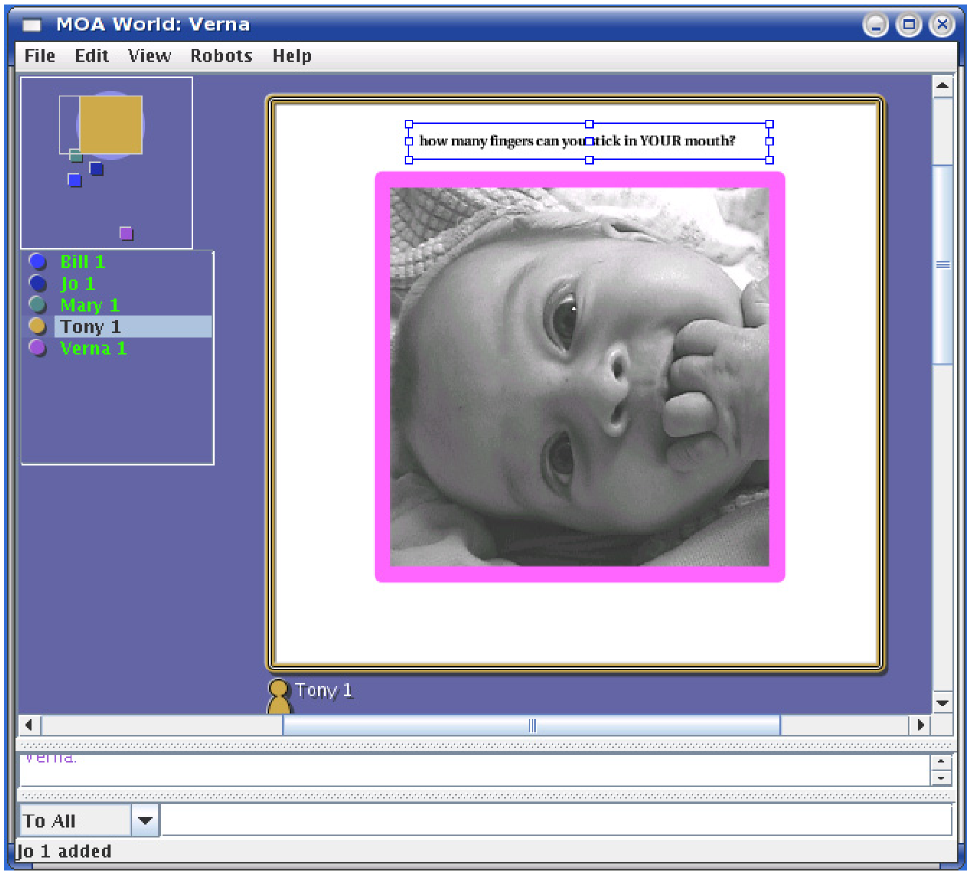

MOA World: Verna
File Edit View Robots Help
Bill 1
Jo 1
Mary 1
Tony 1
Verna 1
how many fingers can you stick in YOUR mouth?
Tony 1
To All
Jo 1 added

Figure 5.4: SketchWorld drawing being observed in the community view during traffic simulation.

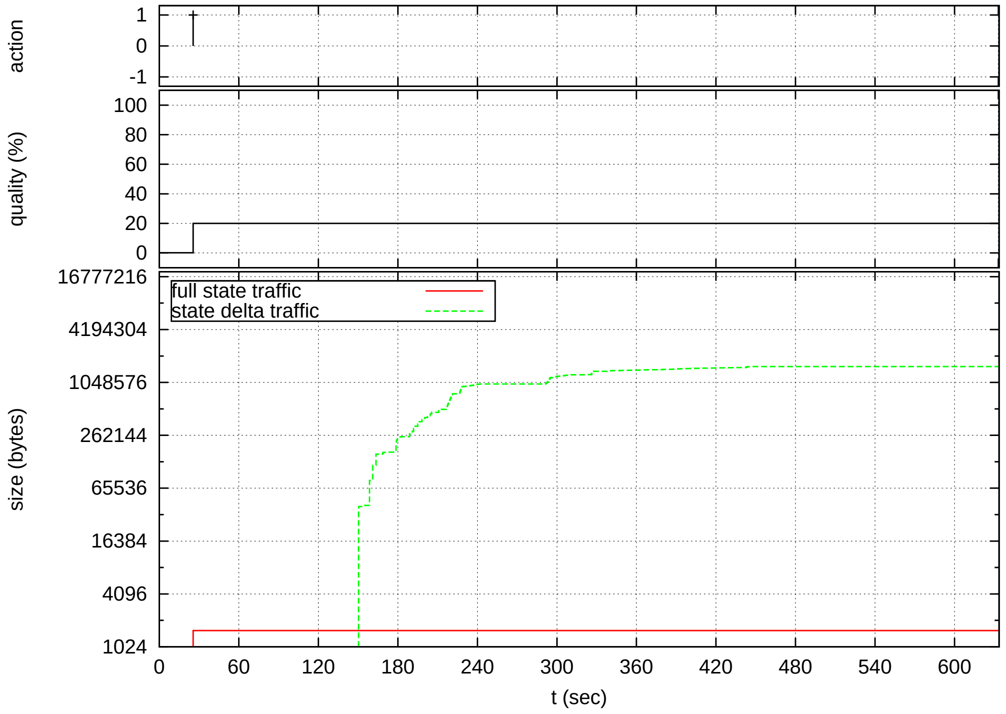


Figure 5.5: Traffic received by 20% quality observer during traffic simulation.

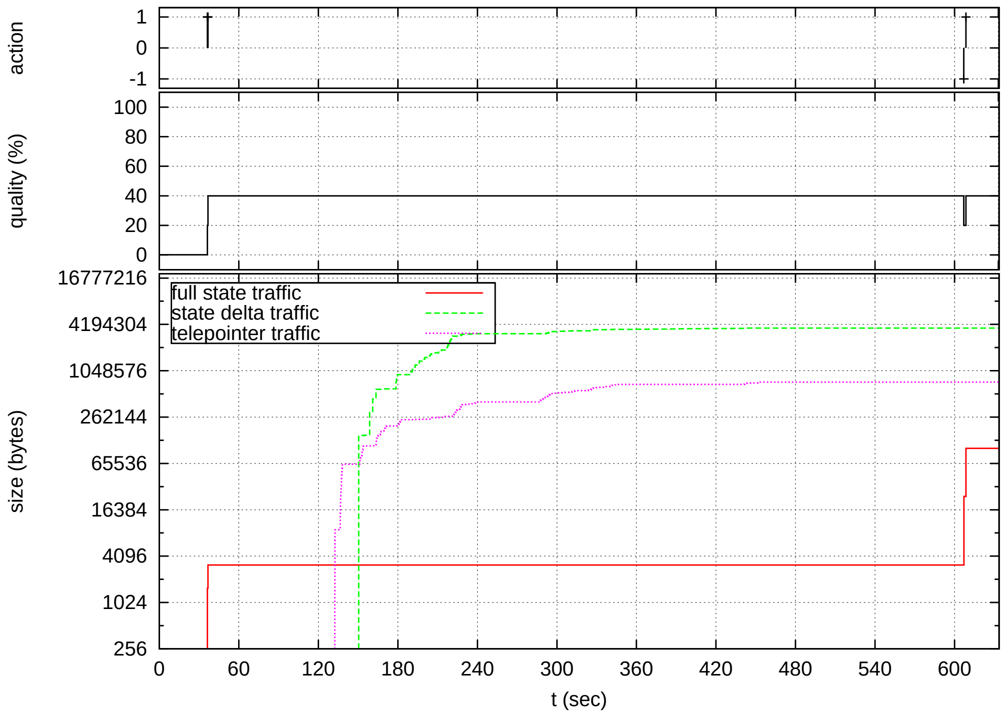


Figure 5.6: Traffic received by 40% quality observer during traffic simulation.

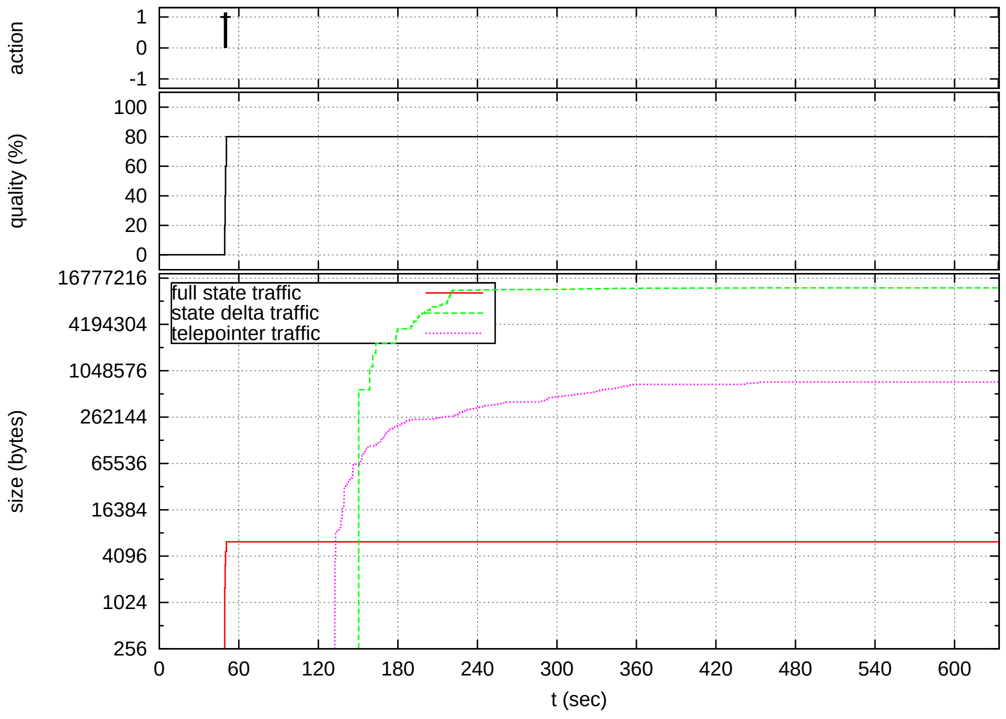


Figure 5.7: Traffic received by 80% quality observer during traffic simulation.

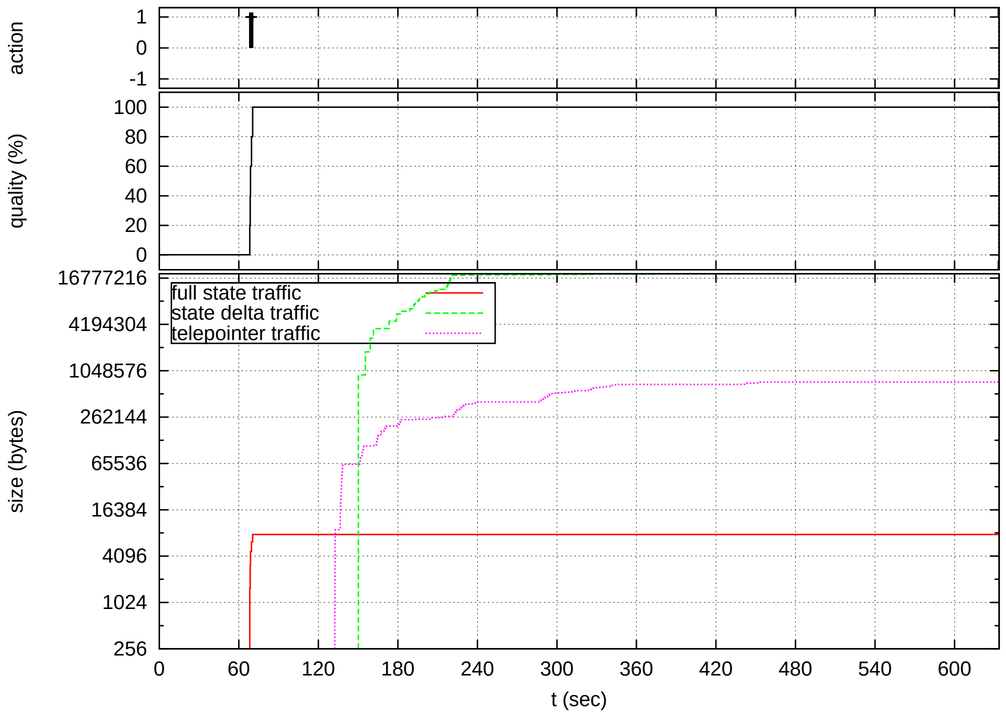


Figure 5.8: Traffic received by 100% quality observer during traffic simulation.

