MULTI-SERVER COLLABORATION SYSTEM FOR DISASTER RELIEF MISSION PLANNING

By

Chang Kuang

Bachelor of Science in Environmental Engineering, Tsinghua University, China
Bachelor of Science in Computer Science, Tsinghua University, China

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

Master of Science

at the

Massachusetts Institute of Technology

June 2001

Author..
Department of Civil and Environmental Engineering
May 15, 2001

Certified By...
Fenißsky Peña-Mora
Associate Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted By................ ....
Oral Buyukozturk
Chairman, Department Committee on Graduate Studies

MULTI-SERVER COLLABORATION SYSTEM FOR DISASTER RELIEF MISSION PLANNING

By
Chang Kuang

Submitted to the
Department of Civil and Environmental Engineering
June 2001
In Partial Fulfillment of the Requirements for the Degree of
Master of Science

Abstract

Disaster relief missions involve geographically dispersed participants working in a stressful environment towards a common goal. When a large-scale disaster occurs, issues such as search and rescue, survivor relocation and infrastructure rebuilding, need to be addressed in a short amount of time. Several constraints exist in this environment such as cultural and organizational differences among the relief agencies, the need for flexibility to handle changes, and widespread uses of wireless networking and PDAs (Personal Digital Assistants) by disaster relief personnel for their private and professional use. These constraints lead to the need for a multi-server collaboration system that includes support for meeting protocol enforcement, wireless device access, multi-server synchronization, seamless client handoff, and reliable and scalable collaboration services during periods of highly unstable network conditions. This dissertation presents a component-based collaboration infrastructure to support multi-server, multi-client and multi-device collaboration sessions. The system components include a data repository, a collaboration manager and an information policy enforcer to address the constraints of disaster relief planning missions. The proposed infrastructure can be configured in multiple ways allowing access to any collaborator regardless of their network or device limitations. Although the requirement analysis is conducted based on disaster relief mission planning, the system architecture, design and implementations can be extended to general collaboration situations.

Five important mechanisms aimed at improving the performance and scalability of the multi-server collaboration system and improving the collaboration efficiency for collaborators, are discussed in details. These mechanisms are:

- Mechanism for improving performance and reliability
- Mechanism for seamless handoff of clients among multiple servers
- Mechanism for managing heterogeneous interfaces carried by distributed collaborators
- Mechanism to make applications collaboration-ready
- Mechanism to make participants collaboration-savvy

Thesis Supervisor: Feniosky Peña-Mora
Title: Associate Professor of Civil and Environmental Engineering

ACKNOWLEDGMENT

May 2001

I would like to thank my advisor Prof. Feniosk Peña-Mora, for the ceaseless guidance and support in my research and thesis. I feel especially grateful for Prof. Peña-Mora, from whom I learn how to improve not only on my research capability, but also on my life attitude.

Special thanks to all the members of the team in its various incarnations from IESL to inMeeting.com and beyond: Kiran Choudary, Jaime Solari, Justin Mills, Gyanesh Dwivedi, Sen Sugata, Sanjeev Vadhavkar, Padmanabha Vedam, and Ajit Sutar, without whom this research would not have as much fun. I would like to express my immense gratitude for Joan McCusker for her constant help for my research and daily life throughout the two years of my stay here.

Finally, I would like to thank my family and my friends for their continuous support over the past years.

中国快些强大起来吧！

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter presents the necessity of collaboration systems for disaster relief mission planning, and why multi-server systems are more suitable than traditional collaboration systems. The chapter also describes the background researches carried out by Da Vinci Initiative at MIT. The last part of the chapter lays out the architecture of this thesis.

1.1 Motivation

Disaster relief mission planning involves geographically dispersed participants working in a stressful environment towards a common goal. When a large-scale disaster occurs, issues such as search and rescue, survivor relocation and infrastructure rebuilding, need to be addressed in a short amount of time. Disaster relief relies on teams from multiple organizations, drawing on their expertise regardless of their physical location. Unlike many other situations, teams are being comprised of individuals and organizations, professional as well as volunteer. Technology driven collaboration systems could enable these teams to be more effective by increasing the quality and quantity of the collaboration during a disaster relief and mission planning.

1.1.1 Disaster Relief Mission Planning

Natural disasters, such as hurricanes, earthquakes, El Nino, forest fires, landmines, and volcanoes, cause serious loss, destruction, hardship, unhappiness, even death. Unfortunately, natural disasters occur frequently worldwide. Table 1 lists disasters occurred in the first quarter of the year 2001 (source: http://www.reliefweb.int/).

Table 1 - Natural Disasters Occurred in the First Quarter of 2001

Nation/Region	***Disaster***	***Period/Date***
Zambia	Floods	Mar 2001
Peru	Floods	Mar 2001
Brazil	Oil Platform Accident	Mar 2001
Ukraine	Floods	Mar 2001
Hungary	Floods	Mar 2001
Romania	Floods	Mar 2001
Zimbabwe	Floods	Mar 2001
Vanuatu and Fiji Islands	Tropical Cyclone Paula	Mar 2001
DR Congo	Volcanic Activity Nyamulagira	Feb 2001
Afghanistan	Earthquake	Feb 2001
Malawi	Floods	Feb 2001
Indonesia	Earthquake	Feb 2001
El Salvador	Earthquake	Feb 2001
Afghanistan	Cold Wave	Feb 2001
Mozambique	Floods	Feb 2001
Pakistan	Earthquake	Jan 2001
India	Earthquake	Jan 2001
Bolivia	Floods	Jan 2001
Ecuador	Galapagos Islands Oil Spill	Jan 2001
El Salvador	Earthquake	Jan 2001

Figure 1 intuitively shows the disasters occurred in the week between Apr. 1, 2001 and Apr. 6, 2001 [1]:

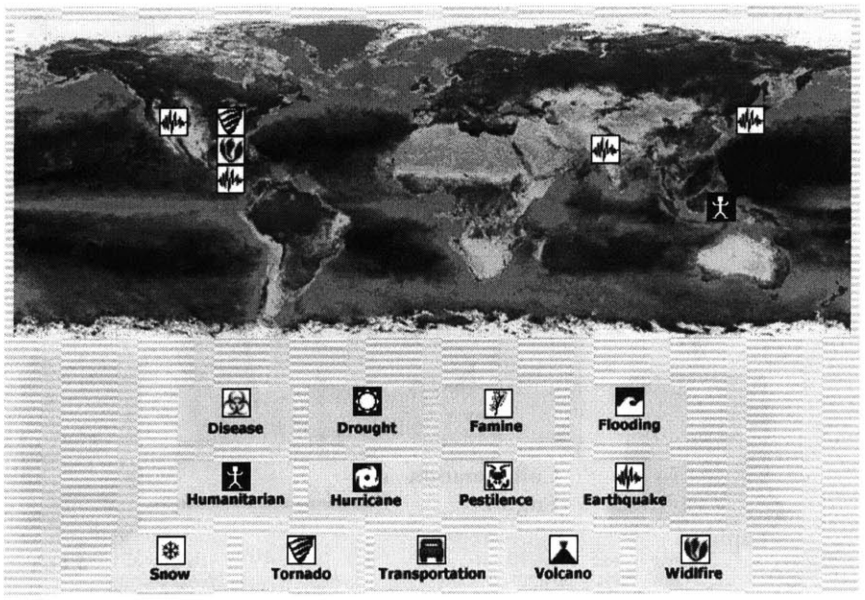


Figure 1 - Worldwide Disasters Occurred Between Apr. 1 and Apr. 6, 2001

1.1.2 Collaboration System

Distributed collaboration systems have been research topics in academic institutions [2,3] and research labs [4,5], and have been developed by commercial companies, such as WebEx.com (www.webex.com), Centra.com (www.centra.com), Placeware.com (www.placeware.com), and HelpMeeting.com (www.helpmeeting.com). Although these collaboration systems are evolving in terms of geographical dispersion, synchronous degree and information richness as a result of advanced technology and complicated software engineering practice, they are still fundamentally single-server systems.

1.1.2.1 Feature of Single-Server System

Single-server system normally runs a centralized server. Physically, collaboration service providers might deploy several servers to serve different collaboration service, but throughout the active collaboration session, only one server serves the client. Thus, a single server will control the whole collaboration session. Conceptually, this case is called single-server architecture. The logical topology of the servers and clients is similar to the star-topology, as illustrated in Figure 2.

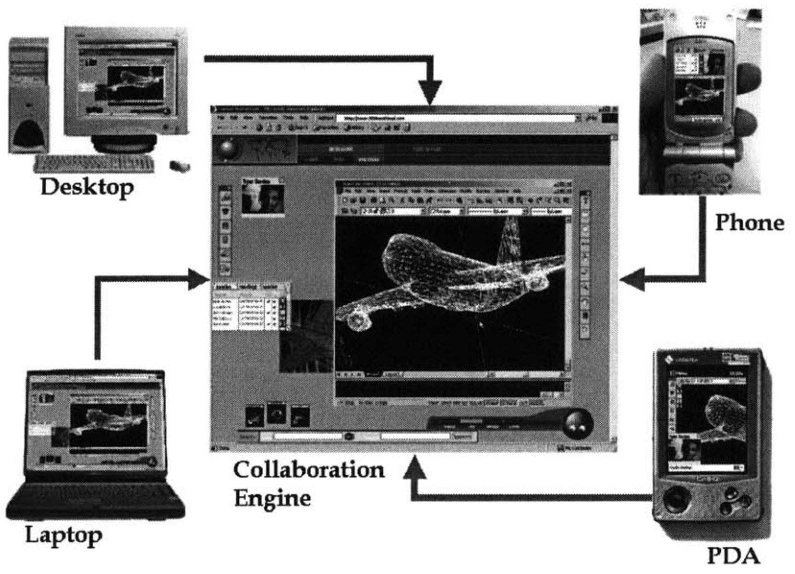


Figure 2 - Star Topology of Single-Server Collaboration Systems

The centralized single server does most of the heavy-duty work, such as:

- Schedule collaboration session
- Handle login/logout from multiple clients
- Enforce collaboration protocols
- Act as a central server for various collaboration services and tools, like text-chat, whiteboard, audio-video, real-time polling, and application sharing.

