A Thesis

entitled

A Low Cost Interactive System for Distance Learning

by

Prashanth R Reddy

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Master
of Science Degree in Electrical Engineering

_______________________________________
Dr. Patricia R. Komuniecki, Dean
College of Graduate Studies

The University of Toledo

December 2011
An Abstract of

A Low Cost Interactive System for Distance Learning

by

Prashanth R Reddy

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Master of Science Degree in Electrical Engineering

The University of Toledo
December 2011

The impact of technology on education in both traditional classes and Distance Learning (DL) is enormous. The Internet and other network technologies have given the students, different avenues through which a formal education can be obtained [1].

This research proposes a low cost interactive Distance Learning system. The system includes a whiteboard, where users can draw, write and erase. The application is based on the client-server model and it supports the broadcasting of both text messaging and mouse-based drawings. It was developed using the Java programming language and it supports a variable number of different virtual classrooms through the use of dynamic servers.
For my parents, Raveendranath Reddy and Shobha Reddy, brother, Rajeev Reddy, and all my friends
Acknowledgements

I sincerely thank my advisor Dr. Jackson Carvalho for proposing me the problem this dissertation addresses, for his guidance and mentorship. It has been a great learning experience working with him. The financial support from the EECS Department in the form of a graduate assistantship is gratefully acknowledged. Special thanks to all my friends for their enthusiastic and generous support throughout my research study. Above all, I thank my parents, Raveendranath Reddy and Shobha Reddy, and my brother, Rajeev Reddy, who have been a constant source of encouragement and support for me throughout my journey in the Master’s Degree program.
Contents

Abstract iii
Acknowledgements v
Contents vi
List of Tables ix
List of Figures x
List of Abbreviations xi
1 Introduction 1

1.1 Problem Statement ................................................................. 1
1.2 Research Approach .................................................................. 3
1.3 Organization ............................................................................ 3

2 Literature Review 5

2.1 Distance Learning History ....................................................... 5

2.1.1 Correspondence Study .......................................................... 6
2.1.2 Telecommunications ........................................................... 9
2.1.3 The Networking of Computers ............................................. 11

2.2 Study of Interaction in Web-based Instruction .......................... 13
2.2.1 Web-based Interaction ................................................................. 13
2.2.2 The Interaction Design ............................................................... 14

2.3 Distance Learning System at the University of Toledo and Lorain Community County College ................................................................. 16

3 Client/Server Model VS Peer to Peer Model ................................. 20

3.1 Client/Server Model ........................................................................ 20
3.1.1 Architecture ............................................................................... 21

3.2 Peer to Peer Model ........................................................................ 25
3.2.1 Architecture ............................................................................... 26

3.3 Comparison of the models ............................................................. 29
3.3.1 Ease of Development ................................................................. 29
3.3.2 Scalability .................................................................................. 30
3.3.3 Reliability .................................................................................. 31
3.3.4 Manageability ............................................................................ 31
3.3.5 Security ...................................................................................... 32

4 Implementation Characteristics ....................................................... 33

4.1 Database Layer ............................................................................... 38
4.2 Application Layer ........................................................................... 41
4.2.1 Broadcast Messenger ............................................................... 42
4.2.2 Whiteboard ............................................................................... 44

4.3 GUI Layer ...................................................................................... 46
4.3.1 Login Screen .............................................................................. 46
4.3.2 User Registration Screen ................................................................. 47
4.3.3 Messaging Screen ............................................................................. 48
4.3.4 Whiteboard Screen ........................................................................ 50

4.4 Example Scenario ............................................................................... 51

5 Conclusions and Future Work ................................................................. 58

5.1 Conclusion .......................................................................................... 58
5.3 Future Work ......................................................................................... 59

References .............................................................................................. 60

A Source Code ........................................................................................ 65

A.1 GUI Layer .......................................................................................... 65

A.1.1 Login Screen .................................................................................... 65
A.1.2 New User Registration Screen ......................................................... 67
A.1.3 Text Messaging Screen .................................................................... 69
A.1.4 Whiteboard Screen .......................................................................... 70

A.2 Network Communication ..................................................................... 72

A.2.1 Broadcasting Text and Whiteboard Data ......................................... 72
A.2.1 Receiving Text Data ......................................................................... 72
A.2.2 Receiving Whiteboard Data ............................................................... 73
List of Tables

4-1 Students........................................................................................................................................40
4-2 Professors .......................................................................................................................................40
4-3 Courses........................................................................................................................................41
List of Figures

3-1 Two-tier Client/Server Architecture. .................................................................22
3-2 Three-tier Client/Server Architecture. ..............................................................24
3-3 Peer-to-peer Architecture....................................................................................27
4-1 Communication flow between Virtual Classroom and Database. ......................34
4-2 Communication between Clients, Dynamic Server and Database Server. .........35
4-3 Architecture of Client/Server communication model.........................................37
4-4 System Architecture ............................................................................................38
4-5 User Login Screen Shot......................................................................................47
4-6 User Registration Form.........................................................................................48
4-7 Text Discussion Board..........................................................................................49
4-8 Whiteboard Screen...............................................................................................50
4-9 Text-based messaging screen of a Student while the Instructor is teaching........52
4-10 Whiteboard of a Student while the Instructor is teaching....................................52
4-11 Text screen of a Student at mid-session............................................................53
4-12 Whiteboard screen of a Student at mid-session................................................54
4-13 Screen shot of the Student’s text-based messaging..........................................55
4-14 Screen shot of the Student’s whiteboard screen while erasing..........................56
4-15: Screen shot of the Instructor’s after the demonstration by a student.................57
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AESP</td>
<td>Appalachian Education Satellite Project</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ARPAnet</td>
<td>Advanced Research Projects Agency network</td>
</tr>
<tr>
<td>CGI</td>
<td>Common Gateway Interface</td>
</tr>
<tr>
<td>CMS</td>
<td>Course Management System</td>
</tr>
<tr>
<td>DE</td>
<td>Distance Education</td>
</tr>
<tr>
<td>DHTML</td>
<td>Dynamic Hyper Text Markup Language</td>
</tr>
<tr>
<td>DL</td>
<td>Distance Learning</td>
</tr>
<tr>
<td>ETFA</td>
<td>Educational Television Facilities Act</td>
</tr>
<tr>
<td>FCC</td>
<td>Federal Communications Commission</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HTML</td>
<td>Hyper Text Markup language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hyper Text Transfer Protocol</td>
</tr>
<tr>
<td>ICS</td>
<td>International Correspondence School</td>
</tr>
<tr>
<td>ID</td>
<td>Identification</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>ITFS</td>
<td>Instructional Television Fixed Service</td>
</tr>
<tr>
<td>ITU</td>
<td>International Telecommunications Union</td>
</tr>
<tr>
<td>IVDL</td>
<td>Interactive Video Distance Learning</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LCCC</td>
<td>Lorain County Community College</td>
</tr>
<tr>
<td>MB</td>
<td>Mega Bytes</td>
</tr>
<tr>
<td>NSF</td>
<td>National Science Foundation</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>PCs</td>
<td>Personal Computers</td>
</tr>
<tr>
<td>PERL</td>
<td>Practical Extraction and Report Language</td>
</tr>
<tr>
<td>PHP</td>
<td>Hypertext Processor</td>
</tr>
<tr>
<td>QOS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RAS</td>
<td>Registration, Admission and Status</td>
</tr>
<tr>
<td>RTCP</td>
<td>RTP Control Protocol</td>
</tr>
<tr>
<td>RTP</td>
<td>Real-time Transport Protocol</td>
</tr>
<tr>
<td>SNMP</td>
<td>Simple Network Management Protocol</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
</tr>
<tr>
<td>STIN</td>
<td>Stanford Instructional Television Network</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>Transmission Control Protocol/Internet Protocol</td>
</tr>
<tr>
<td>TV</td>
<td>Television</td>
</tr>
<tr>
<td>USA</td>
<td>United States of America</td>
</tr>
<tr>
<td>VRML</td>
<td>Virtual Reality Modeling Language</td>
</tr>
<tr>
<td>WAN</td>
<td>Wide Area Network</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
1.1 **Problem Statement**

Distance education is institutional-based, formal education system where the group of learners and instructors are separated, and where interactive communication systems are used to connect the learners, resources, and the instructors [2]. The Distance Learning (DL) application can be examined for its performance only by the technology used for its carriage system. The carriage system is responsible for sharing any data among the users using the application.

The DL system can be classified into three basic types based on their carriage systems [3]:

- Correspondence;
- Broadcast; and
- Interactive.

The correspondence approach is any system where there is a considerable time delay between the instructor distribution and student receipt of any data and vice versa. Communicating with electronic mail and directly communicating one on one are kind of correspondence approach. The broadcasting approach takes the advantage of mass communication media. This approach allows
many students to use the application at the same time. Therefore, when an instructor shares the data, all the students receive it at the same time. In the interactive approach, the information is shared between the users dynamically. In other words, the information is sent and received instantly. Online conferencing on a computer is an example of this approach. Considering technological factors, such as compatibility, availability and maintainability, most of the DL applications available are built combining two carriage systems.