5.4.3 Findings and Interpretation

As expected, the amount of traffic an observer received increased as observation quality increased. During the course of the simulation, occasionally one or more observers would lag behind the others—it would take several seconds longer for their version of the image being observed to be updated to show the latest changes. Eventually the updates would arrive and the lagging observers would catch up with the rest. It should be noted that it wasn't always the high-quality observers who experienced lag, and that sometimes even the 20% observer trailed the others. Examination of trace logs revealed that the sender of the application state messages was the culprit, as messages to a lagging observer were sent long after those bound for non-lagging observers. The remainder of the findings for this simulation are discussed according to the types of data being received by observers.

Full State Traffic

Full application state messages, containing representations of entire drawings, were sent to observers as their quality levels were adjusted. During the simulation, quality levels were adjusted only during the initial setup, in 20% increments, with the 20% observer receiving one full state message, the 40% observer receiving it twice, and so on. The initial setting of quality levels took place before the drawing started, so the resulting messages conveyed an "empty" drawing consisting of merely a background colour.

State Delta Traffic

Differences in delta traffic, consisting of messages containing update information for drawings, were entirely due to the image being resized, as deltas to convey changes such as the addition of line segments would have been the same size regardless of quality.

Telepointer Traffic

Telepointer traffic was nearly identical for 40%, 80%, 100% observers. The 20% observer did not receive any telepointer traffic since it was cut off for qualities below 30%.

Interpretation

The traffic simulation confirmed the relationship between observation quality and the reduction of application state traffic, and also demonstrated how the makeup of the application state influences the reduction.

The presence of lag indicated that the sender of application state updates could become taxed even with a few observers, as some observers experienced long waits for information that other observers received promptly. Lag was also the likely cause of the slight differences in traffic patterns between observers.

The effectiveness of using a cutoff quality level technique as was employed to regulate telepointer traffic suggests that similar strategies may be effective for similar types of information. For instance, it may be helpful to use a cutoff technique for line segment updates: the sender of updates could be programmed to not send update messages to low-quality observers when small line segments are added, deleted, or modified.

5.5 Overall Evaluation Summary

The evaluation of SketchWorld demonstrated that filtering could be introduced in a MOA to reduce the exchange of application-related traffic between users while preserving the core functionality of a MOA, as evaluation participants were able to perform their assigned tasks with little difficulty. The filtering system used also fulfilled the design goal of reducing application-related traffic without introducing significant overload. The evaluation, however, also revealed there is much that could be done to improve SketchWorld, by allowing greater user customization of the community view, improving the filter configuration interface, and creating more sophisticated and efficient mechanisms to further reduce user-to-user traffic and further reduce the possibilities of system overload.

Chapter 6
Discussion

This chapter elaborates on the results of the evaluation of SketchWorld, first by stating that the results of the evaluation suggest that the hypothesis that filtering may be added to a MOA to help avoid overload conditions while still providing a MOA's benefits to its users, then by formulating the lessons learned for MOA design during this study, and finally by discussing possible avenues for future MOA research.

6.1 Findings

6.1.1 Filtering Can Reduce Burdens on Users and Systems

The evaluation of SketchWorld suggests that it is indeed possible to use application state message filtering to reduce the possibility of user and system overload, while still providing the benefits of a MOA to its users. During the course of the experiments performed, filtering substantially reduced the qualities at which remote users' stations were viewed by observing users, yet users felt they were able to perform the tasks assigned to them. They were able to maintain awareness of others and monitor their work, and were satisfied with their ability to control the qualities at which they viewed others.

In a larger community, filtering may be expected to play a more prominent role. Users may need more automated assistance for seeking out experts or for identifying close colleagues as the populations of non-experts and strangers increase.

Perhaps the central issue to the effectiveness of filtering in MOAs other than SketchWorld is the reduction of application state according to a set of prescribed qualities: this reduced state must be assessed in terms of how the user sees it, and the accompanying data reductions that are expected. The study of SketchWorld suggests that filtering would work to some degree for other graphical applications, but it is not known how well filtering would work in spreadsheet or word processing MOAs.

6.1.2 Users Accept MOAs

Users appeared to enjoy and accept MOAs. Even in the compressed time frames of the experiments, they were able to grasp the concept of marrying application and community and felt comfortable with using filters to identify favourite or important users in their community. Users themselves proved to be effective filters, as they were the best judges of their likes and dislikes and could act on their preferences. Users also demonstrated that they were also effective at dealing with and avoiding system overload by controlling their behaviour once they had found the boundaries of their system.

6.2 Lessons Learned

6.2.1 User Perspective

For MOAs to be successful, they must meet the needs of their users. They must be flexible, controllable, provide feedback to automated actions, and allow users to share their work.

MOAs Must Be Flexible

Users want the ability to mould a MOA into an environment that suits their needs and preferences. They want the ability to alter the community view, for instance, to arrange stations in the view according to their relevance. Users also have varying expectations of filters: some will want to configure them to automatically filter based on the work they are doing or their interest in others, some will prefer to disable all automated filters and rely solely on performing manual quality adjustments, and some will opt for a mixture of automated filtering and manual adjustments.

MOAs Must Be Controllable

Users largely accepted the concept of having filters that automatically respond to events and tailor station qualities accordingly, with the provision that they are able to override or augment the decisions made by them. Users also want the ability to arrange stations in the community view, but in a community where the locations of stations are common among all users, users do not want others to have the ability to move stations.

MOAs Must Provide Feedback

While users largely accepted automatic filtering and in some cases found it beneficial, they often were puzzled by its actions. For this reason, more has to be done to provide feedback when filters adjust station qualities. For instance, stations boosted in quality due to their popularity with others could be adorned with special symbols to indicate the reason for the change.

MOAs Must Enable Sharing

The most significant feature lacking in SketchWorld was the ability to copy and paste portions of another user's drawing. Being able to view another user's work implies the ability to obtain a copy of it. Sharing is fundamental to communities—people take part in them to receive help from or offer help to one another, so it is natural to assume that some may be willing to offer their services as well as their expertise.

6.2.2 Systems Perspective

For MOAs to be successful, they must solve significant challenges involving the exchange of possibly large amounts of information among users. SketchWorld's filtering contributed to, but was not the complete solution. Fortunately, areas for improvement were identified.

Filtering Did Reduce Traffic

Filtering did reduce the amount of application state data that was exchanged between users. The strategy of filtering at the source of the data proved effective, and the overhead of observers informing the observed side what qualities they wished to receive application state data was negligible.

While filtering proved effective at reducing the quality of application state data, this did not necessarily translate into reduced data volume, as a drawing consisting of line segments resulted in application state messages of identical size regardless of quality. In a general sense, this issue brings forward the importance of the mapping of observation quality to a resulting representation that reflects the quality both in appearance and in the expense to generate, transport, and store it. The treatment of telepointer data illustrates one possible strategy for dealing with information that does not "shrink" well. Telepointer messages in SketchWorld were the same size regardless of the observation quality, so the decision was made to not send telepointer data when quality dropped below a threshold. The result is that at low quality, there is no telepointer data to deal with and the observer likely does not notice, as a telepointer as part of a very small rendition of a drawing would be extremely small.

The telepointer strategy suggests a similar course of action for dealing with drawings in SketchWorld: in low quality versions of a drawing, omit small elements. In the development of other MOAs, the design of low-quality representations of application state should pay attention to how those representations will appear in the community view, and the implications for what needs to be sent in application state messages.

Data Sources Are the Weak Link

In the course of developing optimal strategies for the delivery of application state data, the source, or observed side should be given the highest priority. SketchWorld demonstrated, especially during the traffic simulation, that the observed side will have difficulties keeping up with the demand of sending copies of its application state to many observers.

Dealing With Quality Changes

The significant increases in application state traffic that accompanied changes to station quality illustrated the need to manage this process more effectively. One approach that would reduce traffic during high to low quality transitions is simply to derive the low-quality version of application state locally rather than request it from the observed side. This approach, however, would likely not be feasible in dealing with low to high quality transitions.

Another related problem was the bursts of application state traffic caused by several quality changes occurring within a short time frame. A simple solution would be to limit the rate at which quality changes occur; however, this conflicts with the principle of responsiveness—changes to the MOA should occur as soon as possible after the reasons to make these changes occur. There is likely a balance point to be found, where a maximum quality change rate allows enough responsiveness that users will find acceptable.

6.3 Future Directions

6.3.1 Longer Term Study

While users generally performed well and were satisfied with using a MOA during experiments taking less than an hour to complete, there are a number of reasons to suggest that conducting a longer term study would enhance our understanding of MOAs. First, a long term study would likely nullify "learning curve" effects typical of a short term study. Users would be given the opportunity to thoroughly learn and experience working with a MOA and hopefully better understand the nuances of filtering. Second, a long term study would more thoroughly explore issues such as user acceptance. Would users choose to use SketchWorld or its offspring on a daily basis, and if so, how would they use it? Would a MOA be treated as a peripheral application much like an email or chat program, or would it assume the role of a "main" application where work gets done?

In preparation for a long term study, however, there is much to do. The issues of performance and stability must be tackled, as an application that bogs down one's computer and malfunctions frequently will not be accepted. Many of the recommendations to improve filter configuration, offer more control of the community view, and provide the ability to copy and paste another user's work should be implemented. Finally, a premise for such a long term study must be devised such that participants are motivated to use the MOA.

6.3.2 Other Applications as MOAs

For this thesis, a drawing application was chosen as the basis for the MOA prototype to build and study primarily because representing a user's drawing as a thumbnail drawing in the community view seemed natural and obvious. It seems reasonable to suggest that at least some of the lessons derived from the study of SketchWorld could apply to other graphically-oriented applications, however it is uncertain how well they would apply to other classes of applications. For instance, the feasibility of producing a MOA based on a word processing application is unknown, and this uncertainty has two sources. The first issue is the representation of a text document in miniature form in the community view. What would be an effective way to show a business letter in a 64 by 64 pixel space? The second issue is the nature of the data structures used to represent reduced-quality versions of a text document. These structures should be efficient to create and transmit, and hopefully it should be possible to easily produce a duplicate copy of it at a reduced quality.

6.3.3 MOAs and Large Communities

During the evaluation of SketchWorld, it became apparent that more work needs to be done from a systems perspective to make it and future MOAs feasible for a large community. In several instances, there were delays and slowdowns that indicated difficulties in the production, transmission, and reception of application state updates, with the sender of these updates being particularly burdened. Filtering was a significant step in combating system overload, as there is little doubt that without it SketchWorld would have been unusable even with a few users, however filtering is merely one step in the solution. For instance, when a station is observed by eight other users, the owner of the station may have to produce eight copies of the station's application state, possibly of eight different qualities, and send them to the observers. Two strategies that may be of benefit would be to create proxy application state servers to remove this burden from the observed user's computer, or to maintain a cache of an application state at various qualities so copies would not necessarily be manufactured on demand.

The question of user acceptance of a large community also needs to be investigated. Due to the relatively small size of community tested, navigation and partner finding were two aspects of the community view that were not explored, and in a large community they will play an essential role. The use of filtering in a large community will likely influence user acceptance, as it is anticipated that users will rely on filtering to play a more important role in simplifying their views of community.