1.1.2.2 Constraints of Single-Server Architecture

Single-server architecture has the advantage of relatively easy implementation, maintenance, deployment and security. However, its smooth functionality relies on two important prerequisites. Basically the single server should be powerful enough to serve all the clients. Furthermore, the network between each client and the central server should be stable and fast enough. Unfortunately, in some circumstances, these requirements cannot be taken as granted.

- Firstly, during a typical collaboration session, participants would like to use services such as text-chat, whiteboard, audio-video broadcasting, real-time polling, tele-pointer and annotation, and application sharing. Some services, like application sharing and video broadcasting, require considerable CPU processing power, making single-server hard to handle multiple simultaneous sessions.
- Secondly, some collaboration services or tools such as application sharing and video-audio multicasting will generate considerable network traffic. By their very nature, multimedia data amounts from video-audio applications are substantial in size. During an application sharing session, the server has

to send the image of part or the whole screen to all the clients. It should be noted that in some cases, dramatic changes of the screen might occur more frequently than that of video broadcasting at certain frame per second speeds. Indeed, some empirical experiences have shown that application-sharing session may generate more traffic than video broadcasting. Thereafter single-server can easily become the bottleneck if the server has limited bandwidth connecting to clients, or network connection suffers from congestion.

- Thirdly, PDAs (**P**ersonal **D**igital **A**ssistant), like Windows CE, palm, phones and pagers, are widespread recently because of its mobility and portability. People are gradually demanding more than using PDA as a PIM (**P**ersonal **I**nformation **M**anager); rather, they hope to collaborate with remote participants with their PDA. We call this category of distributed collaboration as **Mobile Collaboration**. Since PDAs are weak to run PC applications themselves, a system that supports mobile collaboration must rely heavily on server-side computing to provide services for mobile users. In this sense, single-server might not be sufficient for mobile collaboration.
- Fourthly, dynamic wireless networks are gaining popularity rapidly, partly because of the advances of wireless analog systems, partly because of its ubiquity and convenience. Issues with wireless networks such as dynamical change of topologies and resources, unpredictable available bandwidth, variable error rates and limited coverage will get compounded, thus making single-server unable to provide quality of service for each client.

The above constraints might not be critical in normal daily usage of collaboration systems, like a web conferencing participated by a company executives discussing about their future plans. However, in the stressful disaster relief environment, single-server collaboration system might not be functional as required, due to the following limits:

- The disaster relief is a resource-intensive process, with large number of participants and complex patterns of interaction, coordination and collaborations. Centralized server might be the bottleneck during the peaks of collaboration sessions, or when there are bursts of collaboration requests.
- In a disaster environment, it is highly probable that the communication infrastructure, like PSTN (Public Switched Telephone Network) and digital networking system, like backbone fiber network, T1, T3 and ISDN, have already been destroyed or malfunctioned, let alone providing adequate bandwidth for clients to communicate with the central server. Under such circumstances, wireless networks, which have ubiquitous coverage, are especially useful for digital communications. Even when there is no coverage for publicly accessible wireless networks, mobile wireless transmission towers can be deployed in strategically chosen locations to provide data services. However, wireless networking is not the best companion for single-server collaboration systems; rather, multi-server

system is a must for coping with slow and error-prone networking problems.

- In a disaster environment, it is cumbersome or even difficult to widely deploy desktop PCs for a collaboration system, because of dangers or power outages. The PDAs should be deemed valuable in terms of its mobility and no need for electric power supply. Again, as what we argue before, multi-server systems should be better for collaborations among large number of PDAs.

To sum up, in a stressful or even hostile disaster relief environment, multi-server collaboration system with clients and servers connecting with wireless networks and with widespread using of PDAs is much suitable than a single-server collaboration system.

1.1.3 Objective of The Research

Realizing the constraints of single-server systems, this research focuses on designing and developing a multi-server collaboration system for disaster relief mission planning. Besides the functionality analysis, design and sample implementation for a multi-server collaboration system, the research explores the following mechanisms:

- Mechanism for improving performance and reliability
- Mechanism for seamless handoff of clients among multiple servers
- Mechanism for managing heterogeneous interfaces carried by collaborators
- Mechanism to make applications collaboration-ready
- Mechanism to make participants collaboration-savvy

1.2 Background

The research presented in this thesis has its foundation in the work done in the field of collaboration systems at both MIT [2] and Draper Laboratory [5]. Research within the DaVinci Initiative at MIT [6] has focused on interaction and communication within virtual and physical collaboration environments. Draper Laboratory has pursued collaboration for mission planning applications.

Both sides managed to integrate the two collaboration environments, based on the requirements of a disaster relief mission-planning environment. As a result, a flexible architecture for a single-server collaborative system for disaster relief planning has been developed, leveraging the application-oriented approach of Draper and the meeting research at MIT [7].

1.2.1 Collaboration Research at MIT

The research of DaVinci Initiative at MIT has been trying to understand how participants interact in the area of both physical and online meetings [2]. Some general observations of collaborations patterns are identified, among which, the protocols and rules used during a meeting have been the primary research outcomes. Meeting protocols establish how information is shared within the collaboration and how control of the meeting is established. CAIRO (**C**ollaborative **A**gent **I**nteraction and Synch**RO**nization) was developed as a realization of the collaboration research at MIT [2]. CAIRO is a meeting environment that encompasses many of the research results such as membership and meeting protocol enforcement (see Figure 3). Adapting a strong implementation of meeting protocols, CAIRO uses the Internet to communicate in a highly structured environment that includes protocol-enforcing agents to simulate a physical meeting with a facilitator.

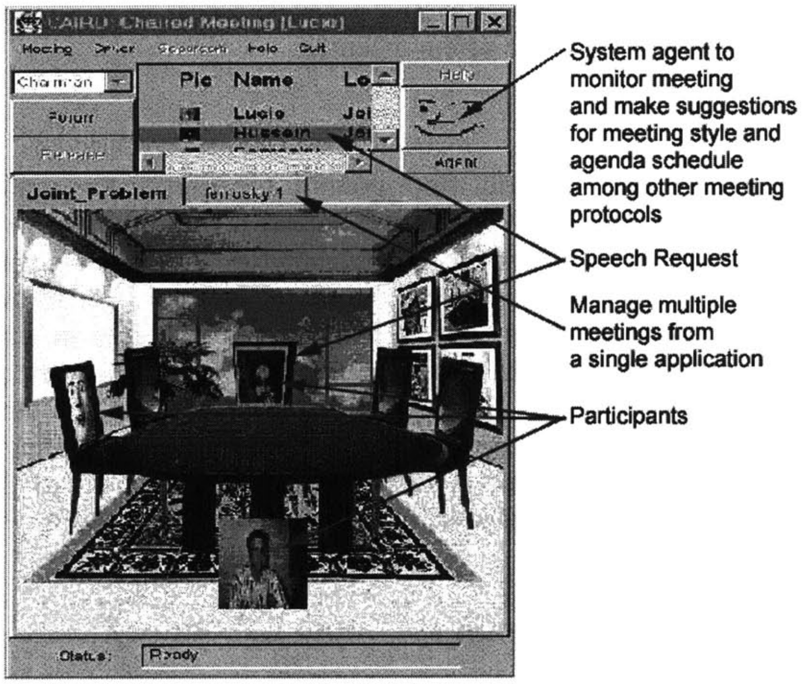

System agent to monitor meeting and make suggestions for meeting style and agenda schedule among other meeting protocols
Speech Request
Manage multiple meetings from a single application
Participants
Pic
Name
Lucie
Hussein
Joint_Problem
Status:
Ready

Figure 3 - CAIRO Meeting Environment (MIT)

1.2.1.1 Meeting Structure

CAIRO meeting environment adopts a strong sense of membership [2]. In addition to the long-term regular participants, there are transient participants involved for a subset of the entire project. These transient participants include people such as guest speakers, experts or consultants. Communication among any participant (regular or transient members) takes place via conversations between all participants or a subset of the participants. Conversations that do not involve all of the meeting participants are termed whispers, modeled after the act of whispering in a physical meeting. As the size of the meeting grows and the number of issues to be resolved privately increases, breakout meetings are formed to further discuss and resolve key issues among a subset of the team.

As finer grain detail is expected, outside participants may be asked to join the breakout meeting based on their area of expertise.

1.2.1.2 Information Policies

Due to security issues such as information confidentiality, information policies are established for each meeting based on a set of meeting protocols, or simple rules of conduct [2]. When participants are involved in many conversations in a single meeting, the collaboration system manages the knowledge and resource transfer within and between them. The meeting environment is also responsible for making information generated in the primary conversation, such as agendas, drawings, and text-chat dialogs, accessible from the private communications. When a participant enters a breakout meeting, certain contexts of the main meeting will move with them, and some will remain private to the parent meeting. For example, a breakout mission-planning meeting may need access to equipment information, outside participants may join for consultation on equipment but these transient members should not have access to personnel information for the mission being discussed in the parent meeting.

1.2.1.3 Meeting Protocols

Within a disaster relief scenario, floor control subordinate to establish protocols could be crucial for a smooth planning meeting. Protocols make meetings more efficient by acting as enablers, requiring tokens for speech, monitoring speech time and supporting preset agendas, allowing members to keep the meeting on track. For instance, consider such a protocol that each participant has to ask permission before he can speak. If a participant chooses to speak without permission they will be breaking the protocols and hence slow the progression of the meeting and potentially divert the direction of the meeting. Protocols usually exist very loosely within a physical meeting because the enforcement of such protocols is left to the meeting participants themselves. When a system like CAIRO [2] is deployed, support of the protocols is handled by the application acting as a neutral third party.

1.2.2 Mission Planning Research at Draper Laboratory

Draper Laboratory [5] conducts researches on online collaboration by focusing on mission-planning applications, especially military mission planning. Since a military environment is usually highly unstable and often involves human life, the focus of the research at Draper has led to some unique requirements like persistence of planning data between connections and the ability for agents and robots in the field to participate.