The projected global demand in United States of America for higher education by year 2025 will be 16 million students and the student enrollment in distance education increased from 5% in 1998 to 15% in 2002 [4]. According to 2010 Distance Education Survey [5], the Sloan Consortium reported “a 21 percent growth in distance learning enrollments when comparing the number of students who took at least one online course during the fall 2009 term to the number of students who participated in an online course during the previous year.” The use of DL optimizes instructional resources such as the number of classrooms and instructors needed. However the investment in technology is large. A standard DL system such as the one currently in use in the University of Toledo and Lorain County Community College for example, requires the use of ¼ of a million dollars router for its operation. This system allows the instructor the possibility of using a whiteboard to share drawings with the students. However such system provides the students with no means of interaction among themselves as well as with the instructor.

Enhancing the DL form of instruction with the introduction of a tool to change the passive role of the students to active learners would improve the quality of instruction.
1.2 Research Approach

The goal of this research is to improve the communication between the students and the instructor involved in distance learning. It addresses the use of an interactive whiteboard to facilitate distance education in a way that users can establish a rich communication by sharing real-time drawings and text. The application has been designed to be used on any kind of personal computers which has access to the Internet. The interaction is available through standard keyboard, pointing device, such as mouse, and/or by touch, provided the display device is touch screen enabled. The application includes the following features:

- Capability to register/delete students;
- A static server to support database operations and a dynamic server to support communication and interaction;
- A real-time text based broadcasting system; and
- A whiteboard, on which a user can illustrate real-time drawing, writing or erasing.

I claim that:

1. A distance learning environment can be improved by facilitating the communication among the students as well as the instructor, and
2. An interactive low cost system can be provided to support claim 1.

1.3 Organization

The remainder of this document is organized as follows. Chapter 2 discusses the literature review. Chapter 3 addresses two basic communication models, Client/Server and Peer-to-Peer. At the end of the Chapter 3, these communication models are compared. Chapter 4 consists of the
implementation of the DL application this research proposes and the screenshots of the application.

Chapter 5 provides the conclusion of the work done and future work.
Chapter 2

Literature Review

This chapter is structured into three parts. The first includes a review of literature regarding the history of distance education system. It addresses correspondence study, telecommunications, computers and networking. The notion of interaction in web-based distance education is explained in the second part which reviews different approaches for defining and studying interaction and the design of communication. The third section discusses the distance learning system used at University of Toledo and Lorain Community County College.

2.1 Distance Learning History

In order to understand the perspective of interaction, a literature analysis on the distance learning system is necessary. This section reviews the essential breakthroughs in the distance learning system which includes: correspondence study, telecommunications via televisions (TV), and computer-based interaction with the use of internet and World Wide Web (WWW).

The history of distance learning system is vital for being aware of the design of the distance learning system, but for the most part, it is ignored [6]. The regulation of the distance learning
system deals realistically based on the practical rather than theoretical considerations, but not really on past [7].

2.1.1 Correspondence Study

The distance learning system has evolved from in-person communication to on paper broadcasting using printing press, and subsequently, electronic media based [8]. A correspondence study program had started for Great Britain by Isaac Pitman in 1840. This program delivered letters for a penny where students learned shorthand; hence this program was called “Penny Post”. At this period, the formalized correspondence study program became popular in Berlin, Germany [2]. The recognized university courses started at Oxford University in 1870 [9], it was followed by Skerry’s College at Edinburgh in 1878 and the University Correspondence College in London in 1887 [8]. Hermod started a correspondence program in Sweden on English course in 1886, which was one of the largest distance learning institutions then [8].

The correspondence education system was introduced to the United States of America (USA) by Anna Eliot Ticknor in 1873. Anna Eliot Ticknor was the founder of The Society to Encourage Studies at Home, an organization founded in Boston, Massachusetts, included assigned readings, tests and consultation with a teacher [2]. The New York State started sanctioning academic degrees for correspondence study programs, first delivered by the Chautauqua College of Liberal Arts in 1883. As correspondence delivery was used along with necessary attendance, the degrees contributed were among the first hybrid degrees. William Rainey Harper, a Yale professor, was the creator and leader of the Chautauqua correspondence program [8] [10].

The correspondence programs were not only designed for wealthy individuals in the society. One of the largest correspondence institutions in the USA was started in Pennsylvania. International Correspondence School (ICS) was founded in 1891 by the publisher Thomas J. Foster. 225,000
students were enrolled in ICS by 1900 and by 1920, ICS had an enrollment of more than two million students [10]. Today, ICS is a large publishing empire known as Thompson Publishing [6]. The University of Chicago was founded a year before Foster’s ICS started, creating the first university extension program for an American university. This division was responsible for lecture study, class study, corresponding program, library, and training [10]. Harper moved to Chicago from his home, New York, to head the Chicago extension division [8]. The Chicago’s extension program was halted for seven years starting from 1899, even though it had 125 instructors instructing over 3,000 students [10]. During this period, the University of Wisconsin offered extension services for individuals by establishing correspondence programs in use today [10]. Harper believed that the correspondence was a primary method of instructional delivery for the future throughout his career form Chautauqua to Chicago.

If correspondence is defined to include today’s projects, term papers, email, electronic presentations, discussion boards, and portfolios, then Harper may have acceptably predicted the future of the distance learning system. However, possibly, the characteristics of the learner played an important role in determining the position of Harper. An investigation of the characteristics of a learner in association to distance education is important and should be considered, but it is not a main focal point of this study.

A research should be conducted to uncover any promising influences or associations between the technology, the field, and the boundaries of access to learning resources. The acquirement of the resources in relationship to distance education populations has an influence on interaction within any distance learning system, regardless of the distance. Although, the learners today are more heterogeneous than their correspondence school predecessors, homogeneity still rests in student, technical, social, economic, and cognitive situations for taking part in higher education.
There are mixed reviews by the historians on distance learning system. Joseph Kett, a historian, believes that higher education in institutions beleaguer accommodated instructional delivery by correspondence for revenue [11]. However, Charles Wedemeyer, another historian, believes that the use of correspondence system was a knowledgeable act by institutions to shift from ubiquitous snobbery found in the traditional classroom [12]. Both perspectives are found to have evidence in varying institutional operations like the contextualization of the distance education system itself. The Chautauqua program was bound to wealthy adults whereas ICS was started to educate working class miners. The extension efforts were not a central part in the university, but were operated as a small business within the university [11].

Historically, institutions offering higher education continued to advance and enhance the traditional classroom system as the primary education system. This disagreement resulted in formulating perception that the distance education was not as precise, resulting in the bordering of correspondence as the instructional delivery medium. Correspondence programs were generally closed or folded into continuing education. Correspondence and attempts at using the delivery method for every lesson without a focus on education depreciated the learning outcome [6]. As the technology improved, the significance of correspondence system eroded. The educators eliminated correspondence system from higher education by early 1960s. The terminology was modified to define correspondence study as “independent study” to differentiate it from traditional classroom [7].

Correspondence system continued to provide revenue in telecommunication activities in continuing education [7]. Training through correspondence system and telecommunications provided noteworthy contributions to progress in the USA. Health care professionals learned about new illness and their treatment, workers learned about new procedures for safety and famers learned about agriculture production practices.
2.1.2 Telecommunications

Broadcast radio became the instructional media and delivery tool preferred by educators in distance education system during 1920s and 1930s [8]. Instructions could be delivered to large number of people. However, the educators made serious errors in learning theory to obtain the materials for the new medium.

Although, the learners continued to take courses by broadcast radio, the attempts to change the education system was slow [10]. The educators sent information and materials by mail to patch up the design flaw.

Changing the delivery medium without revising methods or contents can be delicate or obvious. A clear change is noticed when the instruction asks the student to “refer to a figure” on a radio broadcast, because the students cannot see the figure on the radio. However, slight differences exist when training across media. The education system should drive media selection and use, but not the technological tools [13]. Bringing radio broadcast instruction to higher education; educators began to realize the need for instruction that was systematically designed, through the conversion of text-based content to aural.

The radio broadcasting became popular in the USA. The educational radio expanded to more than 176 stations located on higher education campuses [8]. Even though, the radio broadcasting continued to be used, the use of the medium for training was replaced by TVs in 1904s and 1950s [8].

The University of Iowa was the first institute to start the distance education through TV in 1934. More than 400 educational programs were on air by 1939 [6]. The educational TV began to get popular after the World War II, after 242 broadcasting frequencies were allocated exclusively for distance education system. The capability of the telecommunication medium to reach large society created systems to sell advertisements. Because it did not attract large number of audience,
educational TV diminished as a major focus. The Ford Foundation spent millions of dollars in grants to maintain the distance education system through TVs in 1950 [6]. A year later Western Reserve University started first continuous series of credit courses using this educational system [10].

The USA promoted this educational system with government legal and financial support in early 1960s. In 1962, the Educational Television Facilities Act (ETFA) raised public funds to promote educational shows on TV [10]. Two years later distance education was brought worldwide. In 1969, Stanford University established the Stanford Instructional Television Network (SITN). It was basically designed to broadcast 120 engineering courses to 900 engineers at 16 participating companies [14]. This delivery tool faded out by 1984 partially due to the limitation of small broadcasting radius of Instructional Television Fixed Service (ITFS).

The satellite communications as a medium of delivery means was much better and is in use today, unlike ITFS. The Appalachian Education Satellite Project (AESP) was established in 1974 to provide basic education and college courses for adults in Appalachian. This organization later offered graduate level courses to rural teachers [10].