Chapter 7
Conclusion

7.1 Summary

The problem examined in this thesis was: due to community size and application complexity, both users and systems may be overwhelmed by the information generated within a MOA. The solution investigated was to use filtering to reduce the amount of information exchanged between users, while still providing the benefits of a MOA to its users. To evaluate the effectiveness of this solution, a prototype MOA, SketchWorld, was built and studied from both the standpoints of users and systems. Evidence was gathered which supported the hypothesis that filtering could help prevent overload while maintaining the benefits of a MOA, and there was also evidence suggesting that users would accept MOAs. The evaluation of SketchWorld also revealed several characteristics of MOAs that should be considered for future research, preferably a study or series of studies involving a longer term evaluation.

7.2 Contributions

MOAs Can Bring Community to the Application The study of SketchWorld suggests that MOAs could be one possible successful integration of application and community, accepted by users and feasible from a systems perspective. Users seemed to accept SketchWorld's presentation of community, as they readily learned how to use the community view that accompanied the working view they were familiar with as a single-user application. SketchWorld was not perfect, but users were quick to suggest reasonable improvements to areas they saw as deficient. The most significant missing feature, the ability to copy from other users' work, suggests the integration of community and application should be stronger than what was done with SketchWorld: sharing is at the core of community, so a MOA should support sharing.

Filters Can Help Prevent Overload, While Preserving Community The study of filtering in SketchWorld suggests that filtering can be used to help prevent overload without significantly diminishing the benefits of community which a MOA is supposed to provide. During the evaluation of SketchWorld, users were able to perform tasks that involved accessing the community even with filtering in place. Furthermore, users did not feel distracted by the actions of filters and still felt that they were in control of their community view while filtering was taking place. The use of several filters at once, however, did lead to some confusion, suggesting that further work needs to be done to simplify filter configuration and to inform the user of the reasons for changes in the community view due to filtering.

SketchWorld As a Reference MOA SketchWorld proved an effective vehicle for the study of MOAs, as it led to evidence suggesting MOAs are feasible and would be accepted by users, and it also exposed several weaknesses and deficiencies in its design that need to be reckoned with in future work. SketchWorld or its components could also be used as the basis for future explorations into MOAs.

MOA Design Guidelines The following guidelines were derived during the design, implementation, and evaluation of SketchWorld:

- *MOAs Must Be Flexible.* MOAs must stretch to accommodate their users, who have varied needs and expectations.

- *MOAs Must Be Controllable.* Users must be able to override automated features such as filtering.

- *MOAs Must Provide Feedback.* When a user's view of community changes, he/she must be given indications of the reasons behind those changes.

- *MOAs Must Enable Sharing.* The ability to share work with other users is an intrinsic expectation of MOAs.

- *Traffic Sources Are Key.* The key battleground to make MOAs feasible from a systems perspective is at the sources of data that is exchanged among users.

7.3 Future Work

The future work proposed is divided into two areas: ideas derived directly from the study and evaluation of SketchWorld, and tangential ideas that are related to MOAs but are not based on the work discussed in this thesis.

7.3.1 Future Work Derived From the Evaluation of SketchWorld

The following future work is based directly on findings from the study of SketchWorld:

- *Enhanced community view.* Experiment with additional adornments for stations in the community view to reflect popularity, activity level, and other factors monitored by SketchWorld's filtering system, with the purpose of explaining to users the reasons for filter actions. Permit copy and paste between users.

- *Customizable community view.* In response to comments from study participants, allow user customization such as configuring the display order of the station list to indicate station popularity, activity level, and other criteria. Provide the user with the ability to locally customize the location of stations, or switch to a shared view where all stations appear in the same locations for all users. Provide the user with the ability to do work within the community view rather than in a separate window.

- *"Smarter" application states.* Attempt to lessen the burden on remote users by making better use of locally-held application state information. For instance, derive a lower-quality version of the application state rather than requesting it from the remote user.

- *Different forms of quality.* Experiment with the use of sparse application states (omitting insignificant details), update frequency, or other means of reducing the transfer of data at lower qualities.

7.3.2 Other Future Work

The following future work is not based directly on work done in this thesis, but may otherwise be of interest for future MOA-related research:

- *Show admirers.* In the community view, display interested users alongside the stations they are interested in.

- *Peer to peer architecture.* Study the possible advantages of a peer to peer architecture over SketchWorld's client/server architecture, such as redundancy and load distribution.

- *Separate community and application.* Study an architecture consisting of a standalone "community browser" capable of supporting several types of "community-ready" applications that communicate with the community browser.

- *Filter according to system load.* Monitor system load factors and enact filtering to keep them within acceptable levels. For instance, an instance of SketchWorld would resort to sending lower-quality updates if it detected problems sending updates to several observers.

- *Other applications as MOAs.* Attempt to build MOAs for word processing or chess to study different application state requirements and different interpretations of quality.

References

[1] Dzmitry Aliakseyeu, Sriram Subramanian, Jean-Bernard Martens, and Matthias Rauterberg. Interaction techniques for navigation through and manipulation of 2d and 3d data. In *Proceedings of the workshop on Virtual environments 2002*, pages 179–188. Eurographics Association, 2002.

[2] Gary E. Anderson, T. C. Nicholas Graham, and Timothy N. Wright. Dragonfly: linking conceptual and implementation architectures of multiuser interactive systems. In *ICSE '00: Proceedings of the 22nd international conference on Software engineering*, pages 252–261, New York, NY, USA, 2000. ACM Press.

[3] Ronald M. Baecker, Dimitrios Nastos, Ilona R. Posner, and Kelly L. Mawby. The user-centered iterative design of collaborative writing software. In *CHI '93: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 399–405, New York, NY, USA, 1993. ACM Press.

[4] Steve Benford, John Bowers, Lennart E. Fahlén, Chris Greenhalgh, and Dave Snowdon. User embodiment in collaborative virtual environments. In *Proc. ACM Conf. Human Factors in Computing Systems, CHI*, volume 1, pages 242–249, 1995.

[5] Steve Benford, C. Greenhalgh, T. Rodden, and J. Pycock. Collaborative virtual environments. *Communications of the ACM*, pages 79–85, July 2001.

[6] Stephen A. Brewster. Using nonspeech sounds to provide navigation cues. *ACM Trans. Comput.-Hum. Interact.*, 5(3):224–259, 1998.

[7] M. L. Brown, S. L. Newsome, and E. P. Glinert. An experiment into the use of auditory cues to reduce visual workload. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 339–346. ACM Press, 1989.

[8] Kirsten Cater, Alan Chalmers, and Colin Dalton. Varying rendering fidelity by exploiting human change blindness. In *Proceedings of the 1st international conference on Computer graphics and interactive techniques in Austalasia and South East Asia*, pages 39–46. ACM Press, 2003.

[9] Kirsten Cater, Alan Chalmers, and Patrick Ledda. Selective quality rendering by exploiting human inattentional blindness: looking but not seeing. In *VRST '02: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 17–24, New York, NY, USA, 2002. ACM Press.

[10] *Using a Landscape Metaphor to Represent a Corpus of Documents*, volume 716 of *LNCS*, 1993.

[11] Elizabeth F. Churchill and Sara Bly. Virtual environments at work: ongoing use of muds in the workplace. In *Proceedings of the international joint conference on Work activities coordination and collaboration*, pages 99–108. ACM Press, 1999.

[12] Andrew Cockburn and Saul Greenberg. Making contact: getting the group communicating with groupware. In *Proceedings of the conference on Organizational computing systems*, pages 31–41. ACM Press, 1993.

[13] Andy Cockburn. Revisiting 2d vs 3d implications on spatial memory. In *Proceedings of the fifth conference on Australasian user interface*, pages 25–31. Australian Computer Society, Inc., 2004.

[14] Julian Dibbell. A rape in cyberspace. *Village Voice*, XXXVIII(51):36–42, December 1993.

[15] Judith S. Donath. Visual who: animating the affinities and activities of an electronic community. In *MULTIMEDIA '95: Proceedings of the third ACM international conference on Multimedia*, pages 99–107. ACM Press, 1995.

[16] Paul Dourish and Victoria Bellotti. Awareness and coordination in shared workspaces. In *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, pages 107–114. ACM Press, 1992.

[17] Clarence A. Ellis, Simon J. Gibbs, and Gail Rein. Groupware: some issues and experiences. *Commun. ACM*, 34(1):39–58, 1991.

[18] Thomas Erickson, David N. Smith, Wendy A. Kellogg, Mark Laff, John T. Richards, and Erin Bradner. Socially translucent systems: social proxies, persistent conversation, and the design of ̀babble ́. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 72–79, New York, NY, USA, 1999. ACM Press.

[19] Bernd Fröhlich and John Plate. The cubic mouse: a new device for three-dimensional input. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 526–531. ACM Press, 2000.

[20] L. Garton and B. Wellman. Social impacts of electronic mail in organizations: A review of the research literature. *Communication Yearbook*, 18, 1995.

[21] Saul Greenberg and David Marwood. Real time groupware as a distributed system: concurrency control and its effect on the interface. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 207–217, New York, NY, USA, 1994. ACM Press.

[22] C. Greenhalgh and Steve Benford. Massive: A collaborative virtual environment for teleconferencing. *ACM Trans. on Computer-Human Interaction*, 2(3):239–261, September 1995.

[23] J. Grudin and S.E. Poltrock. Computer-supported cooperative work and groupware. In M. Zelkowitz, editor, *Advances in Computers*. Academic Press, Orlando, FL, 1997.

[24] Robert Hagedorn, editor. *Sociology*. Holt, Rinehart and Winston of Canada, Limited, 1980.

[25] Jungpil Hahn. The dynamics of mass online marketplaces: a case study of an online auction. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 317–324. ACM Press, 2001.

[26] Mark Handel and James D. Herbsleb. What is chat doing in the workplace? In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 1–10. ACM Press, 2002.

[27] Patrick Hanks, editor. *The Collins Dictionary of the English Language*. Wm. Collins Publishers Pty. Ltd., 1985.

[28] Randall W. Hill, Jr. Perceptual grouping and attention in a multi-agent world. In *Proceedings of the third annual conference on Autonomous Agents*, pages 418–419. ACM Press, 1999.

[29] A. Hillman. *Community, organization, and planning*. McMillan, 1950.

[30] Ellen A. Isaacs, John C. Tang, and Trevor Morris. Piazza: a desktop environment supporting impromptu and planned interactions. In *Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 315–324. ACM Press, 1996.

[31] Jr. Joseph J. LaViola. A discussion of cybersickness in virtual environments. *SIGCHI Bull.*, 32(1):47–56, 2000.

[32] Matthew Kam, Jingtao Wang, Alastair Iles, Eric Tse, Jane Chiu, Daniel Glaser, Orna Tarshish, and John Canny. Livenotes: a system for cooperative and augmented note-taking in lectures. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 531–540, New York, NY, USA, 2005. ACM Press.

[33] Robert Kraut, Carmen Egido, and Jolene Galegher. Patterns of contact and communication in scientific research collaboration. In *Proceedings of the 1988 ACM conference on Computer-supported cooperative work*, pages 1–12. ACM Press, 1988.

[34] J. Chris Lauwers and Keith A. Lantz. Collaboration awareness in support of collaboration transparency: requirements for the next generation of shared window systems. In *CHI '90: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 303–311, New York, NY, USA, 1990. ACM Press.

[35] D. R. Montello. A new framework for understanding the acquisition of spatial knowledge in large-scale environments. *Spatial and Temporal Reasoning in Geographic Information Systems*, 27(4):741–750, 1999.

[36] Jonathan Munson and Prasun Dewan. A concurrency control framework for collaborative systems. In *CSCW '96: Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 278–287, New York, NY, USA, 1996. ACM Press.

[37] John F. Patterson. A taxonomy of architectures for synchronous groupware applications. *SIGOIS Bull.*, 15(3):27–29, 1995.

[38] John F. Patterson, Mark Day, and Jakov Kucan. Notification servers for synchronous groupware. In *CSCW '96: Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 122–129, New York, NY, USA, 1996. ACM Press.