1.2.2.1 Mission Planning Applications

Two mission-planning applications were developed to support the needs of modern mission planners. They are the Geospatial Application [8] that concentrates on geographical information and the Temporal Synchronization Application [9] that deals with time constraint issues. Both applications were developed independent of any collaboration environment and independent of each other, however both developments used a similar architecture.

1.2.2.2 Draper Application Architecture and Data Persistence

Collaborative environments typically use applications centered on communication between participants. Once non-communication applications are introduced into this environment, persistence is needed to preserve the application data between sessions. These applications center on data: coordinates of a GIS application or time and actions information of a project management application. Consider a military collaboration environment, among geographically distributed participants using an unreliably network for communication. The persistence of the collaboration data would allow participants to rebuild their session from information stored on the repository rather than lose all access to information generated prior to their connection.

1.2.2.3 Multiple Device Interface

As Draper research progressed, it became apparent that many collaborators in a mission-planning environment would be in the field executing the plan as it was being developed. It is impractical to expect participants in the field to have capabilities similar to those in the planning facilities, and isolation of these participants on a separate system, such as two-way radios, would leave them at a great disadvantage and decrease their ability to contribute to the collaboration. Therefore, a goal of the mission-planning objective at Draper is to incorporate multiple hardware devices such as handheld computers and wireless web-enabled devices into the collaboration. Using handheld devices and specialized content based on limitations, field rescuers could stay in sync with mission planning officials, resulting in an efficient collaboration.

1.2.3 Collaboration between MIT and Draper

Draper, recognizing MIT's collaboration experience and desiring to improve their application-centered approach to collaboration, established a joint research initiative in the summer of 1998. At the end of the first year, the Temporal Synchronization Application was integrated into the CAIRO system, making use of the collaborative environment [10]. Collaborators could log into the CAIRO meeting environment, having access to the Temporal application as well as existing ones such as project management, utilizing an existing set of protocols.

Continuing development of both the CAIRO meeting environment and the Draper applications focuses on system architecture for collaboration, integrating a data repository, collaboration protocols and interfaces to support alternative hardware devices. A component-based collaboration infrastructure to support multi-device web-based collaboration sessions has been designed to address such requirements [7]. The system components include a data repository, a collaboration manager and an information policy enforcer to address such constraints. The system can support communication applications like audio or messaging as well as collaborative applications like simulation systems, GIS mapping and mission management. To support such a wide range of applications and provide access to their data from various hardware devices, the design of the system developed is application and configuration neutral for greatest flexibility and applicability.

1.3 Thesis Overview

Chapter 2 provides an overview of historical development of collaboration systems and the technology advances that make collaboration system development possible and give birth to new development possibilities. Chapter 3 focuses on requirement analysis, design and sample implementation of a multi-server collaboration system, specially geared toward disaster relief mission planning, although most of the design and implementation can be extended to other collaboration scenarios without substantial modifications. Chapter 4 switches to three important mechanisms specially designed for multi-server systems: mechanisms for performance and reliability improvement, mechanisms for dynamic and seamless handoff mechanism of clients among multiple servers, and mechanisms for managing heterogeneous client interfaces. Chapter 5 discusses an important mechanism to make virtually any applications collaboration-ready. Server-side computing seems like the suitable solution. Chapter 6 discusses another mechanism to make participants collaboration-savvy and efficient. The basic tool is a self & cross assessment. A web-based Questionnaire Service Provider is designed and implemented for this purpose. Chapter 7 is the conclusion and possible future work. We believe that multi-server collaboration systems should be better than single-server collaboration system, especially in situations where one server is not able to handle all requests, networking between clients and servers are congested or prone to error.

Chapter 2

Collaboration System Development and Technology Advances

This chapter provides an overview of historical development of collaboration systems and the technology advances that make collaboration system development possible. For practical purpose, some detailed information about wireless technology and popular PDAs extensively used in this research are provided.

2.1 Collaboration System Development

Technology driven collaboration systems could enable distributed teams to be more effective by increasing the quality and quantity of the collaboration. The Internet has given virtual teams a global network for communication and allows applications to access data from anywhere in the world. In this section, we retrospect both the historical development and current trends of collaboration systems.

2.1.1 Historical development

Compared to traditional telephone conferencing and video conferencing, such as those from PictureTel and VTEL, collaboration systems had evolved in three dimensions by making full use of digital medias: more distributed, more synchronous, and richer contents.

Place: ***geographically distributed collaboration* vs. *collocated collaboration***

People are limited in their accessibility to information and unable to carry out efficient dislocated collaboration. If they do not use telephone conference or videoconference to carry out a real-time and synchronous collaboration, they basically have to either lower the synchronous degree of their collaboration or sacrifice the information richness during the collaboration. For example, if all participants do not have audio conferencing or videoconferences facilities, they have to use tools like email, which is basically an asynchronous collaboration means. They might also use software solutions like NetMeeting and CAIRO to hold a meeting, but neither of them supports mobility – people have to sit down and link their PCs to the network, and their audio and video dispatching capability is limited – in this sense, the information richness is degraded.

Synchronous degree: ***synchronous collaboration* vs. *asynchronous collaboration***

Generally speaking, collocated meeting is synchronous, because participants sit together and talk face to face. But people are moving more frequently and more distantly than before, and they need synchronous collaboration with dislocated members. Synchronous degree of collaboration should not be compromised because of the distances among collaborators.

Information Richness: ***rich session* vs. *plain format***

During a collaboration session, participants demand rich information exchange like application sharing, video, audio, computerized feeling, and real-time polling, so that the system can fully support an efficient collaboration. Traditional collaboration system only supports information of simple format (like telephone voice, text chat, whiteboard).

Technology: ***digital media* vs. *traditional telephone or video facilities***

Bits transferred by networks are replacing human voice and video transmitted by analog communication lines. The digital network beats analog network by its ubiquitous presence and the ability to transfer virtually any kind of media types. Further, collaboration systems are evolving from desktop-based computers communicating via wired networks to now include telephones and wireless devices. Technologies such as voice over IP (VoIP) and wireless data transmission have enabled smart devices to participate in collaborations [11].

2.1.2 Current Trends

Several academic institutions are investigating specific areas of collaborations and how they might affect a system solution for end-to-end collaboration needs. ADLIB, a joint research effort between Loughborough University and Building Research

Establishment, is developing an agent-based system for collaborative design of light industrial buildings [12]. The Key Centre of Design Computing at the University of Sydney in Australia has studied efficiencies of face-to-face versus online collaboration with respect to architectural design situations [13]. Research at the University of Colorado investigates the use of groupware and workflow within online collaborations [14]. The Center for Integrated Facility Engineering at Stanford University has explored online collaboration within a construction process. During the development, design and construction phases, information can be shared between geographically distributed participants [15].

Commercial companies such as WebEx, Centra and Placeware have developed web-based conferencing environments for corporations and individuals. Their solution supports several basic collaboration features, but generally makes little use of persistence for meeting information and protocols for control of the floor. Each commercial solution employs a 3-tier architecture so as to separate application logic from the client and the data storage. A collaborative engine provides the logic, enabling the communication between participants and transfer of application data [16]. This architecture has enabled web-based solution providers to scale their applications to handle the increase in system load. A similar approach has been followed for the collaborative system developed between MIT and Draper [7].

In addition, recent work on bridge inspection at Carnegie Mellon University has explored the use of handheld computing devices in the field [17]. As demonstrated in that research through the use of a portable computer, a digital camera and a microphone, engineers are able to collect data directly into a format that is suited for later analysis. This work has yet to incorporate the collaboration aspect of tying the field agent to a networked system so that others can share in the information gathered and contribute to its analysis in real-time. Additional work at Carnegie Mellon within the Human Computer Interaction Lab has taken a unique approach to collaboration. Participants are equipped with mobile devices, connected to information resources and other participants, establishing a collaborative session whenever they need help [18].

To date, most research has focused primarily on the quality of communication and the development of new applications, rather than improving the quality of collaboration. Many current tools neglect the inefficiencies of meeting online such as miscommunication and uncontrollable participants.

2.2 Technology Advances

Technology, both hardware and software, develop so fast that collaboration systems should be improved in both functionality and applicable scopes by making full use of some cutting edge technologies.

The widespread proliferation of wireless networks and increasing use of small, portable computers has stimulated mobile or nomadic computing. Also called "anytime/anywhere" computing, mobile computing has increasingly interesting and important applications for business, telecommunications, personal communications, national defense, emergency and disaster management, real-time control systems, remote operations of appliances and Internet access. Recent activity in mobile computing strongly indicates that mobile computers and their wireless communication links will be an integral part of future. Collaboration systems should be improved in both functionality and applicable scopes by making full use of these cutting edge technologies.

2.2.1 Wireless Networking

Wireless networks can be categorized into three types according to its geographical coverage:

- *WAN*: Wide Area Network, which normally covers a city, a nation, or even continental region.
- *LAN*: Local Area Network, which typically serves in a building or close buildings.
- *PAN*: Personal Area Network, which is effective only in a small region, typically inside a building.

2.2.1.1 Wireless WAN

The wireless data industry is still working out the kinks in developing and building data transmission networks and agreeing on standards and protocols. There is more than one competing vision. Experts believe that the growing market for wireless data will support multiple networks and protocols -- and faster speeds.

- **GSM** (**G**lobal **S**ystems for **M**obile Communications) based on TDMA (**T**ime **D**ivision **M**ultiple Access). GSM is a digital cellular or PCS standard used throughout the world and the *de facto* standard in Europe and Asia.
- **CDPD** (**C**ellular **D**igital **P**acket **D**ata). CDPD is a packet data protocol designed to work over AMPS (the original cellular network) or as a protocol for TDMA.
- **CDMA** (**C**ode **D**ivision **M**ultiple **A**ccess), a spread spectrum air interface technology used in some digital cellular, personal communications services and other wireless networks.