Satellite transmissions were used, along with other media, through the latter half of the century. By 1972, the Federal Communications Commission (FCC) made mandatory for all the cable operators to provide an educational channel. The satellite transmissions are typically one-way, synchronous transmission. Educators using this technology were limited in their engagement because they had to wait for delayed response from the learners. This limitation was eliminated by the invention of the interactive TV.

Closed-circuit systems, microwave, broadcast, satellite, cable television and interactive television altered, supplementing the traditional classroom. This hybridization occurred when it was required to explain the complex subject and to help learner motivation [13]. The new tool, the personal computer, came into existence as a focus on delivering system in late 1970s and 1980s.
2.1.3 The Networking of Computers

The innovations allowing computers to communicate through a network radically changed the way in which distance education is delivered. The computers began joining discrete media into a single platform in 1980s [13]. Audio cassette tapes that were historically separated from text were merged into an audio/data CD-ROM. The videos once recorded for the use by educational TV are now being streamed or delivered as a DVD. The computers were used for distance courses throughout the 1980s [15].

Local Area Network (LAN) and Wide Area Network (WAN) were developed once the networking technologies enabled computers to communicate with each other. Both LAN and WAN are closed systems. But LAN is used to connect within the building premises, whereas WAN was used to connect many buildings worldwide. The organization connected their computers using internet as globalization increased [10].

The focus on connection speed led the National Science Foundation (NSF) to fund projects providing high-speed connection among universities. The communication system uses the knowledge acquired from the Advanced Research Projects Agency network (ARPAnet) system [16] and provided access for universities that are not connected to ARPAnet. The NSF System, known as NSFNet, connected universities with five supercomputer centers, enabling users to interact using exchange documents, emails, communicating on bulletin boards and using library services [6]. This network was the first to allow users to communicate from home [16]. The use of the Transmission Control Protocol/Internet Protocol (TCP/IP) to communicate among machines on ARPAnet and the development of Domain Name Service (DNS) at the University of Wisconsin were the two events that shaped today’s internet.

Web languages needed to be created in order to use WWW, and an application that allows the user to read the Web pages. The most familiar protocol used for communications on Web is
Hypertext Transfer Protocol (HTTP) [10]. This protocol allows the computers to send and receive packets of data over internet and the most familiar language, Hyper Text Markup Language (HTML) displays the information on the Web browser. HTML is not the only language used to develop Web pages. Dynamic Hyper Text Markup Language (DHTML), Extensible Markup Language (XML), Hypertext Processor (PHP), Virtual Reality Modeling Language (VRML) and Common Gateway Interface (CGI) for dynamic Practical Extraction and Report Language (PERL) documents emerged soon after HTML. The first browser was launched by Mosaic in 1993 [16]. The first search engine, called Gopher, was developed by the University of Minnesota. Soon later sophisticated search engines were developed [16].

The technology advancement in creation of WWW, DNS, search engines and software tools to create Web pages fascinated faculty to develop online course materials [10]. Although, instructing with global access supported ease of access for both students and instructors, it involved risk of violating fair use.

The students still had to switch between different software tools for using various related applications (for example websites, chat, email etc.). This switching between the windows frustrated the users. So, all these applications were merged into a single system called Course Management System (CMS) [13].

Two key CMS products used in distance education, blackboard and WebCT, were merged, leaving a small market share for alternative applications. An open source CMS product, called Sakai, was developed by collaborating institutions with the primary goal for transporting objects between these institutions. There are more than 100 educational institutions as members throughout the world today.

In the past decade, there is an exponential growth of distance education system. There were 3,077,000 students enrolled in distance education courses at two and four year accredited institutions
by year 2001 [17]. In the academic year 2006-2007, two and four year accredited institutions reported an estimated 12.2 million enrollments in distance education courses [18]. The internet has created a new level of competition with an entry of for-profit universities, offering online education, competing with the universities offering traditional education [19].

Distance education is influenced by the application of new technologies for teaching postsecondary coursework, by increased competition, and by a shifting population of students with experience using the Web. This shift is “a reflection of changing educational values and philosophies” [19].

2.2 Study of Interaction in Web-based Instruction

This section addresses how web-based interactions are interpreted exploring important studies that contributed to our knowledge of traditional classrooms.

2.2.1 Web-based Interaction

Interaction based on web, opens a range of dynamic possibilities for interacting among users [13]. Distance education via web inclined the transfer of instructional models to using instructional design models that sustain a situation with affluent communication. These design models needed to be aware of and responsive to any interaction, in order make it a learner-centered instruction. Therefore, makes the web-based interaction a multi-dimensional model. In other words, makes more interactive by introducing two-way media, instead of one-way media. Moore developed three interaction models: learner-to-content, leaner-to-instructor and learner-to-learner [20]. Later Hillman and others added dynamic models between learner and interface [21].

The interaction model, learner-to-content, examines the interaction between the learner and the content. The learner’s behavior, constructivism and knowledge are considered when interacting with
the instructional content. The learner-to-instructor section includes all the interactions between the learners and the instructors. Similarly, learner-to-learner section includes all interactions among the learners in a classroom, not considering if the classroom is physical or virtual.

In another study, Moller divided interactions into three segments. The first segment of academic community was consisted of learners and instructors. In this segment, Moller measured peer interaction and collaboration to be the intellectual community. The third segment, the interpersonal community involved inter-personal encouragement or assistance [22].

With the consideration of the work done by the earlier researchers, Jung examined the interaction using academic, collaborative and social interactions [23]. Academic interaction is content-centered which consists of online materials and a feedback, task-orientated, is provided. The course of action where learners discuss issues related to learning and collaborate with others is called collaborative interaction, according to Jung. According to Jung, social interaction consists of social feedback, encouragement and motivation from others [23]. These studies indicate that the interaction should be considered for designing the instruction.

2.2.2 The Interaction Design

The proof towards a foundation for continuing research in this area is provided by the literature regarding the design of interaction. The study of interaction in a traditional classroom has contributed to the understanding of the organization of classroom discussion [24] [25]. For understanding the online discussions, development of topic is particularly useful.

There are two main studies that contributed to the understanding of the discussions in classrooms. They have significant implications for technology-based discussions. These two studies explored the student’s perceptions of discourse creation [26]; patterns of task setting sequences [27]; and the structure of flow of topic conversations [25].
Student perception study, by Stokoe, found that “off topic” discourse was dominant and is important in the communication organization. It is usual for instructors to engage learners in “off topic” discourse conversation in a traditional classroom. Stokoe’s research proved that student’s perception is significant in contributing to the overall learning experience.

Benwell and Stokoe explored task setting sequences of students, in 2002, and found a three-part sequence of control, instructors typically use in defining the parameters of classroom discussion [27]. At first, the researchers found that the instructors define the task related to the discourse. Secondly, they can decide the time allocation to the line of discourse. Thirdly, the researchers found that the instructor has the ability to re-direct the discourse back to the task or vice versa.

The design of instruction is a significant factor for an adequate learning outcome. The quality of the interaction is very much dependent upon the control of the instructor. Students will be more likely to solve the course objectives, if the instructor uses effective instructional design practices to develop a course.

Interaction remains a fundamental factor in meeting the learning objectives and their achievement or breakdown, throughout the history of education. The literature suggests the practice of designing instruction based on research was necessary before the arrival of instructional design.

Educators using the “cultural reproduction” [28] are no longer assisted by analog technological devices. In fact, they are passing along a culture that is largely digitalized. The digitalization of information created and will continue to create, a quality of virtual communities. These communities will demand new knowledge in order to analyze, access and socialize online interaction regardless of the tool that may be presented.
2.3 Distance Learning System at the University of Toledo and Lorain Community County College

As part of a partnership agreement between The University of Toledo and Lorain County Community College (LCCC) some courses offered by the electrical engineering and computer science department at The University of Toledo may be attended by students through Distance Learning. To support this a number of classrooms are equipped with terminals/computers which can handle Interactive Video Distance Learning (IVDL) connections. Each one of these rooms is connected to the central control room through a fiber wire and to a video router, at their respective campuses. The video router is used for dynamic setup where the data can be routed to any of those rooms to any of the codecs used by the University of Toledo. The video router also gives up plenty of flexibility and redundancy in case something stops working in a room or a codec. A codec is an encoding/decoding device or algorithm that controls how data are compressed during encoding, and decompressed during decoding [29].

The communication in this application is made by making several exclusive IP calls by using the H.323 protocol. The H.323 protocol establishes a peer-to-peer connection. This protocol standard offers groundwork for audio, video and data communications over the IP – based networks, which includes internet. This protocol is an umbrella recommendation by the International Telecommunications Union (ITU) and do not provide a guaranteed Quality of Service (QOS). The networks implementing the H.323 protocol dominate corporate desktops and consist of packet-switched TCP/IP and Internetwork Packet Exchange (IPX) over Ethernet and Token Ring network technologies [30] [31]. The H.323 protocol paradigms are important building blocks for LAN – based applications for multimedia communications. This includes several protocols like H.245 Real-
time Transport Protocol/RTP Control Protocol (RTP/RTCP), Q.931, H.225.0 – Registration,
Admission and Status (RAS) and also audio/video codecs that compresses and decompresses data streams. The media streams are transmitted on RTP/RTCP, where RTP transmits the actual data and RTCP transmits status and control information.