[39] W.G. Phillips. Architectures for synchronous groupware. Technical Report 1999-425, Queen's University, Kingston, Ontario, Canada, May 1999. Available from `www.cs.queensu.ca`.

[40] Atul Prakash and Michael J. Knister. A framework for undoing actions in collaborative systems. *ACM Trans. Comput.-Hum. Interact.*, 1(4):295–330, 1994.

[41] K. J. Pulo. Recursive space decompositions in force-directed graph drawing algorithms. In *APVis '01: Proceedings of the 2001 Asia-Pacific symposium on Information visualisation*, pages 95–102, Darlinghurst, Australia, Australia, 2001. Australian Computer Society, Inc.

[42] Howard Rheingold. *The Virtual Community: Homesteading on the Electronic Frontier*. MIT Press, Cambridge, Massachusetts, 2000.

[43] Teresa L. Roberts. Are newsgroups virtual communities? In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 360–367. ACM Press/Addison-Wesley Publishing Co., 1998.

[44] T. Rodden and G. Blair. The problem of control. In *Proceedings of the ECSCW European Conference on Computer Supported Cooperative Work*, pages 49–64, Amsterdam, 1991. Klewar Press.

[45] Tom Rodden. Populating the application: a model of awareness for cooperative applications. In *Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 87–96. ACM Press, 1996.

[46] Hanan Samet. Data structures for quadtree approximation and compression. *Commun. ACM*, 28(9):973–993, 1985.

[47] Manojit Sarkar and Marc H. Brown. Graphical fisheye views of graphs. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 83–91, New York, NY, USA, 1992. ACM Press.

[48] L. F. Schnore. *Sociology, an introduction*, chapter Community. John Wiley, 1973.

[49] Jouni Smed, Timo Kaukoranta, and Harri Hakonen. A review on networking and multiplayer computer games. Technical Report 454, Turku Centre for Computer Science, April 2002.

[50] David A. Smith, Alan Kay, Andreas Raab, and David P. Reed. Croquet - a collaboration system architecture. *c5*, 00:2, 2003.

[51] Marc A. Smith, Shelly D. Farnham, and Steven M. Drucker. The social life of small graphical chat spaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 462–469. ACM Press, 2000.

[52] Randall B. Smith. Experiences with the alternate reality kit: an example of the tension between literalism and magic. In *Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface*, pages 61–67. ACM Press, 1987.

[53] Randall B. Smith, Ranald Hixon, and Bernard Horan. Supporting flexible roles in a shared space. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, pages 197–206. ACM Press, 1998.

[54] Siriwan Suebnukarn and Peter Haddawy. A collaborative intelligent tutoring system for medical problem-based learning. In *IUI '04: Proceedings of the 9th international conference on Intelligent user interfaces*, pages 14–21, New York, NY, USA, 2004. ACM Press.

[55] Monica Tavanti and Mats Lind. 2d vs 3d, implications on spatial memory. In *Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS'01)*, page 139. IEEE Computer Society, 2001.

[56] J. Thomas, K. Cook, V. Crow, B. Hetzler, R. May, D. McQuerry, R. McVeety, N. Miller, G. Nakamura, L. Nowell, P. Whitney, and P. Wong. Human computer interaction with global information spaces - beyond data mining, 1999.

[57] P. Thorndyke and B. Hayes-Roth. Differences in spatial knowledge acquired from maps and navigation. *Cognitive Psychology*, 14:560–589, 1982.

[58] Melanie Tory, Torsten Moller, M. Stella Atkins, and Arthur E. Kirkpatrick. Combining 2d and 3d views for orientation and relative position tasks. In *Proceedings of the 2004 conference on Human factors in computing systems*, pages 73–80. ACM Press, 2004.

[59] F. B. Viegas and J. S. Donath. Chat circles. In *Proceedings of the CHI 99 conference on Human factors in computing systems*, pages 9–16. ACM CHI, March 1999.

[60] Barry Wellman. For a social network analysis of computer networks: a sociological perspective on collaborative work and virtual community. In *SIGCPR '96: Proceedings of the 1996 ACM SIGCPR/SIGMIS conference on Computer personnel research*, pages 1–11. ACM Press, 1996.

[61] Barry Wellman. Virtual community: introducing a new siggroup focus area. *SIGGROUP Bull.*, 19(1):18–20, 1998.

[62] C. D. Wickens and J. G. Hollands. *Engineering Psychology and Human Performance*. Prentice Hall, Upper Saddle River, New Jersey, 2000.

[63] Jieping Ye, Ravi Janardan, and Qi Li. Gpca: an efficient dimension reduction scheme for image compression and retrieval. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 354–363, New York, NY, USA, 2004. ACM Press.

[64] Xiaolong Zhang and George W. Furnas. Social interactions in multiscale cves. In *Proceedings of the 4th International Conference on Collaborative Virtual Environments*, pages 31–38, September 2002.

Appendix A

Qualitative Experiment Questionnaires and Responses

A.1 Consent Form

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF SASKATCHEWAN
INFORMED CONSENT FORM

You are invited to participate in a study entitled "Reducing Overload in Multiuser Online Applications". Please read this form carefully, and feel free to ask questions you might have.

Researcher(s):

Carl Gutwin, Department of Computer Science (966-8646)
gutwin@cs.usask.ca
Roger Blum, Department of Computer Science (966-2327)
rkb479@mail.usask.ca

Purpose and Procedure: The purpose of this study is to examine a potential overload problem with Multiuser Online Applications (MOAs). A MOA allows users to not only perform their own work, but also observe and converse with other users and overload may occur when there are too many users, or the users are trying to do too much work for the application to handle.

For this study, you and several other participants will be asked to perform a series of tasks using a MOA that allows you to create drawings and be able to observe the drawing activities of the other participants. In addition, you will be able to use a built-in chat interface to converse with the other participants.

Your goals for this study are to first to experiment with the controls that govern how you will be able to view the work of others and determine the settings that work best for you, and second to evaluate how well the chosen settings work.

There will be a brief presentation to all participants as a group before the study begins to explain how the application works, during which time you will be free to ask questions. If you feel more comfortable asking questions in an individual setting, there will be time alloted for that immediately following the presentation.

During the study, you will be given two questionnaires to fill out in privacy. None of the other participants will see the responses you provide to them. The first questionnaire will ask you for demographic information such as your age and gender, while in the second questionnaire you will assess the performance of the application. The session will take up to 60 minutes to complete. At the end of the session, you will be given more information about the purpose and goals of the study, and there will be time for you to discuss the experiment and ask questions about the research. This debriefing session will involve all participants. If you wish to discuss the study further with the researcher in an individual setting, you may do so after the debriefing session. There will be questions asked to the group during the debriefing with the intent of promoting discussion. You are free to respond to them in the group setting or in private with the researcher afterwards, or not at all.

Potential Risks: Risks to you may include frustration and fatigue from performing the experiment. To minimize these risks, you will be given break periods between each task and advised you may choose to quit the experiment at any time without any prejudice or penalty. If you wish to take unscheduled breaks during the experiment, you may also do so. You will also be reminded that your performance of tasks will reflect the merits of the system being used and not your own expertise.

Potential Benefits: You will receive a \$10.00 payment for your participation. As one way of thanking you for your time, we will be pleased to make available to you a summary of the results of this study once they have been compiled (they will be made available on the HCI web site, hci.usask.ca). This summary will outline the research and discuss our findings and recommendations.

For the wider community, the study hopes to demonstrate whether or not the methods identified to avoid user and system overload can be effective in a Multiuser Online Application.

Storage of Data: The research materials (data collected, questionnaires and observations) will be stored with complete security by the research supervisor at the Department of Computer Science for a minimum of five years following the investigation.

Confidentiality: The data collected from this study will be used in articles for publication in journals and conference proceedings. All of the information we collect from you (data logged by the computer, observations made by the experimenters, and your questionnaire responses) will be stored so that your name is not associated with it (using an arbitrary participant number). Any write-ups of the data will not include any information that can be linked directly to you.

Right to Withdraw: Your participation is voluntary, and you may withdraw from the study for any reason, at any time, without penalty of any sort, and without losing any advertised benefits. Withdrawal from the study will not affect your academic status or your access to services at the university. If you withdraw, your data will be deleted from the study and destroyed. In addition, you are free to not answer specific items or questions on questionnaires. If you withdraw from the study at any time, any data that you have contributed will be destroyed at your request.

Questions: If you have any questions concerning the study, please feel free to ask at any point; you are also free to contact the researchers at the numbers provided above if you have questions at a later time. This study has been approved on ethical grounds by the University of Saskatchewan Behavioural Research Ethics Board on June 20, 2006. Any questions regarding your rights as a participant may be addressed to that committee through the Ethics Office (966-2084). Out of town participants may call collect. Once results of the study are known, they will be published on the HCI web site (http://hci.usask.ca).

Consent to Participate: I have read and understood the description provided above; I have been provided with an opportunity to ask questions and my questions have been answered satisfactorily. I consent to participate in the study described above, understanding that I may withdraw this consent at any time. A copy of this consent form has been given to me for my records.

(Name of Participant)	(Date)
(Signature of Participant)	(Signature of Researcher)

A.2 Participant Profiles

A.2.1 Profile Form

Questionnaire # 1 - Participant Information

Participant #:

Age:

Occupation/University Major:

Gender (circle one): Male Female

Approximately how many hours per week do you use a computer?:

Approximately how many hours per week do you play computer games?:

Approximately how many hours per week do you use drawing programs?:

List the drawing applications you have used:

A.2.2 Profile Responses

Participant	Age	Occupation/Major	Gender	Computer hrs/wk	Computer games hrs/wk	Drawing programs hrs/wk	Drawing apps used
1	23	Nursing	F	25	0	0	-
2	22	Research Assistant/Psychology	F	40	0	0	Microsoft Paint
3	23	Psychology Graduate Student	F	40	0	0	-
4	32	Computer Science	M	40+	4	1	Photoshop MS Photo Editor MS Paint
5	27	Computer Science, MSc	M	60	2	1	MS Paint Photoshop Illustrator
6	25	Computer Science	M	50	4	2	Omnigraffle Photoshop Paintshop Illustrator MS Paint Gimp
7	50	Computer Science	M	20	0.25	0.01	Corel Draw Paint
8	26	Computer Science	M	30	2	1	Windows Paint

Table A.1: Participant Profile Responses

A.3 Task Responses

A.3.1 Task Response Form

Questionnaire - Phase 2

Participant #:

Configuration used:

Did the configuration behave as expected? Yes / No
Reasons/Explanation:

Did the configuration help you complete the task? Yes / No
Reasons/Explanation:

Were you able to maintain awareness of other users? Yes / No
Reasons/Explanation:

Were you able to determine the identity of other users? Yes / No
Reasons/Explanation:

Were you able to maintain focus on interesting users? Yes / No
Reasons/Explanation:

Were you able to control the level of detail you saw of other users? Yes / No
Reasons/Explanation:

Did you feel overloaded by the information about others shown to you? Yes / No
Reasons/Explanation:

Were you able to keep track of what others were doing? Yes / No
Reasons/Explanation:

Did you enjoy performing the task using the configuration? Yes / No
Reasons/Explanation:

Was there significant mental demand required to perform the task? Yes / No
(thinking, deciding, calculating, remembering, looking, searching, etc.)
Reasons/Explanation:

Was there significant physical demand required to perform the task? Yes / No
(pushing, pulling, turning, controlling, activating, etc.)
Reasons/Explanation:

Was there significant temporal demand required to perform the task? Yes / No
(time pressure due to pace of the task)
Reasons/Explanation:

Was there significant effort required to perform the task? Yes / No
(mental, physical work)
Reasons/Explanation:

Did you feel your performance was good? Yes / No
(accomplishing goals of the task)
Reasons/Explanation:

Did you feel frustrated while performing the task? Yes / No
(insecure, discouraged, irritated, stressed, annoyed, etc.)
Reasons/Explanation:

Comments:

A.3.2 MOA Functionality Questions

Part	Beh.	Help	Aw.	Ident.	Foc.	Cont.	Overl.	Track	Enjoy
1	Y	N	N	Y	Y	Y	N	N	Y
2	-	N	N	Y	N	Y	N	Y	N
3	Y	N	Y	Y	N	Y	N	Y	Y
4	Y	-	Y	Y	Y	N	N	Y	Y
5	Y	N	Y	Y	Y	Y	N	Y	N
6	Y	Y	Y	Y	Y	Y	N	Y	Y
7	-	Y	Y	Y	N	Y	Y	Y	Y
8	Y	N	Y	N	Y	N	N	N	N

Table A.2: Functionality Question Responses

A.3.3 Task Load Questions

Part	Mental	Physical	Temporal	Effort	Performance	Frustration
1	N	N	N	N	Y	N
2	N	N	N	N	Y	N
3	N	N	N	N	N	N
4	N	N	N	N	Y	N
5	N	N	N	N	Y	N
6	N	N	N	N	Y	Y
7	Y	N	N	-	N	N
8	Y	N	N	N	Y	N

Table A.3: Task Load Question Responses

A.3.4 Participants' Filter Configurations

Participant	4	5	6	7	8
Bias (%)	100	0	0	0	0
Recent Visit Weight/	50	0	0	10	100
Strength (%)	100	0	0	100	100
Frequency Weight/	90	0	0	10	0
Strength (%)	100	0	0	100	0
Mutual Interest Weight/	100	0	0	70	0
Strength (%)	90	0	0	70	0
Random Selection Weight/	0	0	0	20	0
Strength (%)	50	0	0	100	0
Social Network Weight/	0	0	50	40	0
Strength (%)	50	0	80	100	0
Community Rating Weight/	0	100	50	30	0
Strength (%)	50	100	80	100	0
Activity Level Weight/	0	0	50	20	0
Strength (%)	60	0	80	100	0
Current Tool Weight/	0	0	30	10	0
Strength (%)	60	0	40	100	0

Appendix B

Quantitative Analysis - Station Graphs

B.1 Data Received by User 4

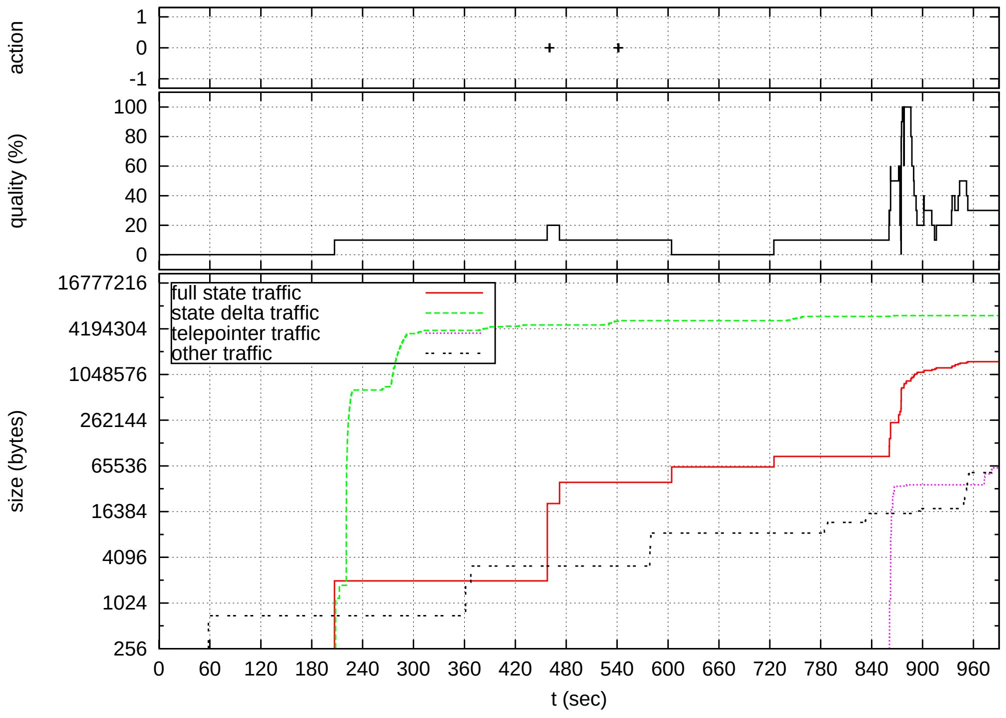


Figure B.1: Traffic received by User 4 from User 5.

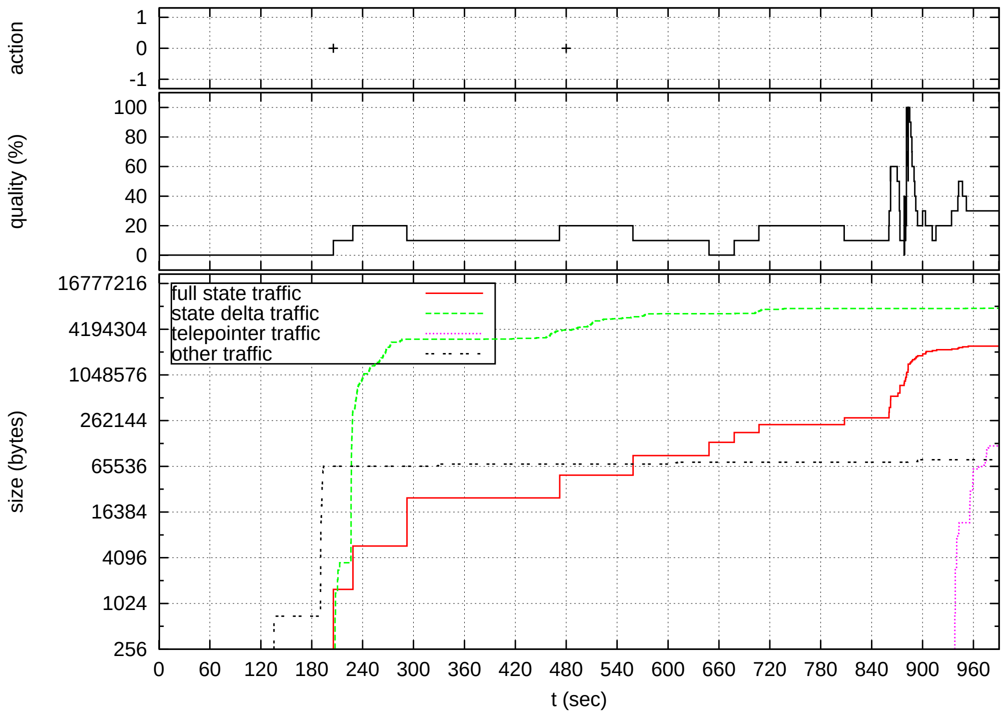


Figure B.2: Traffic received by User 4 from User 6.

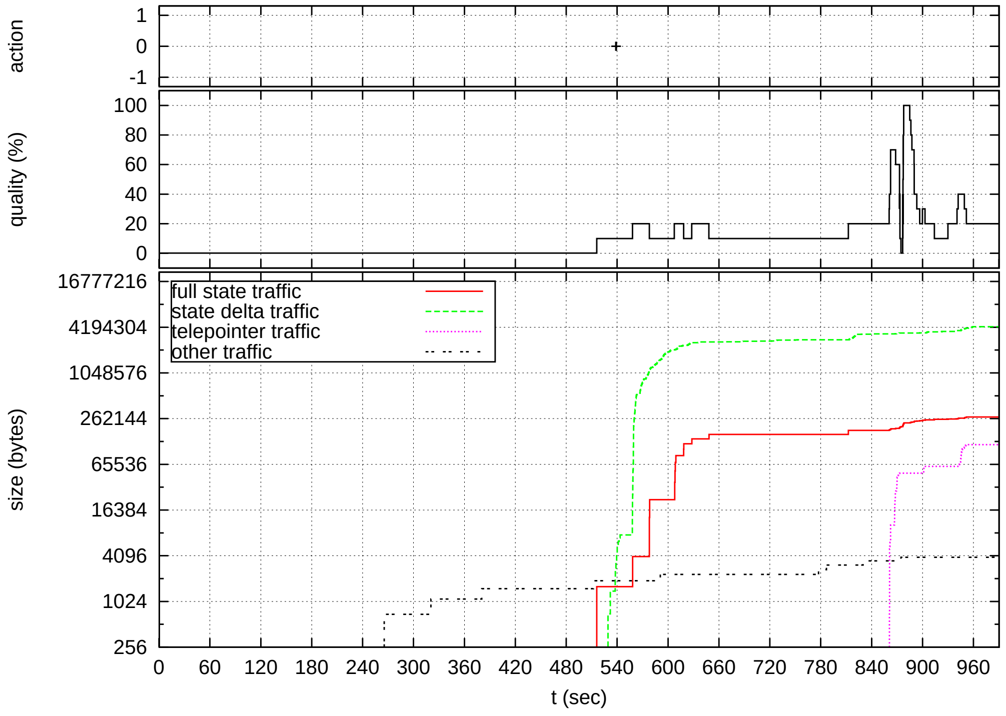


Figure B.3: Traffic received by User 4 from User 7.

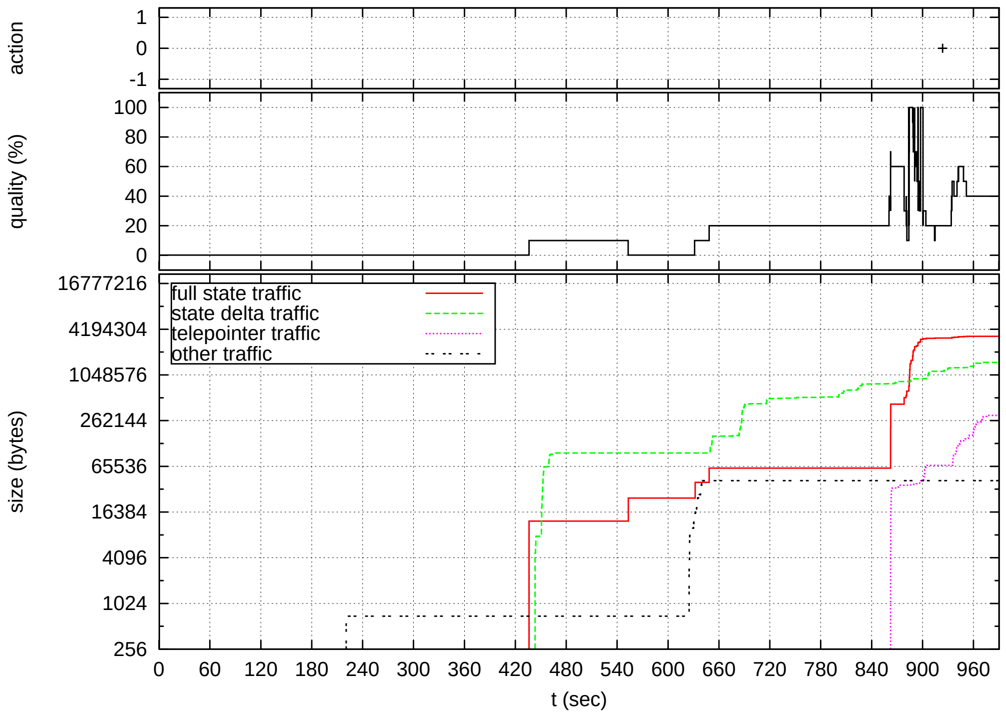


Figure B.4: Traffic received by User 4 from User 8.