Wireless-capable Internet access can be classified into three broad categories:

- **Full user mobility** - Examples include cellular digital packet data (CDPD), in which users can access data while on the move, such as in a car. Yet, Internet access via fully mobile systems occurs at relatively slow speeds.
- **Portable wireless data** - Such networks let users access the Internet while they are in the coverage area, using a laptop or handheld device, and a small wireless modem. But these networks don't offer full mobility during online

sessions. In other words, if a user tries to access the Internet while in a moving car, the connection may drop. Speeds are generally higher than with fully mobile systems but are still relatively slow when compared to today's dial-up modem technology.

- **Fixed wireless data** - This offers service to a location, such as an office or home, through larger customer-premises antennas seen in the mobile or portable setups. The fastest data throughputs -- up to T-1 speed -- are available over fixed wireless networks.

Wireless networks face great challenges. The surrounding environment interacts with the signal, blocking signal paths and introducing noise and echoes. As a result, wireless communication is characterized by lower bandwidths, higher error rates, and more frequent spurious disconnections. These factors can in turn increase communication latency resulting from retransmissions, retransmission time-out delays, error-control protocol processing, and short disconnections. Mobility can also cause wireless connections to be lost or degraded. Users may travel beyond the coverage of network transceivers or enter areas of high interference. Unlike typical wired networks, the number of devices in a network cell varies dynamically, and large concentrations of mobile users, say, at conventions and public events, may overload network capacity.

2.2.1.2 Wireless LAN

In a wireless configuration, an access point is connected to wired LAN or bridging with other access points, and is responsible for communicating with PCs, laptops or devices that have wireless connectivity via radio frequency (RF) technology. Wireless LAN is useful in hotspot areas, like airports, health-care centers, warehouses, and etc.

2.2.1.3 Wireless PAN

Wireless PAN is a short-range wireless network for personal communications. The underlying technology is called Bluetooth, a radio-based technology that enables devices to communicate wirelessly at close range, without the need for direct line of site or additional communications protocols. It has gained much attention in the industry because of its potential ability to let cell phones communicate with PCs and PDAs.

Bluetooth has been in the development stage for more than a year and still faces technological hurdles. For instance, it runs on the same radio frequency as the 802.11 wireless LAN standard, thus interference with wireless LANs is a potential problem. There are safety concerns as well. Members of the airline industry are concerned about Bluetooth's ability to turn on a device automatically when it receives a signal, as this could be potentially dangerous in flight.

2.2.2 Handheld Devices

Currently there are about more than 300 million wireless phones, wireless-capable PDA (**P**ersonal **D**igital **A**ssistants) as well as next generation wireless data phones (smart-phones). They represent a huge opportunity in terms of its sheer size and the new paradigms made possible by pervasive computing technology.

PDAs are characterized by its limited processing power, small RAM capacity, low battery supply, small screen real estate, no input devices, and limited user control. People are normally using their PDAs or cell phones as PIMs (**P**ersonal **I**nformation **M**anager). They can use PDA to access small amount of data via wireless networks, like stock price, weather forecast, email text reading, bank account access, and etc, but these are just contents – little to do with transactions and applications. The best way to access applications is to synchronize PDAs with a desktop or laptop so as to make use of PC's processing power.

All these devices tend to have wireless access, which keeps them connected to the network even from remote locations. As networked computing platforms move from wired PCs or UNIX boxes to wireless connected PDAs, one can expect the PDAs to run more meaningful and sophisticated applications like those on traditional PCs or UNIX machines, as well as networked applications supported by distributed collaboration environment.

Hand-held apparatus can be roughly put into three categories

- PDAs: Windows CE, Palm Pilot, Visor
- Smart Phones: Nokia, Ericsson, Samsung, …
- Two-way pagers: RIM, Skytel

The following table lists current popular PDAs.

Table 2 - Current Market Leaders of PDAs

Model	*Image*	*Spec.*	*Interface and Networking*
Compaq iPaq H3635 Pocket PC		**CPU:** 206 MHZ **Memory:** 32MB **Display:** TFT 4096 bit **Weight:** 6.8 oz	Interface: serial port, IrDA, USB cradle. Networking: iPaq does not have a CompactFlash slot, but it supports Expansion Packs for PC Card and the CompactFlash. The PC Card Expansion Pack enables wireless connection when used with a CDPD card or an IEEE 802.11b HR wireless LAN PC card. The CompactFlash Expansion Pack allows the user to plug in a variety of cards, such as storage, local area network connectivity and other third party options like a modem or barcode scanner.
Casio Cassiopeia E-125		**CPU:** 131 MHZ **Memory:** 32MB **Display:** TFT 65,536 bit **Weight**: 9.0 oz	Interface: serial port, IrDA, USB cradle. Networking: E-115 is the only PDA that has Type II Compact Flash port. Thus CF LAN Card, CF Modem and CF Digital Phone Card for either LAN or WAN access.

Model	*Image*	*Spec.*	*Interface and Networking*
HP Jornada 548		**CPU:** 133 MHZ **Memory:** 32MB **Display:** LCD 4,096 bit **Weight:** 9.1 oz	Interface: serial port, IrDA, USB cradle. Networking: Jornada 548 only supports Type I Compact Flash port. CF LAN Card, CF Modem and CF Digital Phone Card for either LAN or WAN access.
Handspring Visor Deluxe		**CPU:** 16 MHZ **Memory:** 8MB **Display:** monochrome **Weight:** 5.4 oz	Interface: IrDA, USB cradle, Springboard Networking: Visor has a proprietary Springboard expansion slot. Several modems can be snapped into the slot for dial-up via RJ11 or GSM phone. On Sep. 19, Handspring announced a Visorphone, which can turn the gadget into a cell phone.
Palm, Inc. Palm IIIc		**CPU:** 16 MHZ **Memory:** 8MB **Display:** 256 bit **Weight:** 6.0 oz	Interface: IrDA, serial cradle Networking: PalmModem Connectivity kit to dial-up via RJ11. Palm Ethernet Cradle to connec to Ethernet via RJ45.
Palm, Inc. Palm Vx		**CPU:** 16 MHZ **Memory:** 8MB **Display:** grayscale **Weight:** 4.0 oz	Interface: IrDA, serial cradle Networking: PalmV Modem to dial-up via RJ11. OmniSky CDPD wireless modem to access OmniSky.
Palm, Inc. Palm VII		**CPU:** 16 MHZ **Memory:** 2MB **Display:** grayscale **Weight:** 6.7 oz	Interface: IrDA, serial cradle Networking: PalmModem Connectivity kit to dial-up via RJ11. Palm Ethernet Cradle to connec to Ethernet via RJ45. It also has an internal radio transceiver to connect to the BellSouth Wireless network.

Chapter 3

Multi-Server Collaboration Systems

This chapter provides detailed information on requirement analysis, design and sample implementation of a multi-server collaboration system. Although the requirement analysis is specially geared toward disaster relief mission planning, most of the design and implementation should be able to be extended to other collaboration scenarios without substantial modifications.

3.1 Generic Collaboration System

This research presents a conceptual model for a multi-server collaboration infrastructure, which can be deployed in any environment under random configurations, allowing multiple servers, multiple clients and multiple participants to dynamically join, coordinate and quit. Based on the research done at MIT and Draper, as well as the current work in the collaboration field, system requirements for server synchronization, service handoff, data persistence, meeting organization and protocols, as well as multiple device collaborations has been developed.

At first, we briefly describe system requirements for generic multi-server collaboration systems. Then we switch to systems specially designed by disaster relief mission planning. We use a sample case to demonstrate the challenges of a complex collaboration environment and the justification for the system requirements.

As we point out in chapter 1, single-server collaboration system is constrained by non-ideal network conditions and possible bottleneck of CPU processing power of the central server. Multi-server collaboration system is a natural solution to overcome the difficulties posed by the single server. Multiple servers, mobile or deployed in strategically selected locations, are interconnected with fast and stable networks, and synchronized with each other during any collaboration session. Each server has exactly the same snapshot of its data storage and session status. Therefore, each client will get the same services no matter which server it is connecting to.

3.1.1 Applicable Situations

Ideally, a multi-collaboration system should be capable of handling the following three situations, basically including the possible combinations of both fixed and mobile networking conditions:

Wired Network, Fixed Servers

If certain clients are connecting to a server via wired networks, it is probable that network traffic between clients and this server increases gradually as the collaboration session proceeds. Hence, the server may not be able to get requests or send responses fast enough. We say the network connection to this server is suffering from congestion. If there is another server that can serve the same services, the collaboration system can redirect some clients to connect to that server, thereby balancing the working load with multiple servers during the collaboration session. This is different from the load balancing before the collaboration session starts, because we are trying to balance loads during the collaboration session as the circumstances of the network and server conditions change.

In this situation, robust peer-to-peer computing can also be supported, where clients and servers are dynamically switching their roles.

Wireless Network, Fixed Servers

Consider the case where certain clients are connecting to a server via dynamic wireless networks of low bandwidth and/or limited coverage. Even if the server is not heavily loaded, it is possible that some clients cannot send requests to this server or get responses from this server within an acceptable time period. In this case, it is more important that multiple servers be deployed in strategically chosen locations so that a client can be passed to servers that can provide good services once the client moves to the new location. Again, this occurs in real-time as the collaboration session progresses, as opposed to managing the process before the collaboration session starts.

Wireless Network, Mobile Servers

Consider the case where servers and clients are mobile and interconnected with dynamic wireless networks, which are slow, error-prone and have limited coverage. In this situation, multi-server approach proposed in this research is a must for non-stop collaboration services. One client might lose connection with its original server as a result of moving out of wireless network coverage, either because the client is mobile, or because the server is mobile, or both. It is extremely important for this client to be served seamlessly by another nearby server. From a server perspective, each server should be able to synchronize with other servers via wireless network, thus reducing networking traffic by distributing application data to multiple storage places. However, each server must have its own storage so that poor network conditions will not prevent any server from being incapable of keeping up with the collaboration session.

It should be noted that one of the main requirements in this case would be the seamless client handoff while connecting from one server to another with minimal overhead and delay.

3.1.2 Goals of Multi-Server Collaboration System

The most important goal of multi-server collaboration system is its scalability. In a distributed system, growth in numbers of clients and servers can be sustained gracefully, with the advantage that increasing requests from clients will not exhaust existing servers, and new servers can join the system seamlessly.