# 2.3.1 Equipment used for the Application

The University of Toledo has installed equipment for the distance learning application.

The equipment installed in the control room are:

- **Tandberg C60 codec:** This is a hardware/device codec that takes the data signal (audio/video) encodes/compresses it and sends to another codec where it decodes/decompresses the signal. The cost of each codec is $21900.00 [32].

- **Polycom MGC/MGC+ Unified Conferencing Bridges:** The Tandberg codec does support multipoint calls (calls with more than three terminals at a time), provided the license is purchased, which are expensive. The bridge is the cheaper alternative solution for it. It serves in negotiating with different protocols as well [33].

- **Autopatch Video Router:** The video router allows tie up all the classroom terminals to all the codecs installed, and allows manipulating of audio as well. The University of Toledo currently uses the 64 input/128 output Autopatch router. The manufacturer’s suggested retail price of each video router is $2015.00 [34].

Once the data signal reaches the control room, it routes through a variety of equipment, amplifiers, bi-amps, audio mixers and video switcher. Amplifiers, bi-amps and audio mixers are used to transmit clean audio signals in and out of the classroom. The video switcher routes the video signal to the terminals in the classroom or to any of the codecs installed. The terminals may be projectors, monitors, etc.
The classrooms are usually controlled by the AMX controllers and the AMX touch panels. The soft code is written and loaded into these controllers to control the devices/terminals in the classroom. The interface is created and displayed on a touch panel which is then transmitted to the control room for manipulation.

A variety of equipment installed in the classroom are:

- **Amplifiers**: The audio amplifiers are used to amplify the audio signal so that the signal is not lost.

- **Audio Mixers**: The audio mixer combines and processes a number of audio signals. These audio mixers records and broadcasts the audio signal.

- **Video Switcher**: The video switcher is responsible to re-route the video signal to the respected destination device. The video switcher used by the University of Toledo is Extron Electronics MTPX Plus 816. This video switcher in priced at $8926.20 each [35].

- **AMX Controller**: This controller integrates and controls wide variety of devices. In our case, projectors, monitors and touch panels. The controller installed by The University of Toledo is NI-3100 NetLinx Integrated Controller. This is a high-performance controller with 64MB of onboard Random Access Memory (RAM). This AMX Controller is priced at $2420.00 [36].

- **AMX Touch Panels**: These touch panels used for displaying the interface. The University of Toledo uses NXT CV10 touch panels, which supports composite/S-video multimedia and one-touch control options and accommodate graphics, icons and video windows [37]. Each of these AMX NXT CV10 touch panels are priced at $5700.00 [38].

- **Projectors**: The University of Toledo uses a variety of projectors in the classroom for projecting the video received from the control room. The Epson 1800 series projectors are mostly used. The Epson Powerlite 450 short-throw projectors are used with smart
boards. The Epson Powerlite 1880 multimedia projector is priced at $1399.00 [39] and Epson Powerlite 450 multimedia projector is priced at $1599.00 [40].

- **Microphones:** In the classroom, the students are provided with Shure Microflex MX392 microphones, which has a price tag of $214.00 [41] and the instructors are provided with an infrared microphone, Lightspeed CAT 800.

- **Camera:** The high definition cameras are installed for video recording and conferencing.

- **Smart board:** The classrooms are installed with the Smart Board 885ix interactive whiteboard system. This whiteboard has several tools which the user can select, like pen, eraser etc. This whiteboard allows two users to use the same whiteboard at the same time. With this multi-touch functionality, it makes easy for the instructor to demonstrate in a interactive way. It comes with a projector which creates virtually no shadow or glare, unlike the projectors that are mounted on the ceiling or on the walls. So the students in the classroom have clear view. The projector is network enabled, so it can send and receive commands to and from web server or Simple Network Management Protocol (SNMP) client. These smart boards are priced at $5899.00 each [42].

Therefore, for setting up a control room, the LCCC invests around $25000.00 to $15000.00 for each classroom. For maintaining the control room and classroom, there are three staff members working every day. Hence, the cost incurred for maintenance of the distance learning system is between $25000.00 and $40000.00 per annum.
Chapter 3

Client/Server Model VS Peer to Peer Model

There are two basic models of communications namely, client/server model and peer to peer model. The following sections address the fundamental characteristics of such models.

3.1 Client/Server Model

Client/Server model is a distributed communication model in which the client applications send requests for services to the server application. These client and server applications often run on different machines interconnected by a computer network. Application that uses the internet for information retrieval from the WWW may be seen as an example of such communication model. However, this model can be usually applied to the systems in which an institution runs applications with multiple tasks distributed among different computers in a network.

A client application is a process or program that sends messages to a server via the network [43]. These messages are requests to the server to perform tasks, such as retrieving data from a database or returning a piece of information on the server’s disk space.
The server application is a process or program that listens for client requests that are transmitted via the network [43]. Servers receive client requests and carry out actions, like database queries or reading files from the disk.

A client/server communication environment may use a variety of operating systems and hardware from multiple vendors and standard network protocols like Transmission Control Protocol/Internet Protocol (TCP/IP) provide compatibility. Vendor independence and freedom of choice are further advantages of the model. For example, inexpensive PC equipment can be interconnected with mainframe servers.

Client/server systems can be scaled up in size more readily than centralized solutions since server functions can be distributed across more and more server computers as the number of clients increase. Server processes can thus run in parallel with each process serving its own set of clients. However, when there are multiple servers that update information, there must be some coordination mechanism to avoid inconsistencies.

The drawbacks of the client/server model are that security is more difficult to ensure in a distributed environment than it is in a centralized one, that the administration of distributed equipment can be much more expensive than the maintenance of a centralized system, that data distributed across servers needs to be kept consistent, and that the failure of one server can render a large client/server system unavailable. If a server fails, none of its clients can make further progress; unless the system is designed to be fault-tolerant.

3.1.1 Architecture

The client/server communication model has two types of architecture, namely, two-tier architecture and three-tier architecture. The developers will have to make a decision on which architecture is suitable for the application at the start of any client/server application. In the
client/server model, the same machine can act as both client and server, but generally they both run on different machines. The types of architecture discussed in this section focus on creating the distributed applications, i.e. the client and server run on different machines.

### 3.1.1.1 Two-tier Architecture

In the implementation of two-tier client/server architecture, the three components of an application (presentation, processing and data) are divided into two software packages: client application and server application. These software packages or entities are referred to as tiers [44]. The figure 3.1 shows the two-tier client/server communication model.

![Figure 3-1: Two-tier Client/server Architecture.](image)

The GUI for presentation is handled by the client for client side. Usually the server need not have GUI, but in case the server needs to have GUI, then that part is exclusively handled by the server application. The processing is split between the client and server. The client makes a request and the server processes it. The data is stored on and accessed through the server application only. The client takes the responsibility on application logic whereas the server application handles intensive tasks. For client to make a service request on server, the client must know the syntax of the
server or should transmit via Application Program Interface (API) by creating a tight link between the two layers. The client must also know the location of the server (IP address) for creating a link and the outline of the data organized to make requests on data. The client application can take the advantage of the logic written in server application to centralize basic tasks such as data security and validation.

One of the advantages of using two-tier architecture is the time, in other words the time taken for developing a two-tier architecture application is very less. A single developer can create a database, set it up on a remote server machine, develop a GUI and create a client along with its application logic.

The two-tier environments support a range of data structures including the in-built functionalities. The two-tier environments work well in relatively homogeneous environments with fixed business rules. This architecture is not suited for dispersed, heterogeneous environments where the rules always keep changing.

Because the application logic resides on client machine, the two-tier model faces the application re-distribution problems. If there is a change in any rule of the business, then the application logic in each client have to be re-written and re-distributed which is quite a difficult process.

The security in the two-tier architecture is complicated. The developers usually develop a client/server application without the middleware technologies which increases the security. They provide the end users with the passwords to access the database. Using this type of security, the database can get corrupted.

3.1.1.2 Three-Tier Architecture

In the implementation of three-tier client/server architecture, the three components of an application (presentation, processing and data) are divided into three separate software packages.
This helps to overcome some of the issues in two-tier architecture. This architecture can have more than one server. The figure 3.2 shows the three-tier architecture pictorially.

![Three-tier Client/server Architecture](image)

Figure 3-2: Three-tier Client/server Architecture.

Unlike two-tier architecture, each tier in three-tier architecture has a dedicated machine. Whenever the manipulations or access on data are required by the client, a request is made to the functionality server, which is the middle tier. When the functionality server listens to such requests, it can process the request or can forward the request to another functionality server. For its best performance, the middle tier should be written in highly portable programming language like Java. Using the multi-threading feature in Java, the functionality servers can be accessed by multiple clients.

One advantage of three-tier architecture over the two-tier architecture is that the client in three-tier architecture need not know the syntax or the organization of the database to access it. The three-tier clients can only send the data as parameters and specify the data structure to accept returned values. Because the client need not know the organization of the database, the data can be
organized in any format. Therefore, even if the business rules change or database is modified, the client’s application logic does not change. Hence, there is no need of re-distributing.