B.2 Data Received by User 5

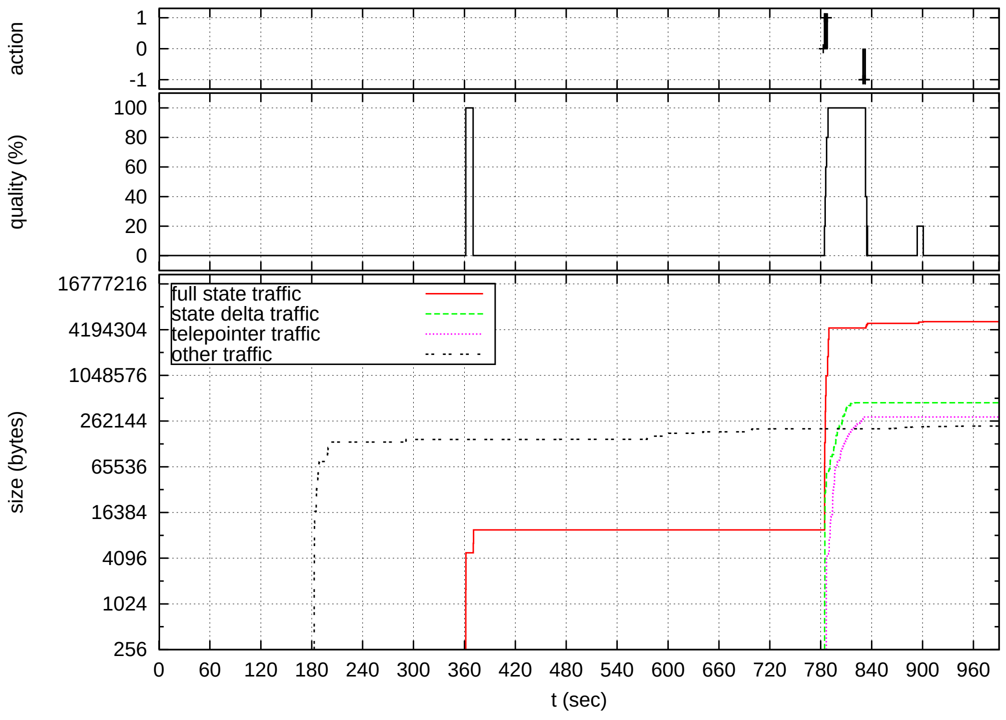


Figure B.5: Traffic received by User 5 from User 4.

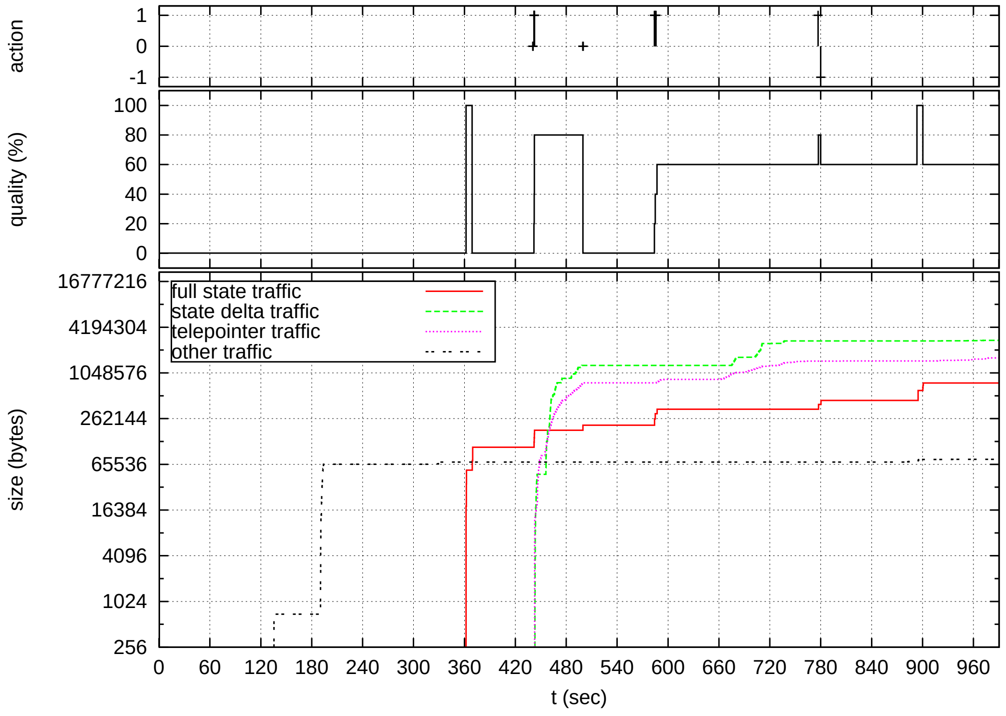


Figure B.6: Traffic received by User 5 from User 6.

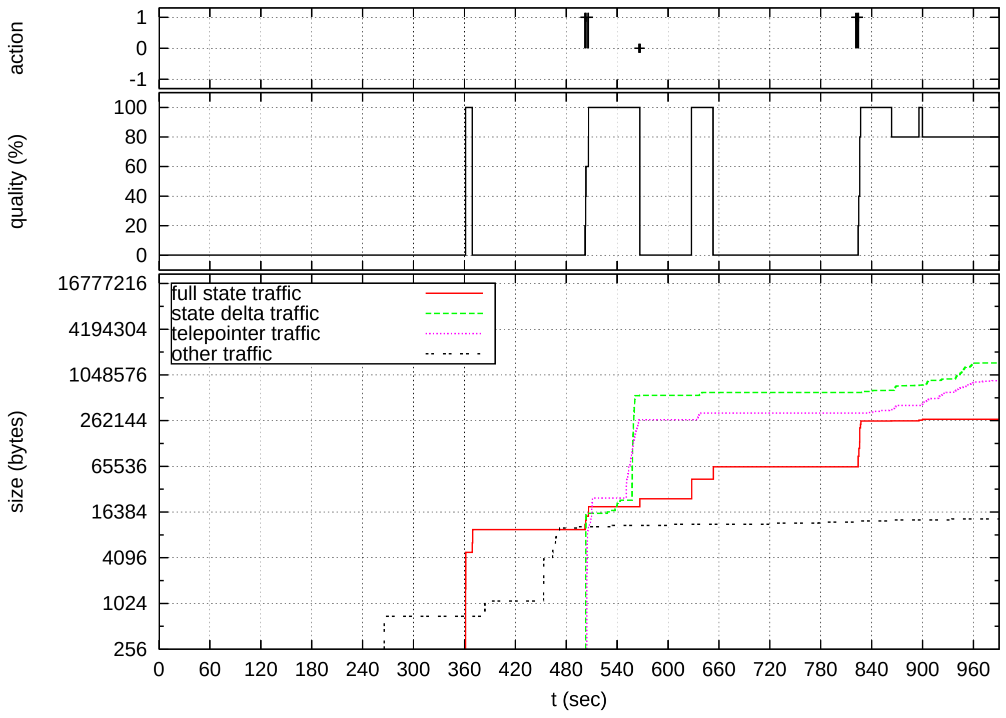


Figure B.7: Traffic received by User 5 from User 7.

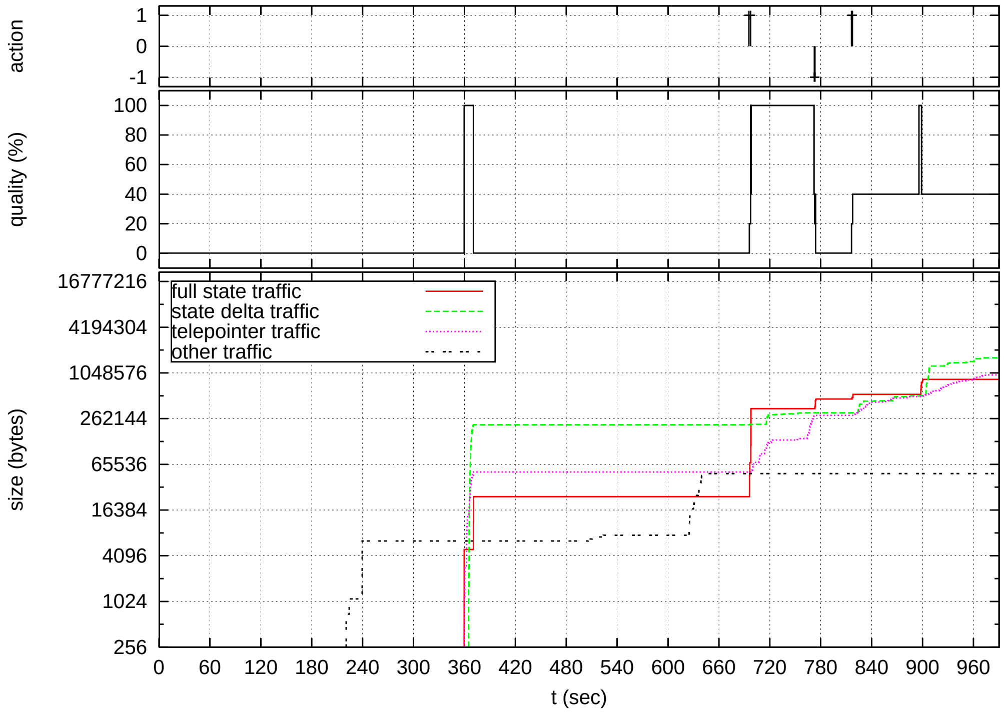


Figure B.8: Traffic received by User 5 from User 8.

B.3 Data Received by User 6

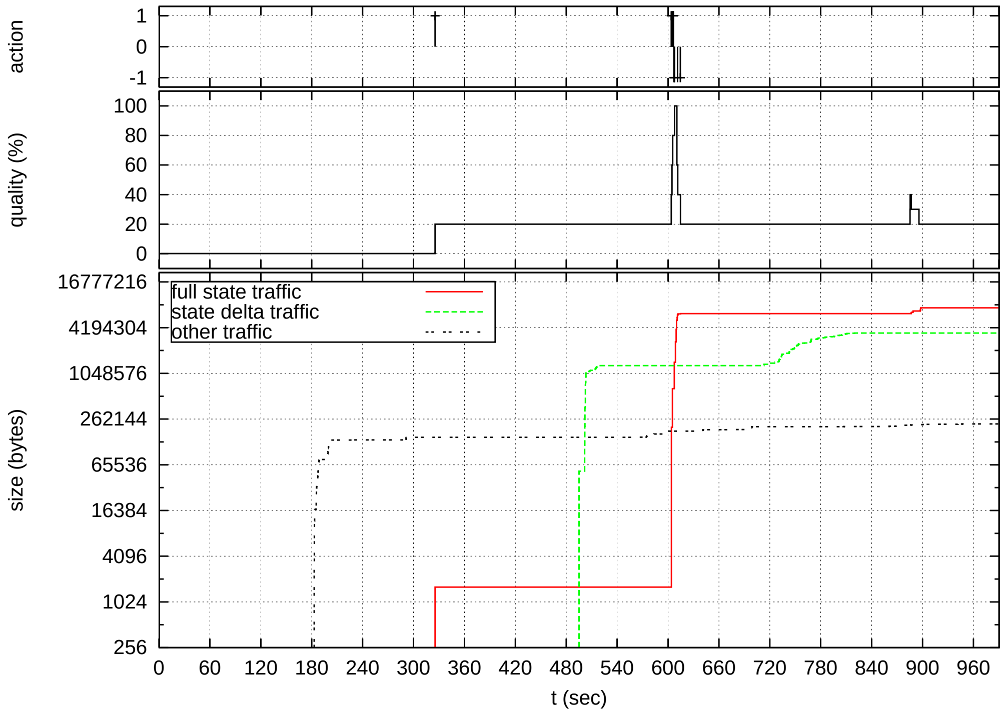


Figure B.9: Traffic received by User 6 from User 4.