Local autonomy is another reason for multi-server collaboration system. Since the nature of the collaboration system requires participants and data in geographically areas that are often decentralized, it definitely makes sense to implement a distributed system. In this way, the clients can physically connect to the nearest server, or the server that has the highest bandwidth. This results in local autonomy of the server allowing participants to carry out a collaboration session without resorting to remote servers when network conditions do not allow them to connect to those remote servers.

The other goals of multi-server collaboration system are to provide a series of transparencies that shield the clients of such a system from the complex details of providing those transparencies. In an introduction on distributed databases, the author lists seven transparencies that are what he considers to encapsulate the desired functions of a distributed database system [19]. We adapt four transparencies for a multi-server collaboration system:

- ***Location transparency*** Clients do not know which server they should connect to. The collaboration system decides this based on algorithms using current real-time variables such as network conditions, server loading, and active collaboration services as input. In other words, clients only need to know at least one "introductory" server to join a collaboration session; it does not need to know the full picture of the locations and available servers.
- ***Service transparency*** Each server can provide exactly the same service to whichever clients that are connected to it during a collaboration session. The client does not need to know which services it should request when it joins a session by connecting to any available server. The collaboration system itself should be able to direct the clients to use suitable modules and services.

- *Copy transparency* The servers should synchronize with each other. Application data can be copied, and copies are maintained automatically. If a server is newly booted up, it should be able to replicate the active collaboration sessions from other servers. If a server rejoins the system after being standalone for a while, it should be able to catch up with other servers by synchronization based on incremental updating algorithms.
- ***Fragment transparency*** Application data can be partitioned or fragmented to different servers to get better performance and availability. The partition and re-combination of application data are fully handled by the system, and clients need not know where and how to get the data.

The above goals of multi-server collaboration system are the ideal features. Like in the DDBMS (**D**istributed **D**ata**B**ase **M**anagement **S**ystem) field where researchers are listing ideal features but might not be able to implement them shortly, we hereby bring forward a practical solution for multi-server collaboration systems.

3.1.3 Practical Solution

For a practical multi-server system, data consistency and availability is essential to ensure that each client will get the same service no matter which server it is connecting to. Considering the fact that client typically needs to know what servers can serve it, location transparency is hard to be fulfilled in practical solutions. To achieve this, we introduce a "Location semi-transparency" concept and the corresponding client-server handoff mechanism, in which clients and servers work together to ensure a smooth handoff.

In practical systems, copy transparency and fragment transparency are not compromised. The servers that are always in the collaboration sessions will synchronize with each other during the session. The servers that join a collaboration session will be able to synchronize itself with any active servers. Data fragmentation is automatically handled by the system.

In chapter 4, we will present two mechanisms for practical collaboration systems: Software-Based Handoff Mechanisms (SBHM) for "location semi-transparency" and dynamic handoff, and Performance and Reliability Improvement Mechanism (PRIM) for copy transparency and fragment transparency.

3.1.4 Multi-Server System Architecture

The multi-server collaboration system architecture is a three-tier architecture (as shown in Figure 4), and all components are categorized into three levels: the Client Application level, the Collaboration Engine level and the Application Repository level. Each level performs specific functionality. Notice we are using slightly different terms. The Client Applications are the clients we refer to in the previous chapters, and Collaboration Engines are the servers that provide collaboration services and tools to various clients. Application Repositories are application data we use in the previous chapters.

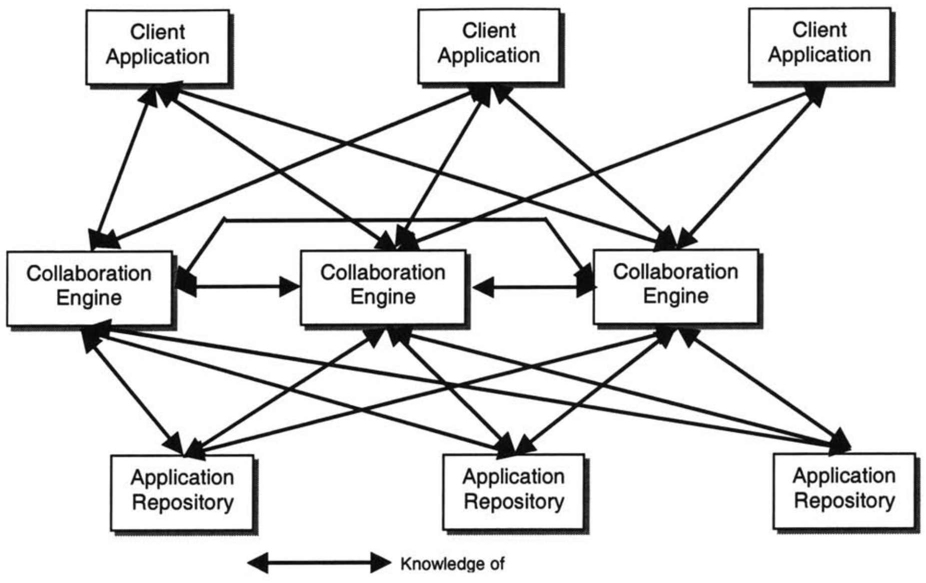


Figure 4 - Multi-Server Collaboration System Architecture

All components are categorized into those three levels based on the Model-View-Controller (MVC) architecture [20] (as shown in Figure 5). In a MVC, the model object knows about all the data that need to be displayed. It also knows about all the operations that can be applied to transform that object. However, it knows nothing whatever about the GUI (**G**raphical **U**ser **I**nterface), the manner in which the data are to be displayed, nor the GUI actions that are used to manipulate the data. The data are accessed and manipulated through methods that are independent of the GUI. The view object refers to the model. It uses the query methods of the model to obtain data from the model and then displays the information. The display can take any form; however, different displays would have no bearing whatsoever on the intrinsic behavior. The controller object knows about the physical means by which users manipulate data within the model. In a GUI for example, the controller object would receive mouse clicks or keyboard input which it would translate into the manipulator method that the model understands. In some situations the controller may interact directly with the view without passing via the model.

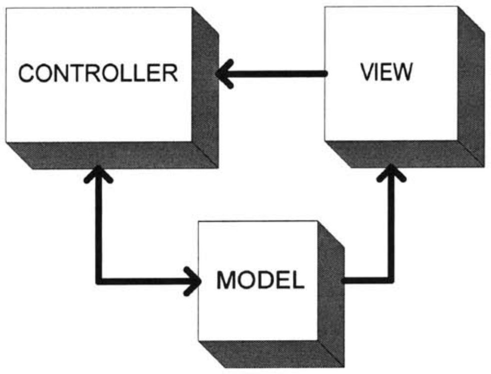


Figure 5 - Model-View-Controller Architecture

Using this architecture, the application data component (the application repository or model) and collaboration service support and logic control (the collaboration engine) are separated from the user interface (the client application or view component running in clients and devices).

Each client application consists of the graphical user interface with minimum computational work and knows information of all the collaboration engines and the network topologies ("location semi-transparency"). They search the network for the application services from the collaboration engine that serves them best. After successfully join a collaboration session, it sends requests and receives responses, and displays collaboration service information, like client lists, text chat, whiteboard drawing, shared applications, video and audio in the user interface.

The collaboration engines act as the controllers in the system architecture by providing the following functionality:

- Provide collaborative services to clients and devices
- Monitor all the clients within each service coverage;
- Maintain the network topology and routing;
- Represent the channels through which the user interacts with the model and control the level of access to information in the application repository.

The collaboration engines acknowledge each other in the network to handle the service transfer for all clients based on the network conditions, client position and quality of service. They maintain the persistency of the information in application repositories and configure the application repositories according to the service requirements and condition of the distributed environment.

During a typical collaboration session, a client joins the collaboration session by first contacting a collaboration engine and sending an "attempt-to-join" message. The system next tries to search for the best collaboration engine according to an algorithm based on the geographical distance, network topology, current network traffic and server availability. The target engine information is sent back to the client. The corresponding collaboration engine will receive request from the client, configure the application repository and other system parameters, and begin to serve the client as the primary engine. In addition, it also notifies all other collaboration engines of this collaboration session and client information. This allows engines to monitor the client's status and movement. Once the working client moves out of the service coverage of the primary collaboration engine or the collaboration engine becomes out of service, the primary collaboration engine will transfer seamlessly the service to the new collaboration engine. This ensures continuity of the service and consistency of the information in the collaboration session. The above process is conceptually illustrated in Figure 6.

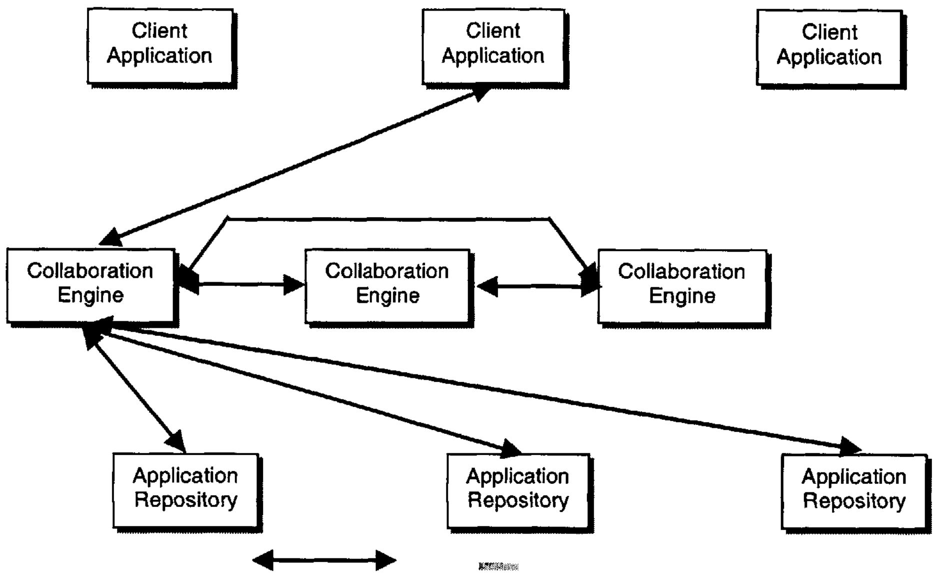


Figure 6 - Multi-Server Collaboration Environment During a Collaboration Session

3.2 Collaboration System Requirement for Disaster Relief

In this section, we will focus on the system requirement for disaster relief collaboration systems. As we point out in the first chapter, disaster relief has some special requirements, and needs to address some special issues, especially dynamic wireless networking and mobile collaboration. This research uses a recent earthquake in India as a case study to identify requirements for a disaster relief collaboration system.