Besides the above stated advantage of three-tier over the two-tier architecture, there are many more advantages. This model allows parallel development of tiers because this model has different software entity for each tier. Therefore, developing each tier by a specialist improves the overall quality of the final application. The network traffic can considerably be reduced by stripping data by functionality servers before it is distributed to clients in LAN. The basic logic in the middle tier can be reused for other applications. The migration costs when modifying or changing client applications, maintenance of work load and development efforts can be minimized by using the reusable code.

As there are three software packages in the three-tier model, the logic to be written is more. The three-tier application needs increased network traffic management, server load maintenance and balancing and better fault tolerance systems. These are the main disadvantages of the three-tier client/server architecture.

3.2 Peer to Peer Model

The peer to peer model is a distributed communication model in which every machine on the network acts as both client and server. Therefore, every system on the network will contain the same application logic. There is no concept of centralized server in this model. In this model, any machine on the network behaves in the same manner. In other words, any machine can make a service request and process the incoming requests. A telephone call to person can be considered a form of peer to peer communication model.
Napster can be considered as one of the most known example of peer to peer models. Napster’s music file sharing application consisted of a distributed network of several thousands of PCs.

Peer to peer applications allow computers to communicate directly with each other without having to go through intermediate channels. This makes communication fast and effective. For example, Napster gave the user the ways to access the other person’s hard drive instead of storing the music files in its central server.

Security is the major concern for the developers looking forward to develop an application based on the peer to peer model. Because the peer to peer model allows users to directly access the other’s hard drives, their sensitive data may be exposed. In order to secure the sensitive data, the organizations would have to install hierarchy of custom access controls. This will increase the maintenance work load on the system managers and administrators. Another obvious security concern is to defend against the sloppy users uploading malicious viruses directly onto the machines on the network.

### 3.2.1 Architecture

As mentioned in the previous section, there is no central server in the peer to peer applications. All the computers in the network act as both client and server and are connected directly to each other. Figure 3.3 demonstrates the basic peer to peer architecture.
As seen in the figure 3.3, all the peers (computers) in the network are directly connected to each other in peer to peer communication model. Each peer will have the same software package implemented in them. Considering the complexity associated with discovering, communicating and managing the large number of systems in a distributed network, the software package is typically structured in a layered manner. The software entity of most of the peer to peer applications can be divided into three layers [45]:

- the base overlay layer;
- the middleware layer; and
- the application layer.

The issues with recognizing other machines in the peer to peer system are dealt by the base overlay layer. It is also responsible for creating a communication channel between all the machines to communicate with each other. This layer must ensure that all machines be aware of the other machines.
The middleware layer consists of further more software components that could usually be reused by many applications. The software entity in this layer is typically invoked by other entities and can be used as a basic supporting infrastructure to develop other applications. The methods in this layer have ability to create a distributed index providing publish-subscribe facility and the security services. The methods in this layer are not necessary for all applications, but these methods are designed to be reused by other applications.

The application layer consists of software entities, which are intended to be used by the end users. This layer is designed to make use of the distributed nature of peer to peer architecture.

3.2.1.1 The Base Overlay layer

This layer is very important in peer to peer applications. This feature should be distributed to all peer to peer systems. The software entity included in this layer consists of the following qualities:

- Recognition: Before the system starts to communicate, it should first recognize all other systems in the network. It should also make itself visible to all the machines.
- Overlay Formation: This creates a communication links by which all the nodes are connected.
- Application-Level Multicast: This is the main functionality of the application. It is responsible to send and receive the service requests.

3.2.1.2 The Middleware Layer

The middleware layer provides some common, reusable, methods that can be used in any other applications. Some of the functionalities offered in this layer are:

- Security: This layer manages secure communication over the network by extending the support for encryption, authentication and access control. Note, the security problems are hard to manage.
• Registration: The node will have to register into the network system before any requests are made.

• Publish-subscribe System: This feature enables the peer to peer application to share the information in a controlled manner. The publishers send the information to publish-subscribe system. The Subscribers, informs the system about the type of information they wish to receive. Only the preferred information is sent to the subscribers.

3.2.1.3 Application Layer

This layer communicates with the end user via a GUI form. It consists of the code for GUI and its logic behind it. This layer has the minimum functionality in the whole system. Its task is only to take the user inputs and make a request to other layers.

3.3 Comparison of the models

Any distributed application can be developed in either client/server model or peer to peer model. The developer has to analyze and decide which architecture is best suited for the application before he/she starts writing the application logic. The following sub-sections discuss some of the issues and the merits and demerits that the developer must consider in choosing the appropriate approach for the application.

3.3.1 Ease of Development

The developers need to consider ‘Ease of Development’ factor before designing the application. The availability of development and debugging tools can accelerate the application development.

There are a considerably large number of application entities available for developing client/server applications. Several vendors provide APIs that can be used straight away, such as
messaging package, application servers, and web servers, for developing a client/server application with central server.

There are some packages available for developing peer to peer applications, such as Sun’s JXTA package [46]. But when compared to client/server packages, these packages are quiet new, therefore not quite fully developed. Hence there is a big chance of encountering an unknown error.

Moreover, debugging and testing in a distributed system is much heavier and time consuming than debugging and testing in a single system, i.e. centralized server, because the interaction is required among several software entities in distributed system.

Therefore, the client/server methodology takes an advantage over the peer to peer approach from an ease of development viewpoint.

3.3.2 Scalability

The scalability of an application is measured in terms of the highest rate or size user interactions that the application can support [45].

As, in the peer to peer application, all the computer are directly connected to each other, it uses large number of computers to solve the problem. Thus peer to peer applications are more likely to provide more scalable solution than that of the client/server application, which relies on a single centralized server.

But, the modern architecture of the client/server model uses multiple servers for better performance and scalable solution. Therefore, if the number of servers in client/server application is same as the number computers in peer to peer application, then the client/server application provides the better scalable result. This is because; the peer to peer application needs more interactions than the client/server application.
However, the cost of maintenance increases as the number of computers increase. Thus, considering the cost factor, peer to peer applications produce more scalable results at lower costs.

### 3.3.3 Reliability

The reliability of the application is measured by its ability to continue to work when one or more system(s) in the network fail [45]. Highly reliable applications can be developed by using either the peer to peer approach or client/server approach. As the peer to peer application consists of many computers doing the same task, the system continues working even if several nodes die, whereas, in a single centralized client/server application, the whole application dies when the server goes down. This problem can be taken care by introducing more servers into the system, but it increases the cost of the application.

Both the applications are reliable, but the reliability in peer to peer application is achieved at much lower cost than the client/server application.

### 3.3.4 Manageability

Even after the application is ready, the developers and administrators must ensure that the application is running effectively. So, the application needs ongoing maintenance.

In a peer to peer application, the application running on different machines could by spread wide in the geographic area. All the computers may not be running on the same operating system. This makes the manageability harder for peer to peer application.

It is easier to manage a single centralized server when compared to managing many distributed systems. Therefore, the client/server application needs relatively less maintenance when compared to peer to peer application.
3.3.5 Security

Security is one of the factors that should be managed by the administrators after deploying the application. There are several tasks in managing the security of the application, such as authenticating only the authorized users to use the application and continuously monitor users so that malicious users do not upload viruses into the application.

The security of a centralized system is easier to maintain when compared to that of the distributed model. In peer to peer application, the security entity must be replicated and distributed to all the systems on the network. Moreover, there are many points for the hacker to enter into the system. In client/server application, the security entity can be only installed on the server, because in this model no two clients can communicate with each other directly without involving the server. Thus, the client/server application is more secure than the peer to peer application.

Considering all the above five issues, the client/server application provides better security, manageability, and ease of development, whereas, the peer to peer application produces the increased scalability and reliability in a more cost-effective way. These discussions just help in choosing the best approach for developing an application. It is the developers who choose the best approach in their point of view.
Chapter 4

Implementation Characteristics

This research proposes a low-cost software to support distance learning through the use of an interactive whiteboard. It allows users to interact by drawing, writing and erasing on the whiteboard as well as through the use of text messaging. Moreover, if the display device is equipped with the touch screen, users can use their fingers to demonstrate the solving of any complex problems. The application has been designed in a way that it can be used on any personal computer that supports Internet/Intranet connections.

The application’s characteristics include:

- A whiteboard interface on which users can share on-screen drawings. The user can write or erase on the whiteboard using his/her finger or a stylus, provided the display device is touch interfaced;

- A broadcasting messenger system, through which users can interact by text messaging by using a keyboard and display interface;

- A database to store the user’s credentials;

- The support for multiple virtual classrooms at any given time.
My approach to this is to structure the application as two fundamental components, the database and a set of virtual classrooms. Figure 4.1 illustrates the communication flow between the virtual classroom, which consists of several students and an instructor, and the database. One important characteristic of this is that the database communication is required only for users to log in. The database communication link is disconnected once the client (student) establishes the direct communication link with dynamic server (instructor) within the virtual classroom. Therefore, the load on the database server is reduced. The instructor can log in as an instructor or as a student on the same machine. If the user logs in as both, an instructor and a student, on the same machine, then this machine works as both, the server and the client.

![Diagram](image-url)

Figure 4-1: Communication flow between virtual classroom and database.
It is assumed that the database server is always up and running. A database crash will block any user, not logged in yet, to log in to the application. It is also assumed that all virtual classrooms have access to Internet or Intranet. The concept of a virtual classroom is interpreted as group of clients (students) associated to a dynamic server (instructors). These clients and dynamic server can be located anywhere, provided that they all have access to Internet. Figure 4.2 shows the detailed communication flow between the clients, the dynamic server and the database server.