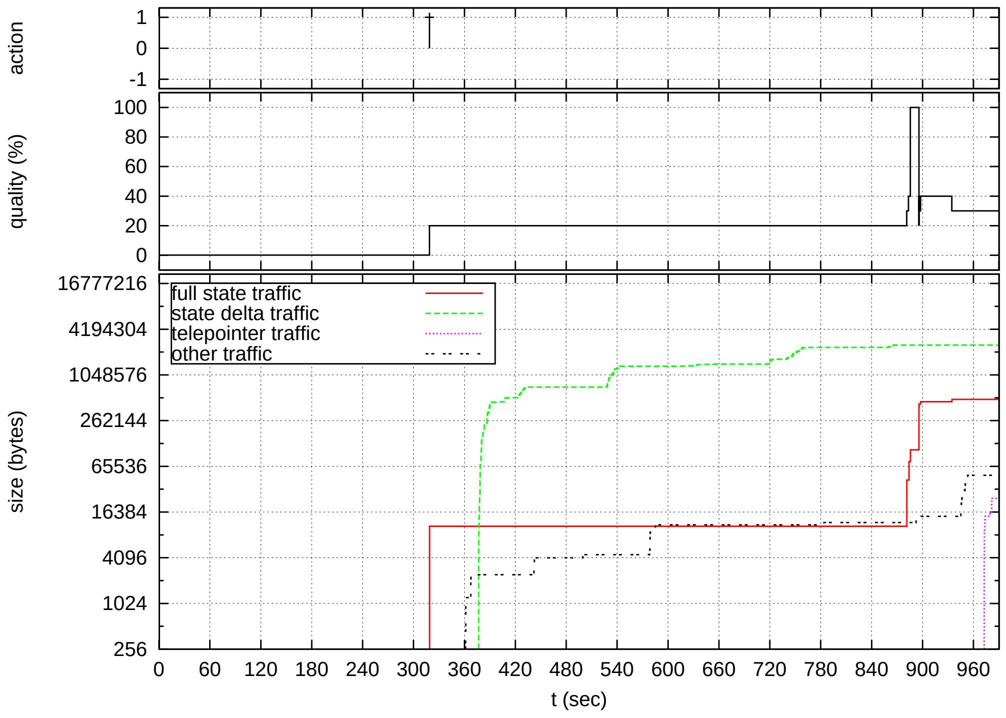


Figure B.10: Traffic received by User 6 from User 5.

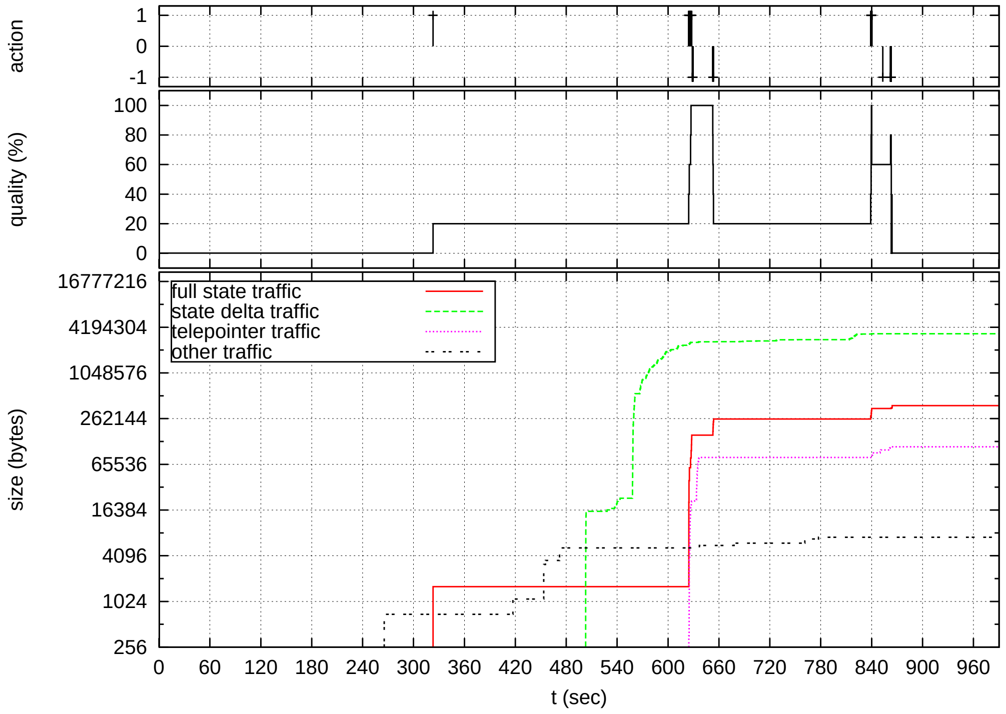


Figure B.11: Traffic received by User 6 from User 7.

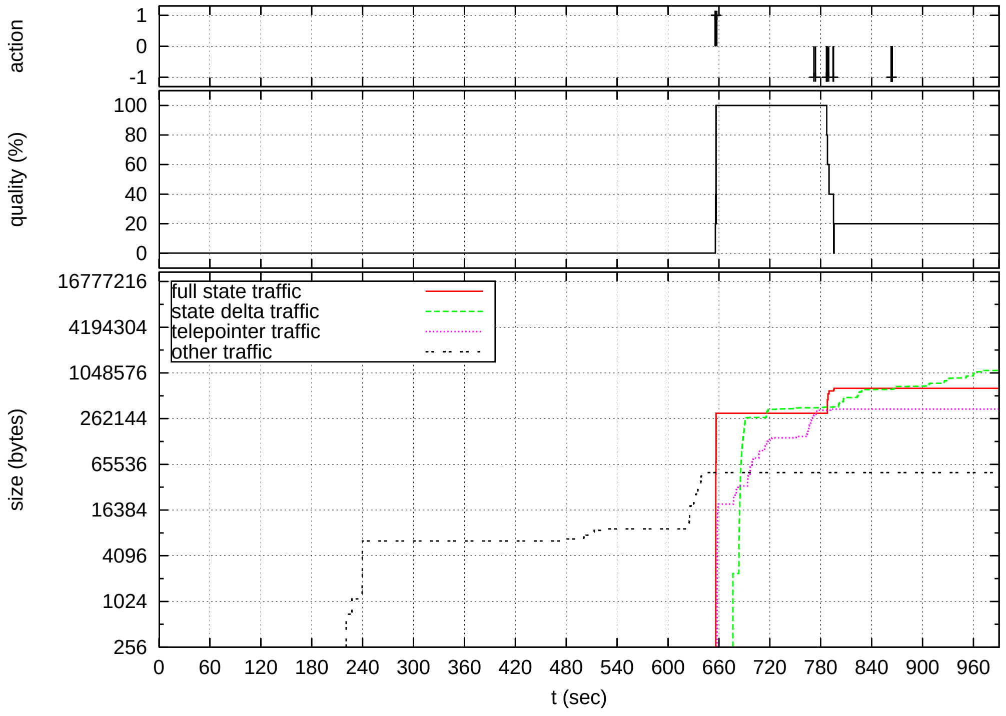


Figure B.12: Traffic received by User 6 from User 8.

B.4 Data Received by User 7

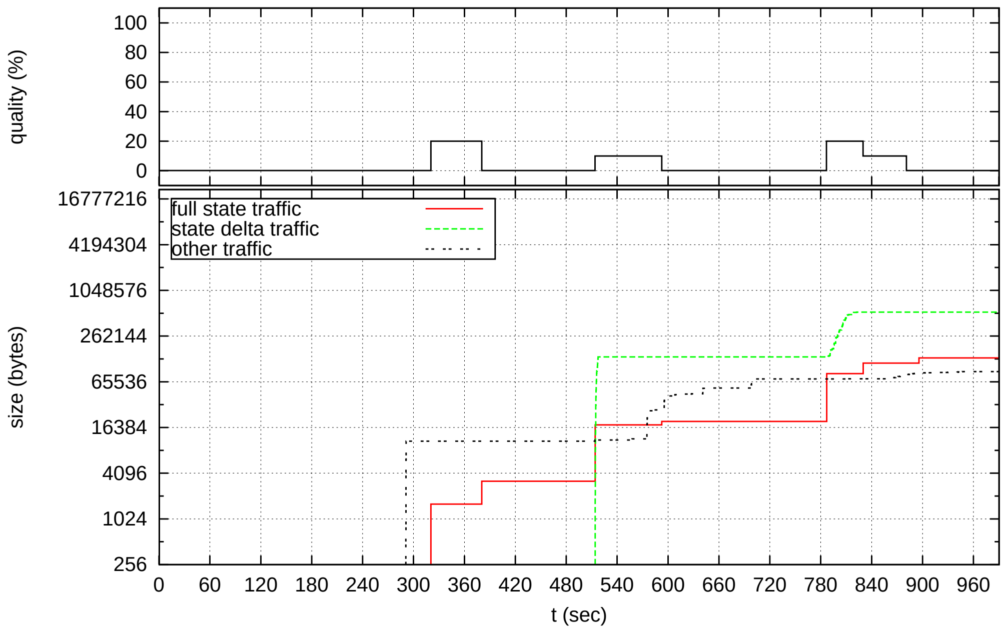


Figure B.13: Traffic received by User 7 from User 4.

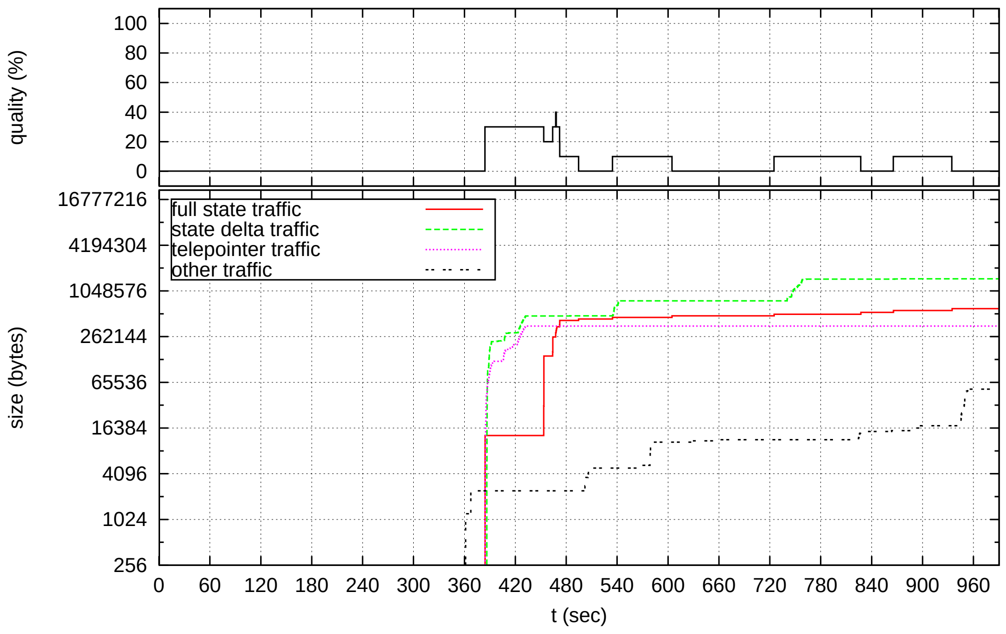


Figure B.14: Traffic received by User 7 from User 5.

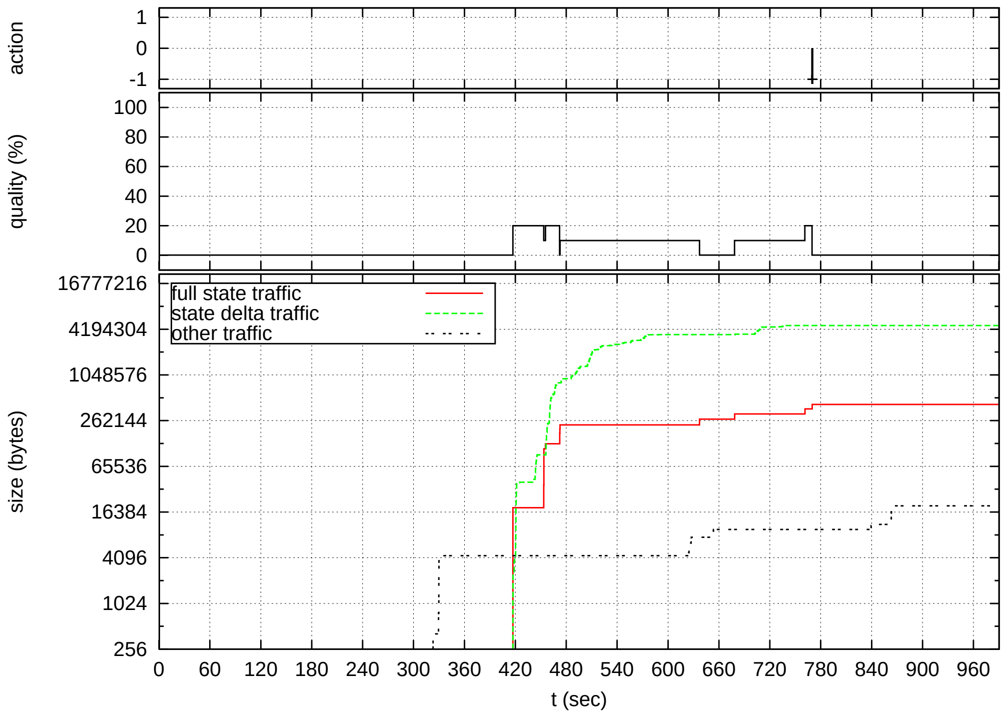


Figure B.15: Traffic received by User 7 from User 6.