3.2.1 Sample Case: Earthquake in India

A major earthquake occurred in Gujarat, India about 65 miles (110 km) north-northeast of Jamnagar, India or about 180 miles (290 km) southeast of Hyderabad, Pakistan at 8:16 PM MST, Jan 25, 2001 (Jan 26 at 8:46 AM local time in India). The earthquake was felt throughout northwest India and much of Pakistan. Also felt in western Nepal and Bangladesh. 18,602 people confirmed dead, 166,836 injured and 600,000 homeless. 332,000 houses destroyed, 751,000 damaged. 20,000 cattle were dead. Damage estimates are $1.3 billion US dollars [21].

Following the quake, emergency crews formed from around the world, to begin the rescue and relief efforts. Many crews were composed of professionals, those who performed rescue operations as a profession, and citizens, those who volunteered their time and effort. This mix of people caused the rescue management efforts to be more difficult. The extent of the damage covered a large area, requiring a geographical distribution of the rescue efforts. In addition, as a result of the quake many lines of communication were severed, forcing rescuers to use more flexible communication solutions.

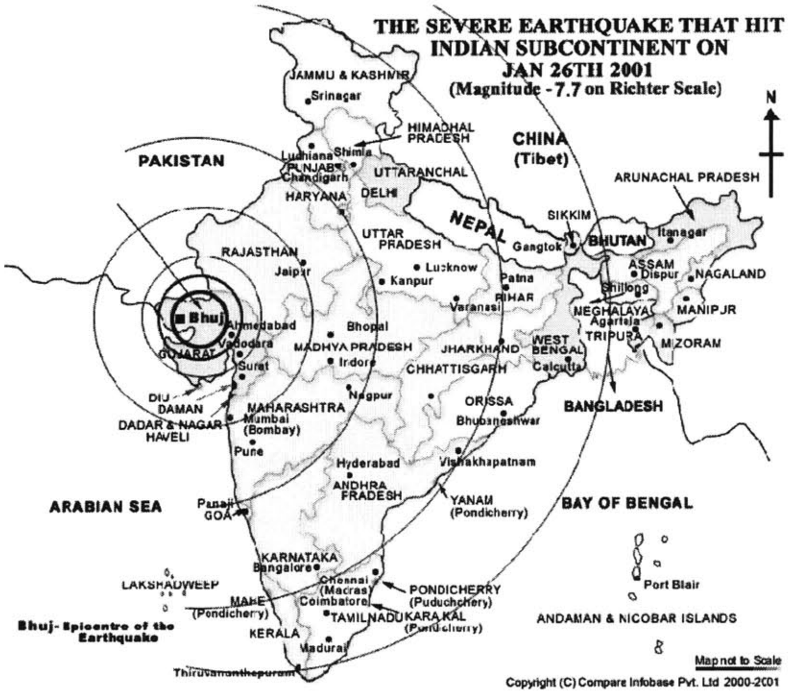

THE SEVERE EARTHQUAKE THAT HIT INDIAN SUBCONTINENT ON JAN 26TH 2001 (Magnitude - 7.7 on Richter Scale)
N
PAKISTAN
CHINA (Tibet)
NEPAL
BHUTAN
BANGLADESH
ARABIAN SEA
BAY OF BENGAL
Bhuj
Bhuj- Epicentre of the Earthquake
ANDAMAN & NICOBAR ISLANDS
Map not to Scale
Copyright (C) Compare Infobase Pvt. Ltd 2000-2001

Figure 7 - Epicenter of India Earthquake

3.2.1.1 People and Parties Involved

Although India is traditionally reluctant to ask for outside help and will want to demonstrate that it can and wants to cope, relief agencies from all corners of the world still flied to India and worked together to allocate resources and direct efforts to specific tasks for the earthquake relief [22]. Volunteers aided professional organizations, such as fire fighting men, policemen, and professional relief workers. However, lack of organization and coordination has strained those relief efforts. According to a spokesman of the International Red Cross, order collapsed outside Bhuj Tuesday and many complained of not having food or water [23]. The involvement of so many groups addressing both immediate concerns, such as rescue of victims, and long-term concerns, such as disease and famine or structural integrity of infrastructure, demanded a hierarchical organizational structure to allow information dissemination such as direction of medical personnel and allocation of equipment. Within the complex organization of the disaster relief environment, there are several issues including the lack training and coordination that could be addressed in real time with a structured collaborative environment. Such an environment would enable several organizations to work together to solve larger problems by laying out the problem and developing a collaborative session structure to meet the requirements. For example, relief efforts could be categorized and corresponding collaborative sessions developed to resolve issues that are concerned with a particular aspect of the operation.

3.2.1.2 Dispersed Geographical Locations

The disaster relief environment is further complicated due to the fact that the parties involved are also geographically dispersed, as in the earthquake case. As rescue crews dispersed among the fallen buildings throughout the large disaster area, their constant movement to quickly locate survivors required the emergency resources to be available simultaneously in various areas. Collaborations among these individuals would have to be dynamic and loosely organized to allow them to do their job effectively in the field and to keep rescue coordinators informed. An open collaborative environment that would support both wired and wireless networking connections, multiple hardware devices, allowing communication between collaborators independent of their physical device capabilities and networking connection types is a necessity.

3.2.1.3 Device and Network Limitations

Field personnel and mission control for planning and rescue missions such as earthquake disaster relief efforts may be forced to depend on radio or wireless communications technology to remain in contact with one another. Field agents typically only had the device used for communication while mission control personnel had access to more sophisticated equipment. Instructions and guidance from mission control and feedback from the field is critical to keep everyone informed and complete a successful mission, but is limited by the devices used by field agents. A small device could serve multiple purposes, providing a communication link to the mission control and enable field workers access to relief related information such as maps and equipment operating instructions. Devices such as PDAs, web-enabled phones and two-way pagers as well as more specialized devices have unique constraints on the type and amount of information that can be displayed. Small devices, with specialized content would enable field agents to accurately and effectively participate in a collaboration session, transmitting information to the session and receiving from the session, typically in the form of directives.

3.2.2 Requirements Analysis

Based on the presented case study, requirements for the collaborative environment can now be detailed [7]. The system will need to provide a structured environment for each collaborative session and information about who the collaborators are. An information policy will be established to determine how information flows and the proper channels for communication. Persistence of collaboration data will be necessary as well as the need for multiple hardware devices to access the system. A mechanism for server synchronization is needed to support coordination and synchronization among multiple servers. A mechanism for software-based handoff can be employed to make clients seamlessly transfer to a new server for non-stop collaboration services.

3.2.2.1 Structured Collaborative Environment

As presented, the search and rescue operation for the India earthquake involved several parties in a distributed environment where multiple types of collaborations will take place using various devices. To solve the problems of confusion and disorganization among collaboration participants, they are placed in a structured environment where policies can be set and enforced to avoid chaos and/or anarchy. By guiding how the participants interact, the system can help in maintaining control of the collaboration, allowing the leaders of the session to be more effective in coordinating resources.

Collaboration Session

A collaboration session is any grouping of participants for the purpose of working and communicating with one another. Sessions take the form of meetings that involve presentations or intense planning where ideas are developed and explored. The session can last for any length of time and can also be stopped and resumed at a later date if the session has not concluded. Session should also be recorded for documentary purpose, and be able to be played back for various purposes, such as staff training.

Session Organization and Side Sessions

A collaboration session may exist within a complex arrangement of other sessions. Sessions may exist in a hierarchical fashion, resembling an organization structure. Collaboration sessions may spawn side sessions that are separate, with their own rules and resources, but still belong to the main collaboration session and retain some access to its resources.

Conversations and Side Conversations

A conversation is any communication between collaboration participants. The conversation is the channel through which information is conveyed to others. Whispers, or side conversations are a particular kind of conversation that contains only a subset of the participants involved in the entire collaborative effort. Communication between all collaborators will certainly take place, sharing information and directing resources. Additionally, some of the collaborators may have questions concerning an order. These cases can be handled in side conversations, involving only the parties that are necessary.

Protocols

Protocols are the rules of collaboration sessions, their organization and the conversations within each session. Protocols limit the flow of information, control the number of participants speaking at the same time and maintain a level of control. The protocols of a meeting define the process by which a participant is able to communicate with others often requiring permission of a session leader. Protocols act at all levels, from individual participant requests to side session requests for access to main session information. Protocols are effective in situations involving participants that do not know each other. In a search and rescue operation or other quick formation collaboration, there is no time to build relationships, forcing participants who have never met to immediately trust each other and recognize orders given by supervisors. By forcing the participants to abide by rules, authority can be established and maintained, allowing a single participant to control a collaborative session involving participants they have no previous relationship with.

Representation of Collaborators

Within any collaboration environment, membership is necessary, enabling individual participants to know who is talking and who is receiving or providing information within an often-faceless collaboration. Collaboration sessions involving participants that have never met introduce issues of trust and acceptance. Visual representation and contextual background information help participants understand each person's position in the session and their credentials on an as-needed basis. Knowing who is speaking and their qualifications greatly improves interactions among participants, which justifies protocol enforcement and establishment of information policies.

Information Policy and Protocol Enforcement

Between sessions and conversations, knowledge and resources need to be transferred to ensure current information is being used. A field rescue collaboration session is at a great disadvantage if it cannot access information generated in a resource management session that is responsible for directing field agents. In a hierarchical, multi-session environment, information dissemination and encapsulation are of key importance. During a single collaboration session, an unknown number of applications are used, requiring the information policy to be independent of any application. The information policy can be implemented in a central location, acting as the interface between the participants and the collaboration session. By removing the policy implementation from the client, control can be centralized.