Figure 4-2: Communication between clients, dynamic server and database server.
As shown in the Figure 4.2, the blue doubled arrow line represents the communication between the database and the dynamic server and clients. This is the very first communication link established when the application is started. The dynamic server and clients initiate this communication when users initially try to log in to the application for both verifying their credentials as well as registering their session. This means both the dynamic server and clients must register to the database first. This activity requires the user identification (ID), password and the course name to be sent to the database. If either the user’s ID and/or password are not recognized, the user will not be logged in and the application prompts the user to enter his/her credentials again. If the user credentials are matched, the database sends an acknowledgement along with the user’s full name and the internet protocol (IP) address of the respective dynamic server, if available. Once the user is successfully logged in to the application, his/her database communication line is disconnected freeing the database queue to be used by other users.

The clients use the IP address received from the database server to establish the communication line between the clients and the associated dynamic server. This communication is based on socket-based client/server model. In client/server communication model, a client is a program which runs on a remote machine making service requests and a server is a program which listens to client requests and processes them. In particular, in socket-based client/server model, a server program listens to a specific port for client programs sending service requests [47]. Figure 4.3 shows the communication flow in the client/server communication model.
In Figure 4.2, the orange doubled arrow line represents the client/server communication. Once this communication line is successfully created, all the associated users are grouped to form a virtual classroom. The application does not allow any two clients to communicate with each other directly for security reasons. All these communication must be through the server in the virtual classroom.

Figure 4.4 depicts the architecture of the whole application. It is composed of the following three main components:

- Database layer;
- Application layer; and
- GUI layer.
The sections that follow address these three layers.

### 4.1 Database Layer

The components which implement a database ensure that the end users always have access to important information at any time in order to service the requests of clients and the dynamic server. For this reason, the database server should always be available. If the database server is not functioning properly, then no new user can log in and no user can request the database for any information, but the users already logged in and connected to virtual classroom can share their information within the virtual classroom. This application uses the MySQL Server [48] for the database server.

The code supporting the registration process checks weather the user has entered all the required fields in the form. And once the user agrees to register, all the entered information is sent to the
database over the network and saved as a record in the database. If the user is already registered and wants to log in, a request is sent to the database and the appropriate information is sent to the application layer or the GUI layer immediately after reception of request.

The information in database can be retrieved, modified or added at any point of time, provided the user has the required permissions. Such permissions are granted only to the database administrator. The database server for this application should be running on one particular machine and its IP address should not be changed. The students are allowed to create their user accounts whenever they wish to, but they are not allowed to register for any courses without the proper permissions. For this reason the students will have to contact the database administrator or their instructor to register for any course.

The database administrator should be aware of the following for maintaining the existing database:

- Data regarding the total number of students and instructors which include their names and respective identification numbers, and;
- The courses offered and their respective course identification numbers;

The structure for the database layer includes three tables, Students, Professors and Courses. Details of such components are illustrated in Tables 4.1, 4.2 and 4.3 respectively.

Every course use these tables, these tables are like primary tables. For each course a table with course name must be created. These tables contain only the identification numbers of the users registered to a particular course. The other tables in the database can be created or deleted at any point of time only by the database administrator – these tables are only for the administration purposes, like retrieving names of the students in a particular course, to check who all are attending the discussion at a particular time, etc. Details of three tables are discussed as follows.
Table 4-1: Students

<table>
<thead>
<tr>
<th>Fname</th>
<th>Lname</th>
<th>Uid</th>
<th>Pass</th>
<th>ip</th>
<th>sip</th>
<th>cno</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mark</td>
<td>Dillon</td>
<td>R00345876</td>
<td>****</td>
<td>72.241.189.56</td>
<td>68.240.195.4</td>
<td>6960</td>
</tr>
<tr>
<td>Rajeev</td>
<td>Reddy</td>
<td>R00246531</td>
<td>****</td>
<td>68.241.168.65</td>
<td>Null</td>
<td>6990</td>
</tr>
<tr>
<td>Sagar</td>
<td>Sharma</td>
<td>R00987456</td>
<td>****</td>
<td>75.220.178.45</td>
<td>68.240.195.4</td>
<td>6960</td>
</tr>
<tr>
<td>Henry</td>
<td>Meyers</td>
<td>R00765567</td>
<td>****</td>
<td>72.241.189.48</td>
<td>74.232.174.33</td>
<td>6743</td>
</tr>
</tbody>
</table>

Table 4.1 consists of all the data related to each student. In this table column identifiers are associated with the following semantics:

fname – First Name
lname – Last Name
uid – Identification number (may be the university id)
pass – password
ip – IP address of the machine on which they are logged in
sip – server IP address to which they have to be connected
cno – Course number of the course registered by that particular student

Table 4-2: Professors

<table>
<thead>
<tr>
<th>Fname</th>
<th>Lname</th>
<th>Uid</th>
<th>Pass</th>
<th>ip</th>
<th>cno</th>
</tr>
</thead>
<tbody>
<tr>
<td>George</td>
<td>Lindsay</td>
<td>R00948231</td>
<td>*****</td>
<td>74.232.174.33</td>
<td>6743</td>
</tr>
<tr>
<td>Steven</td>
<td>Taylor</td>
<td>R00593458</td>
<td>*****</td>
<td>78.212.189.32</td>
<td>6945</td>
</tr>
<tr>
<td>Atul</td>
<td>Kumar</td>
<td>R00680135</td>
<td>*****</td>
<td>68.240.195.4</td>
<td>6960</td>
</tr>
</tbody>
</table>

Table 4.2 consists of all the data related to the instructors. It has the same attributes as the Student’s table, but it does not have the “sip” one because the instructor’s machine itself is a server.
Table 4-3: Courses

<table>
<thead>
<tr>
<th>cname</th>
<th>cno</th>
</tr>
</thead>
<tbody>
<tr>
<td>Independent Study</td>
<td>6960</td>
</tr>
<tr>
<td>Introduction to Java</td>
<td>6743</td>
</tr>
<tr>
<td>Image Processing</td>
<td>6945</td>
</tr>
</tbody>
</table>

Table 4.3 consists of the data regarding the course names and their course numbers. It has only two columns. This table's attributes are associated with the following semantics:

cname – Course Name

cno – Course number of the associated course

For security reasons, all information stored in the database is only accessible to the database administrator and is only used for the back-end manipulations.

4.2 Application Layer

The application layer is the heart of the system. It is responsible for processing of user requests. This layer is also responsible for retrieving the relevant information from the database and displaying it on the GUI layer for the user.

The application layer is divided into two parts:

- Broadcasting messenger application; and
- Whiteboard application.

Before I discuss the above two components in detail, it is important to introduce two important modules which are common to both of them:

- The Validate module; and
• The NewUser module.

The Validate module is the first module to be executed when the application starts. It is responsible to verify the user credentials in the database with the information entered by the user during the log in process. This module allows users to interact with the system by providing a GUI frame to users, asking them to enter their user credentials for verification. The Validate module redirects the users to the main screen of the application, provided the log in was successful and the course instructor is available.

The NewUser module allows new users to register with the application’s database. This class services are available to users through a GUI form.

This form asks the user for all the details required for creating his/her account, such as name, ID and password. The programming logic, supporting this behavior, checks weather the user has entered all the required fields in the form before a record is created in various tables in the database, once the user is ready for creating his/her account. If any required field is missing or it is not valid, the user will be asked to provide all the necessary information again.

4.2.1 Broadcast Messenger

The broadcast messenger component broadcasts any text messages entered by the user. In a virtual classroom, the text message entered by any client is first sent to the dynamic server. The server then broadcasts the text message to all other users logged in. In other words, the dynamic server is responsible to send the text message entered by any user to all other users logged in.

As stated previously, the application proposed here not only supports text based user interaction. It also supports interactions through the whiteboard. The flexibility of allowing both forms of interactions has limited the use of the standard Application Programming Interfaces (APIs) that are available through the Java programming language. For this reason, I designed and coded my
own protocols by using socket programming. In socket communication, the server opens a port and continuously listens on it for any client requests. The clients send requests for connection on that port and once both the server and client are connected, a virtual channel is established between them. This communication channel is open and it will be available for communication until the close command is explicitly called, which should happen when the user logs out of the application.

The broadcast messenger application consists of two parts:

- Client; and
- Server.

The application is designed to redirect the instructor to the server and the student to the client as soon as they log in to the application. The Validate module is responsible to verify their ID and redirect them accordingly. To create a virtual classroom, the instructor has to log in before the students to create a session. If a student logs in to the application before the instructor, then he/she makes multiple communications with database server to check whether the instructor is available yet.

In client/server communication, it is the client which usually initiates or requests for a new connection. The server listens to incoming connection requests. The server starts listening for any new incoming requests again. This process should be running continuously, in the background, until the server is shut down. To achieve this, I have used multi-threading feature provided by java. A thread in java is an execution flow of any process.

The Client module then retrieves the server IP address from the database and by using this IP address it tries to connect to that server by sending a request to open a new connection. When the server accepts this connection, a virtual channel is established for the communication and the client receives a welcome message from the server. In this application, the students request the instructor
to join the virtual classroom. The instructor (dynamic server) creates a thread for every student attending the virtual classroom.