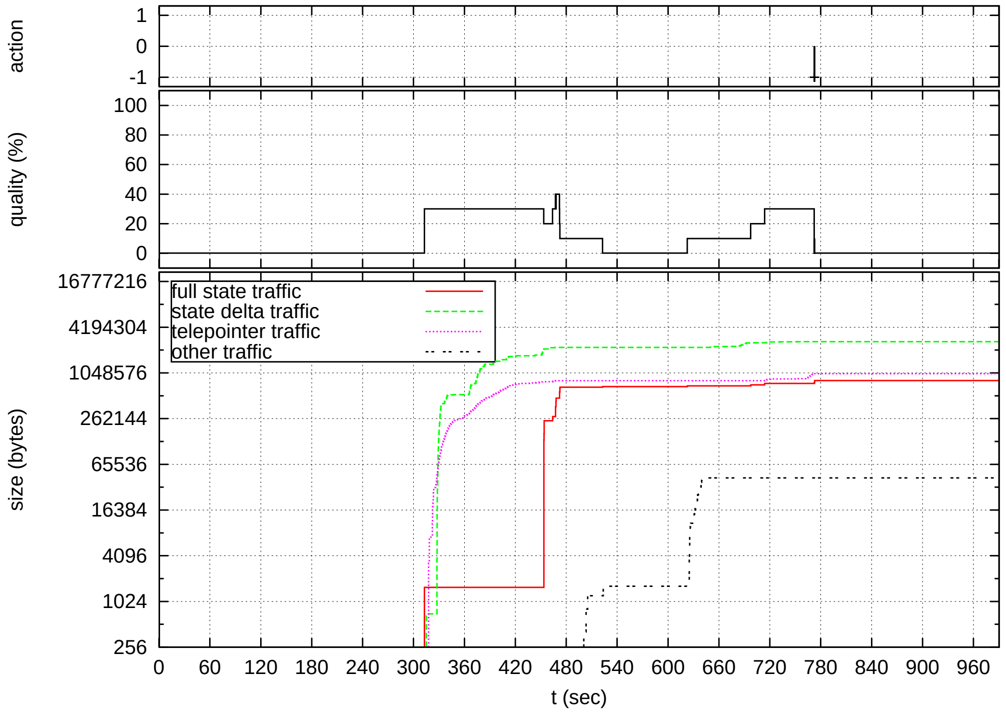


Figure B.16: Traffic received by User 7 from User 8.

B.5 Data Received by User 8

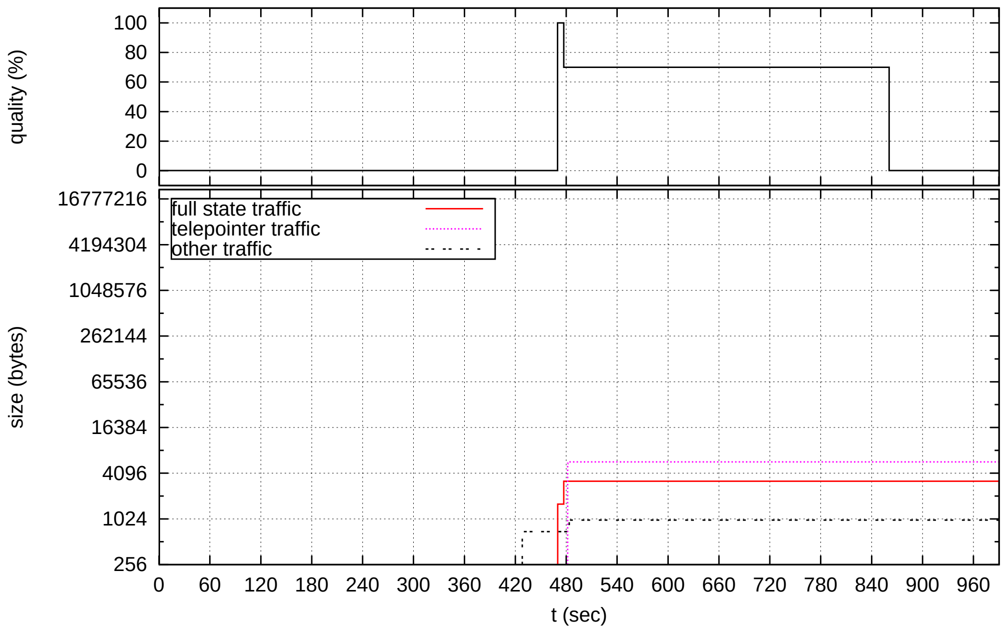


Figure B.17: Traffic received by User 8 from User 4.

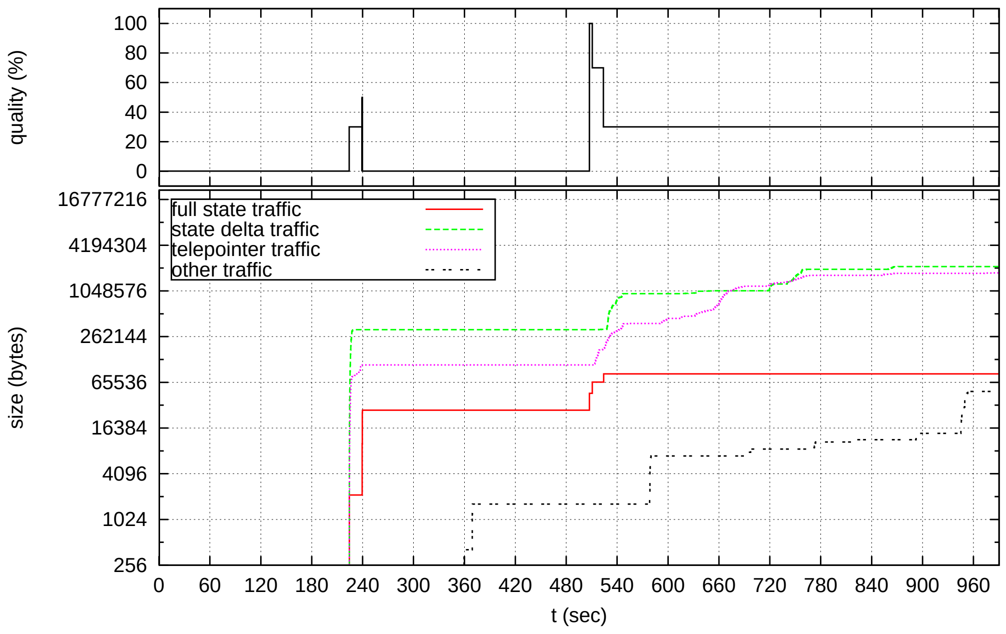


Figure B.18: Traffic received by User 8 from User 5.

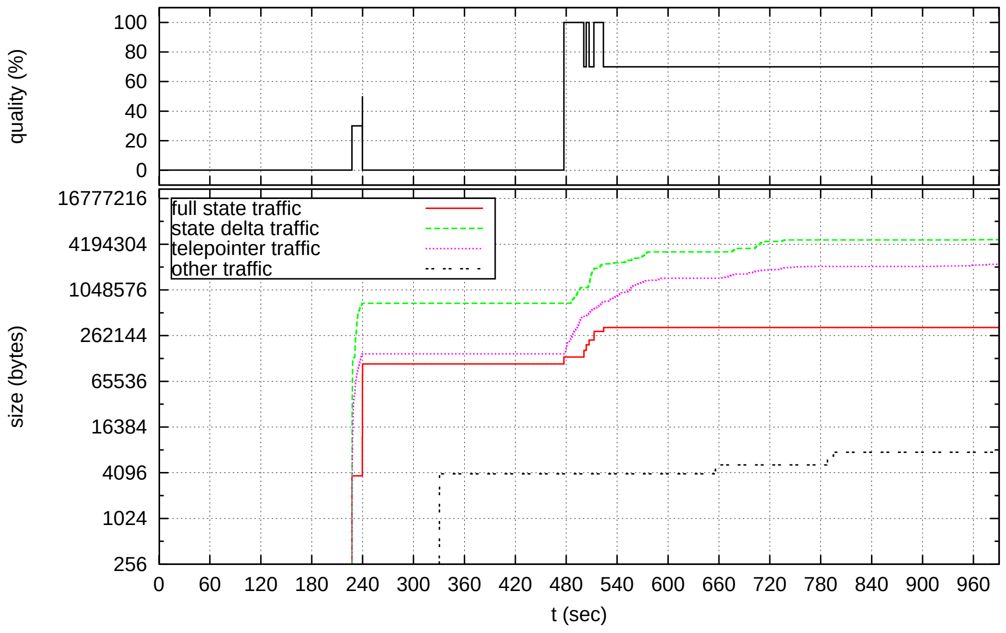


Figure B.19: Traffic received by User 8 from User 6.

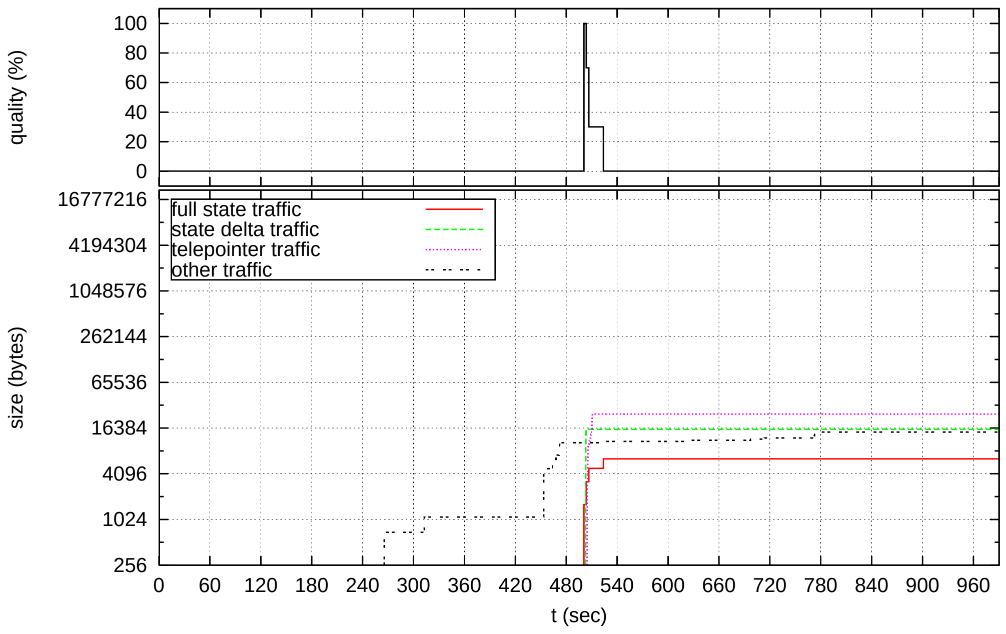


Figure B.20: Traffic received by User 8 from User 7.