3.2.2.2 Persistence of Collaboration Data

Information generated during one session needs to be available to other sessions, during and after. Persistent storage of all information generated during a collaboration session maintains consistency among participants and can be used later for detailed inspection of collaboration sessions. Training exercises and facility updates directly benefit from the examination of historical information, placing an additional requirement for persistency of collaboration information.

Consistent Representation Between Collaborators

Utilizing a common repository for all transactions, collaborators can be assured that their session is up to date with others. Each transaction affects a single source, enabling latecomers to build the session to the current state or retrieve the current state of the session in a single request. During loss of service, participants can store transactions locally, executing them once a connection to the session is re-established.

Session Replay and Analysis

Since all transactions are executed on a repository of information, the transactions themselves can be stored. Transaction logs can be replayed at a later date for study of a collaboration session or to search for a particular piece of information. Replay features are very important for training sessions and can be invaluable for studying the performance of participants in a particular collaboration session. High-level collaboration sessions can also study lower sessions in real time, monitoring progress and providing feedback.

3.2.2.3 Multiple Device Collaboration

With a repository of information in place to centrally collaborate around, the collaborative environment can incorporate various technologies into the collaboration. Imagine a rescue mission operation where the field agents were completely out of communication with anyone else. Chaos would ensue and resource management would be much more complex. Situations involving collaborators whose position cannot be fixed require mobile devices that move with them. The fixed unit will still be preferable, since it gives such participants greater functionality and a better interface, however the collaboration system must support both mobile and fixed units. As an illustration for each interface, two prototype applications will be presented.

Desktop Collaborators

The core of any Internet enabled system is the desktop solution: a device that is fixed in position for a given session. Multi-device functionality exists within this realm, as there are many desktop devices that must communicate. Portability from platform to platform is key in multi-user, multi-session collaborations, where participants are expected to join randomly with minimum hardware requirements. Desktop solutions need to be designed to allow devices of all configurations and capabilities to collaborate. From the military mission-planning research at Draper Laboratory, the desktop prototypes for the Geospatial and Temporal Synchronization Applications are show in Figure 8. These applications are typical desktop computer based applications running on a Windows operating system.

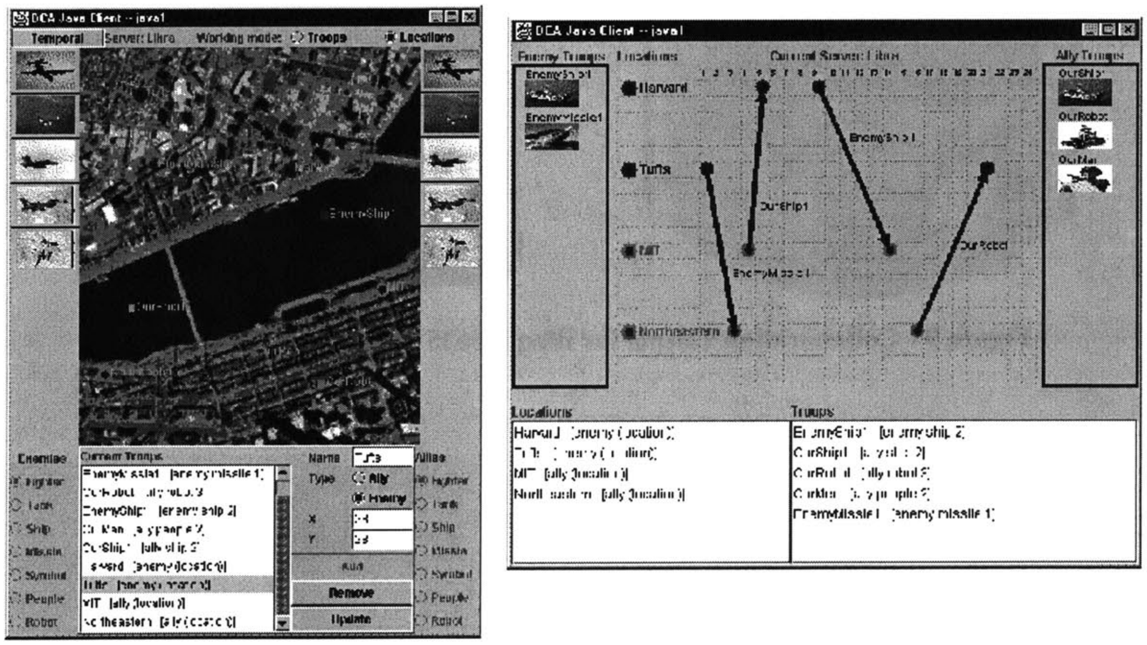


Figure 8 - Collaboration Clients Running on Desktop and Laptop Computers

Handheld/Pocket/Phone/Pager Collaborators

As we point out in the second chapter, there are a couple of PDAs, phones and pagers with different wireless networking. Wireless LAN networks have limited range and may not be feasible in disaster areas, such as the earthquake scenario presented earlier. Handheld and pocket devices such as Palm Pilots, Pocket PCs and cellular phones have wireless access and are substantially more portable and versatile than the devices available for wireless LAN networked computers. Most Palm Pilots, Pocket PCs and cellular phones are now enabled with direct Internet access. With those devices, information is accessible from the field and updates to current session information can be made, utilizing technology to keep those in command up to date with information. As shown in Figure 9, the two collaboration clients for military planning are running on Compaq iPaq H2635 and CASIO Cassiopeia E-125. Similarly, Figure 10 shows the same collaboration clients running on Java-enabled phones and pagers. These four clients are active collaborating in the same collaboration session.

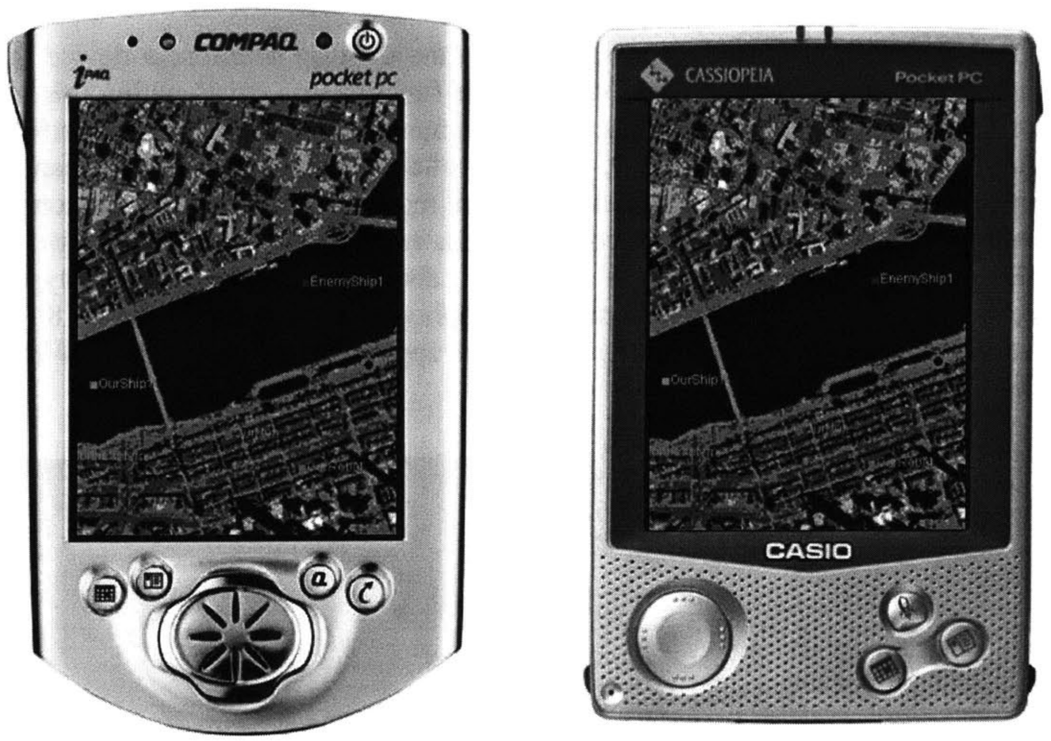


Figure 9 - Collaboration Clients for iPaq H3635 and Cassiopeia E-125

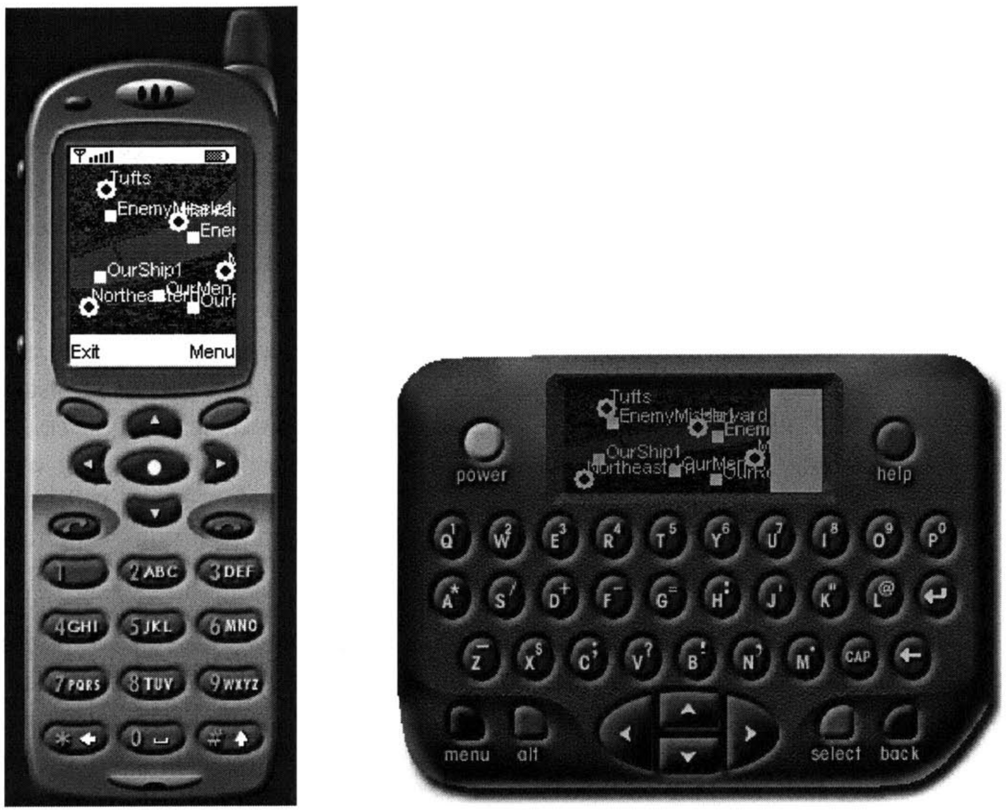


Figure 10 - Collaboration Clients for Java-Enabled Phone and Pager

For disaster relief, it is highly probable that networking and telephone lines are taken down in a disaster area; however, collaboration could continue by deploying a vehicle with a portable wireless antenna to facilitate wireless communication between collaborators. In Chapter 3, we will present the software-based handoff mechanisms for dynamic networked servers and client. Furthermore, although the display capabilities of the devices are limited, collaborators can still access the session and be active participants. In Chapter 5, we will present the core technology to enable PDAs, phones and pagers to actively collaborate by running shared applications in the collaboration engines.