In order to create the socket connection, the socket has to link the server program with a particular port on the computer where it runs so that the client program running anywhere with a socket associated with that same port can converse with the server program. By making the server multi-threaded, the server is able to handle requests from more than one client. As soon as the server is started, its IP address is record in the database. When a client starts, it retrieves the server’s IP address from the database and sends a request, to that server, to open a new connection. To make this request the client invokes the open method in SocketChannel class. The server program is interrupted by the reception of this request and invokes the accept method in ServerSocketChannel class to accept the request from the client. If the socket channel is created successfully, the server program creates a thread for this communication and sends a welcome message to the newly connected client and broadcasts a message to all clients logged in informing about the new connection. The server program continues to listen for requests from other clients.

The users (clients and server) can now enter the text in their frame and click on send button to broadcast the message. Note that no two clients are connected to each other. Therefore, any message they want to broadcast will be routed through the server. The clients only send the message to the server and the server will broadcast it.

### 4.2.2 Whiteboard

Whiteboard allows the students and the instructor to communicate in an interactive manner by providing them with the flexibility to communicate with their hand written figures. This whiteboard application allows the users to draw and erase figures by using a simple pointing device, such as mouse. If the display device used by the users is equipped with the touch interface panels, then the
users can use their fingers or the stylus provided along with the display device to draw figures. The information on the whiteboard frame is distributed to all the users logged in to the application.

The functionality of the whiteboard is achieved by using event handlers. For example, when the user clicks the mouse or drags the mouse pointer, a different mouse event is raised accordingly. There are many events occurring in java, mouse events are essential for this part of application. The main mouse event used in this application is mouseDragged event. This event is raised when the user drags the mouse, which is when the user draws a line. The event handler mouseDragged is executed when this event is raised.

The mouseDragged event handler paints and records the co-ordinates of the every point in the path which the user drew with the mouse. These points are sent to the server and the server broadcasts them to all the clients logged in to the application. The clients receive these points and paints on their screens. Therefore, every client logged in can view the shared whiteboard simultaneously and instantly.

The network communication technique used for the whiteboard application is exactly the same as that of the broadcast messenger, which I have discussed in the previous section. The only difference is that, the broadcast messenger shares text in the form of byte array and the whiteboard application shares the co-ordinates of the point in the form of byte array. The whiteboard window is only displayed when the user is successfully logged in and is ready for network communication.

There are many important networking attributes of the application which are shared by both, the broadcast messenger and the whiteboard. Some of them are IP address, port number, user ID etc. Because the whiteboard window is displayed only when the user is successfully logged in, it does not require any database communication which increases the performances of the application and minimizing the load on the database server. Hence, it does not have to communicate with the static server. The database server is referred as static server in my application because the machine running
database server should be in at same location at all times so that the IP address of this machine is not altered.

As discussed earlier that this whole application is based on client/server communication model and all client data has to be transported to the server first and then the server transmits all the data to all other users logged in, the time taken for complete execution of the application is more than that of the peer-to-peer communication model for smaller networks (less than twenty machines in a network). In this process, some of the data (points) recorded from the whiteboard may be lost in the transportation as the client/server communication model does not guarantee the delivery of the data. The amount of data lost depends on many factors, the time taken for execution of the application, the speed at which the user writes on the whiteboard, the actual physical distance between the client and the server, the speed of the internet, and the processor speed of the machines. Therefore, the users can expect distorted image on the interactive whiteboards.

4.3 GUI Layer

The main components of the GUI layer are.

- Login screen;
- User registration screen;
- Messaging screen; and
- Whiteboard screen.

4.3.1 Login Screen

This is the first screen which appears when the application starts. Here the user is asked to enter his/her credentials. Figure 4.5 shows the screen shot of the login screen.
As shown in the figure 4.5, the users have to enter their user name, password and the course name for which they will be attending. The user name and password are required for user verification and validation. The course name is required to check whether the user is registered to that course. A database connection will be established, once the ‘Sign In’ button is clicked, to access the user’s record for verifying the information provided by the user. If the user is registered to that course, then the user will be logged in to the application and is re-directed to the respected virtual classroom.

The user has the option to register, if he/she is not yet registered to the system. The user will be re-directed to the registration form once he/she clicks on the ‘Register’ button.

**4.3.2 User Registration Screen**

This screen is displayed when the new user wants to register to the database. Figure 4.6 is the screen shot of the registration form.
All the fields in shown in figure 4.6 must be entered correctly for the user to register. A database communication link is established and all the fields in the registration form are retrieved and saved. A record for the user will be created in the database, once the registration is successful. The user’s first name and last name will used as their screen name while attending a class. The user name and the password are required for the users to log in to the application. The user name should be the university student ID, so that the user can be allowed to register if he/she has a record in university database, provided the university grants permission to access their database. A ‘Close’ button is provided in the form to cancel and close the registration form, if the user decides not to register. In this case, no database connection is established.

4.3.3 Messaging Screen

This is one of the two screens opened as soon as the user is logged in successfully. Figure 4.7 shows the screen shot of the messaging screen.
As shown in the Figure 4.7, there is scroll pane under the label ‘Discussion Board’ where all the communication is shown. This field records and displays all the text based messages received from all other users logged in. There is a text field under the label ‘Enter your text’, in which the user can enter any text that is to be sent over the internet. Once the user clicks on ‘Send’ button, the text in the text field is retrieved and is first sent to server. As soon as the server receives the message, it broadcasts to all the clients logged in.

There is no ‘Log out’ button provided for the users. If the user wants to log out, then they can log out just by closing any of the windows opened or by sending the message ‘quit’.
4.3.4 Whiteboard Screen

This is one of the two screens opened as soon as the user is logged in successfully. Figure 4.8 shows the screen shot of the whiteboard screen.

![Whiteboard Screen](image)

Figure 4-8: Whiteboard screen.

The whiteboard screen just like a white paper, where hand written information and/or drawings may be entered. There are no buttons provided in this frame, but there are two tools provided with radio buttons. These tools are:

- **Write**: The users can write on the frame with this tool. If the ‘write’ radio button is selected, then the user can start writing;

- **Erase**: The users can undo any writings by using this tool. If the ‘erase’ radio button is selected, then the ‘write’ button is automatically deselected and the user can erase the contents on the interactive whiteboard.
Any scribbling on this frame is sent to all other users logged in automatically and dynamically. That is, unlike text based messaging, the user need not click any button to send the contents of this frame. As the user scribbles on this frame, the mouse events are triggered and the respective event handlers will take care of broadcasting the contents of the frame at the background.

4.4 Example Scenario

One of the ideas behind the implementation of this application is to facilitate the two way interaction among the students as well as the instructor with the interactive whiteboard. Consider, for instance, the distance learning course where the instructor needs to introduce the concept of logic gates. Assume the students have already been exposed to the notion of Boolean expressions.

The instructor can use the text-based messaging to write the simple Boolean expressions and the interactive whiteboard to actually draw and explain the logic gates. Figure 4.9 and Figure 4.10 show the text-based messaging screen and and whiteboard screen of a student while the instructor is explaining the NOT gate.
Figure 4-9: Text-based messaging screen of a student while the instructor is teaching.

Figure 4-10: Whiteboard of a student while the instructor is teaching.
As shown in the Figures 4.9 and 4.10, the instructor uses the text-based messaging system to let students know what he/she is going to demonstrate on the whiteboard.

After the demonstration of the NOT gate, the instructor continues his/her lecture by showing the AND and OR gates. Figure 4.11 and Figure 4.12 show the instructor's activity while demonstrating the AND and OR gates.

![Text screen of a student at mid-session.](image)

Figure 4-11: Text screen of a student at mid-session.
So far, in this example scenario, most part of text and whiteboard data are shared by the instructor. As I had mentioned in the previous sections the server application is responsible to broadcast the data and the machine on which the instructor logs in acts as the server, the data shared by the instructor is displayed on his/her screen and broadcasted to all the students logged in.

Now, to make the students active, the instructor may raise a question in the discussion board of the text-based messenger. The students can reply to the instructor on the text-based messenger and also use the whiteboard, hence making the application two-way interaction between the students and the instructor. The Figure 4.13 shows such activity.

Figure 4-12: Whiteboard screen of a student at mid-session.
The instructor asked a question and a student is ready to solve it on the whiteboard. But the student has no space on the whiteboard to draw. For this reason, the whiteboard has the tool “erase” to erase the diagrams on the whiteboard. The Figure 4.14 is the screen shot taken while a student is erasing on the whiteboard. The eraser can be enabled by clicking on the “Erase” radio button on the top of the whiteboard.
Figure 4-14: Screen shot of the student's whiteboard screen while erasing.

Figure 4.15 is a screen shot of the instructor's whiteboard after the student has demonstrated his/her solution. Typically every user logged in to a particular virtual classroom share the screens at any given time, because the data in this application is shared at real-time dynamically.
As shown in the Figures 4.14 and 4.15, the student is involved in the interaction. I had mentioned in the previous Chapters that no two clients can communicate each other because the application is based on the client/server model. Any such communication should be via a server.

In this example, the student (client) is using the whiteboard. The data from the student’s text and whiteboard screens are first sent to the server (instructor), and then the server broadcasts the data to all the students in the virtual classroom.

All the interactions in this section are within a virtual classroom and does not involve database anywhere, hence reducing the load on the database server. Also, all the virtual classrooms are by themselves and there is no communication link between any two virtual classrooms. A virtual classroom is capable of holding 300 students and an instructor at any given time.
Chapter 5

Conclusions and Future Work

5.1 Conclusion

This research addressed interaction in distance learning. In Chapter 1 I have made two claims. I repeat them here with comments about how they have been addressed in the solution this work introduces.

1. A distance learning environment can be improved by facilitating the communication among the students as well as the students and the instructor.
   - I designed and implemented an interactive whiteboard which allows a user to create and share real-time drawings.
   - I designed and implemented a broadcast messenger which allows a user to create and share real-time text messages.

2. An interactive low cost system can be provided to support claim 1.
   - I designed and implemented a client/server based system using the Java programming language. The application proposed supports a large number of virtual classrooms with the capability of holding 300 users in each virtual
classroom at any given time. The hardware requirements for the proposed system are a standard personal computer equipped with a mouse, a keyboard and display. The application can be deployed from either Microsoft Windows XP or any later version or any distribution of Linux. It also requires support for Internet. The cost associated with installation, operation and maintenance is therefore minimized.

5.3 Future Work

One of the important reasons to work in this area of research is to encourage future work in the same field so that the distance learning applications with low cost, high efficiency and minimal data loss can be developed. The various encouraging results obtained from the application indicate the following possible upgrades:

- A voice transmission can be integrated into the application, so that the users can also communicate through speech;
- A power point presentation can be integrated to broadcast the presentation slides, already saved on disk, through a separate window;
- The assignment of color codes for every user, so that the users know what modification is done by which user; and
- An increase in efficiency to minimize the data losses. The efficiency can be increased by various factors, for example, by maintaining a powerful server machine, instantly increases the efficiency of the application.
References


5. Lokken, Fred. 2010 Distance Education Survey Results. 2011.


22. Designing Communities of Learners for Asynchronous Distance Education. **Moller, Leslie.** 4, 1998, Educational Technology Research and Development, Vol. 46.


28. **Jonassen, David H.** Handbook of research for educational communications and technology: a project of the Association for Educational Communications and Technology. s.l.: macmillan Library Reference USA, 1996.


32. Tandberg Codec C60. Clary Business Machines. [Online]


35. Pricelist Template Form. Extron Electronics. [Online]
http://www.peppm.org/Products/extron/price.pdf.


42. Smart Board X885i4. SmartBoards.com. [Online]


Appendix A

Source Code

A.1 GUI Layer

This section shows the source code of the important components of the GUI layer.

A.1.1 Login Screen

```java
public class Validate implements ActionListener {
    JFrame valFrame = new JFrame();

    private JButton register = new JButton("Register");
    private JButton signIn = new JButton("Sign In");
    private JTextField uid = new JTextField(20);
    private JPasswordField password = new JPasswordField(20);
    private JTextField courseTxt = new JTextField(20);
    private JLabel uName = new JLabel("User Name :    ");
    private JLabel pass = new JLabel("Password :     ");
    private JLabel course = new JLabel("Course Name :");
    private JLabel newUser = new JLabel("Are you a new user?");
```
private String name;

private String passwordMatch;

private String courseName;

private Connection conn = null;

private Statement stmt;

private ResultSet rs;

private ClientApp ca = null;

private int courseNo = 0;

private String sip = null;

public Validate () {
    valFrame.setResizable(false);
    valFrame.setTitle("Login");
    valFrame.setSize(350,200);
    Container cp = valFrame.getContentPane();
    cp.setLayout(new FlowLayout());
    cp.add(uName);
    cp.add(uid);
    cp.add(pass);
    cp.add(password);
    cp.add(course);
    cp.add(courseTxt);
    cp.add(new JLabel("                                       "));
    cp.add(signIn);
}
A.1.2 New User Registration Screen

public class NewStudent {

    private JFrame f = new JFrame();
    private JFrame jf1 = null;
    private JButton submit = new JButton("Submit");
    private JButton close = new JButton("Close");
    private JTextField fNameTxt = new JTextField(20);
    private JTextField lNameTxt = new JTextField(20);
    private JTextField uidTxt = new JTextField(20);
    private JPasswordField passwordTxt = new JPasswordField(20);
    private JPasswordField conPasswordTxt = new JPasswordField(20);
    private JLabel fname = new JLabel("Your First Name : ");

    cp.add(new JLabel(" "));
    cp.setLayout(new FlowLayout());
    cp.add(new JLabel(" "));
    cp.add(newUser);
    cp.add(new JLabel(" "));
    cp.add(register);
    signIn.addActionListener(this);
    register.addActionListener(this);
    valFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    valFrame.setVisible(true);
}


private JLabel lname = new JLabel ("Your Last Name : ");
private JLabel uid = new JLabel ("User Name : ");
private JLabel password = new JLabel ("Password : ");
private JLabel conPassword = new JLabel ("Confirm Password : ");
private Connection conn = null;
private ClientApp ca = null;

public NewStudent (JFrame jf) {
    this.jf1 = jf;
    f.setResizable(false);
    f.setTitle("New Student Registration");
    f.setSize(375,220);
    Container cp = f.getContentPane();
    cp.setLayout(new FlowLayout());
    cp.add(new JLabel("                               "));
    cp.add(new JLabel("All Fields Are Required                 "));
    cp.add(fname);
    cp.add(fNameTxt);
    cp.add(lname);
    cp.add(lNameTxt);
    cp.add(uid);
    cp.add(uidTxt);
    cp.add(password);
    cp.add(passwordTxt);
A.1.3 Text Messaging Screen

```java
public class MsgFrame extends JFrame {
  JTextArea chatBox = new JTextArea(10,45);
  private JScrollPane myChatHistory = new JScrollPane(chatBox,
                                                      JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
                                                      JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
  private JTextArea userText = new JTextArea(5,40);
  private JScrollPane myUserHistory = new JScrollPane(userText,
                                                      JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                                                      JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
  private JButton send = new JButton("Send");
  private JButton quit = new JButton("Quit");
  private ClientApp chatClientOut;

  public MsgFrame(ClientApp chatClient) {
    this.chatClientOut = chatClient;
    setResizable(false);
    setTitle("Text-Student");
  }
}
setSize(560,400);
Container cp = getContentPane();
    cp.setLayout(new FlowLayout());
    cp.add(new JLabel("Chat History"));
    cp.add(myChatHistory);
    cp.add(new JLabel("Chat Box : "));
    cp.add(myUserHistory);
    cp.add(send);
    cp.add(quit);
}

A.1.4 Whiteboard Screen

public class WhiteFrame extends JFrame {

    public Paper paper;

    public WhiteFrame (ClientApp chatClient) {
        paper = new Paper (chatClient);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        getContentPane().add(paper, BorderLayout.CENTER);
        setTitle("Whiteboard-Student");
        setSize(640,480);
        setVisible(true);
    }
}
class White extends JPanel {

    private HashSet hs = new HashSet();
    private ClientApp chatClient;
    private JRadioButton write, erase;
    private JPanel bp = new JPanel();
    private Color color = Color.BLACK;

    public White (ClientApp chatClient) {
        this.chatClient = chatClient;
        ButtonGroup buttonGroup = new ButtonGroup();
        write = new JRadioButton("Write");
        erase = new JRadioButton("Erase");
        buttonGroup.add(write);
        bp.add(write);
        buttonGroup.add(erase);
        bp.add(erase);
        add(bp, BorderLayout.PAGE_END);
        write.setSelected(true);
        setBackground(Color.white);
        addMouseListener(new L1());
        addMouseMotionListener(new L2());
    }
}
A.2 Network Communication

This section shows the source code for broadcasting, sending and receiving text and whiteboard data.

A.2.1 Broadcasting Text and Whiteboard Data

```java
long num = 0;
long buf_len = writeBuffer.remaining();
while (num != buf_len) {
    try {
        num += client.write(writeBuffer);
    } catch (IOException ex) {
        ex.printStackTrace();
    } catch (InterruptedException ex) {
    }
}
```

A.2.1 Receiving Text Data

```java
try {
    ByteBuffer readBuffer = rb;
    StringBuffer strBuffer = sb;
    String line = "";
    String str = readBuffer.toString();
    strBuffer.append(str);
    strBuffer.append(str);
    readBuffer.clear();
}
```
line = strBuffer.toString();

if ((line.indexOf("\n") != -1) || (line.indexOf("\r") != -1)) {
    line = line.trim();
    msgFrame.chatBox.append(line + "\n");
    sb.delete(0,sb.length());
}

} catch (Exception e) {

}

A.2.2 Receiving Whiteboard Data

try {
    boolean write = true;
    Color color = Color.BLACK;
    char we = readBuffer.getChar();
    int x = readBuffer.getInt();
    char c = readBuffer.getChar();
    int y = readBuffer.getInt();
    readBuffer.clear();
    Point p = new Point(x,y);
    if (we == 'w') {
        write = true;
        color = Color.BLACK;
    }


```java

else if (we == 'e') {
    write = false;
    color = Color.WHITE;
}

whiteFrame.paper.addPoint(p, write, color);

}

catch (Exception e) {

}

```