3.2.2.4 Multiple Server Synchronization

The collaboration services are provided by multiple servers deployed in strategically chosen locations, and every client will get the same service no matter which server it is connecting to. Since clients are unable to keep track of the fast-changing collaboration environment, the servers should be able to synchronize with each other, and keep up with the up-to-date version of the following information:

- Static information for disaster relief, like geographical and demographic information of the disaster area, relief resources available, knowledge about victim identification, excavation, and first-aid.

- Static information for collaboration system, like system members information, lists of servers with its location, capability, coverage, and availability, lists of clients and participants with their capability and availability.
- Dynamic information for collaboration sessions, like current undergoing collaboration sessions, session schedules and agendas, active conversations and side conversations in each session, and participating clients and devices for each session.
- Volatile information for each collaboration session, like the current chats, ongoing whiteboard activities, real-time polling and results statistics, active shard applications, audio and video being broadcast. Notice the word "volatile" does not necessarily mean all the above information will be lost afterwards; rather, it is used because these kinds of information are changing fast and need special attention when designing synchronization schemas and algorithms.

Synchronization schemas basically fall into two categories: full replication and incremental replication. Full replication means that a server builds a mirror to another server by dumping all the available information, while incremental replication means that a server first compares its own content with the target server, then do the necessary updating, adding and deleting so that the two servers will maintain the same contents. It is easy to find that full replication is easy to implement but not efficient, and incremental replication is more efficient but difficult to design and implement.

The choice of synchronization schema is contingent upon server status, information volatility and networking capability. For example, when a server is newly booted up, it has to synchronize with other servers by full duplication; however, if another server rejoins a session after being disconnected for a while, it can be synchronized by incremental duplication. If the information to be synchronized is highly volatile, incremental replication is a must. If a server only has low-bandwidth networking capability, incremental replication should be considered whenever possible.

3.2.2.5 Seamless Service Handoff

When a field agent is carrying a wireless PDA or phone and moving inside a building or building block, he might lose connection to his original collaboration server, either because the signal from the server is blocked, or because the agent is our of the wireless coverage. In order to provide non-stop collaboration services, the agent should be able to connect to another nearby server. Therefore, the collaboration system should have a handoff mechanism so that the client can be redirected to the nearby server before the client loses connection to the original server.

applications. The collaboration manager is designed as a separate component to allow sessions the option of using it or not, resulting in synchronous and asynchronous respectively. Separation of real-time (synchronous) from asynchronous collaboration also enables the system to deliver content to various devices, as some will have limitations that prohibit synchronous communication. The collaboration manager is also responsible for synchronizing with each other during an active collaboration session.

Each component is defined as an abstract specification allowing custom implementations to work within the system. This is important for flexibility, as no system can be designed to handle all possible cases. As a new application is introduced, or new functionality is needed, new components can be developed for a particular implementation.

3.3.1.1 Client Application

The client application is designed to be open and flexible, accommodating new and existing applications. Requirements placed on applications are minimum, allowing legacy applications to be incorporated into a collaborative environment quickly and easily. Two basic requirements are placed on client applications, both of which are optional: the input to the collaboration system and the output. The application can be designed so it can function properly in any configuration: stand-alone mode without any collaboration, input only, output only or both input and output. A sample configuration of a client application is shown in Figure 12.

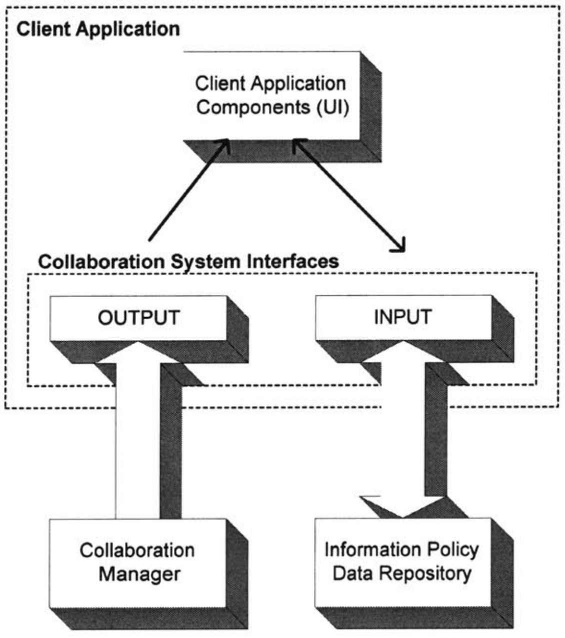

Client Application
Client Application Components (UI)
Collaboration System Interfaces
OUTPUT
INPUT
Collaboration Manager
Information Policy Data Repository

Figure 12 - Sample Client Application Configurations

The input component is the hook into the information policy component that controls the collaborator's interaction with the system. Through this interface, the client sends all transactions to the repository (changes to the application data) and requests

3.3 Solution Details

With detailed requirements in place, a robust, scalable and portable solution can be detailed. The best way to accommodate various configurations and portability is to split the system into modular components that integrate to form the system. Thus, the design for the collaboration system is broken into components performing individual tasks such as storage of session information or managing the collaboration. The components can be deployed in various configurations, adapting to the current demands of the collaboration and the available resources. The system design incorporates the implementation of protocol enforcement engine, data repository, and collaboration engine and client application requirements. To address the needs of different collaborative applications, the system is designed to be scalable and portable, working in varying conditions and able to change with future developments and further research.

3.3.1 System Architecture

Four components with targeted functionality are necessary to give the collaboration system maximum flexibility and scalability. The four components are: the client application, the information policy, the data repository and the collaboration manager (see Figure 11). Separating each component, one can be modified with limited effect to the others. Additionally, the interconnection of system components is flexible so when one component fails, another instance of that component can be substituted with little effort.

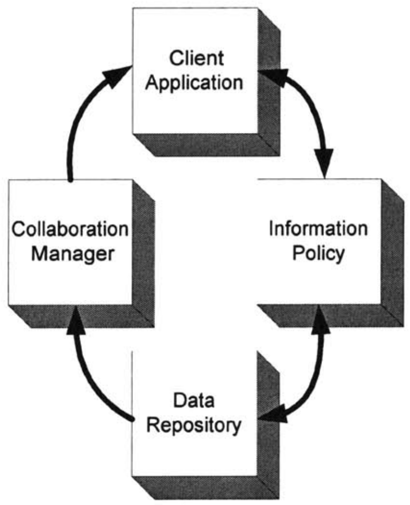


Figure 11 - Four Components in the Collaboration System

The client application is the interface for the collaborators to the system such as a CAD tool or a collaborative mission planning application. The information policy serves as the filter for transactions made by the client applications, enforcing protocols at all levels to everyone in the session. The data repository is responsible for storage of collaboration data, such as who is in the session and any application specific data. The collaboration manager drives the synchronicity of the system, connecting client permission changes. Responses to requests and transactions are fed back into the input component, alerting the client of errors or rejected requests.

When changes are made to the repository, the collaboration system sends notifications to the collaborators via the output component of the client application. Registering with the collaboration manager, the output component is notified when information changes within the confines of the collaboration session. Once notified, the client application must then decide what to do with the information.

The exact path for a single transaction depends on the desired functionality and implementation of the application. By design, a request first enters the repository and then notification is sent to all clients, including the client that made the transaction. Using this circular path, the client application is never made to believe that the transaction took place until the system sends notification.

As mentioned, each application can implement one or both of the components described above. Examples of clients that implement the input component are data entry applications: clients that enter information into the collaboration session but need no notification when others enter information. Such clients may be agents collecting information in the field. Synchronous participation in the collaboration is not a requirement for these clients, however the information gathered might be useful to others in the session that are implementing the output component of the client application. Other applications implement only the output component, for example a "view" type application. These applications may be in the form of mapping applications, displaying locations of field agents on a projection screen. No input to the collaboration session is allowed though the display device, but access to the information and the powerful display capabilities are necessary to collaborators.

3.3.1.2 Information Policy

The information policy component enforces the protocols chosen for the collaboration session. Every collaboration session operates under a specified set of protocols that are changeable at any time during the session. Protocols are determined through parameters such as number of speakers, how long someone can hold the floor and whether or not the session has a chairperson; existing configurations for protocols have been developed at MIT [2].

The information policy component enforces access to information through the parameters of such protocols. The information policy is not responsible for providing access to the data repository, but is responsible for limiting access. For this reason, the information policy is tightly integrated with the data repository. Physically the collaboration manager and information policy work together as the collaboration engine.

The information policy works with a session repository that stores session information associated with any collaboration similar to the structure shown in Figure 13. In addition, the information policy must exist so that any application can utilize the policy to enforce the protocols uniformly across all applications. The placement of the information policy, including a customized information policy component used for a particular application, is shown in Figure 14. The information policy exists at an abstract level to be defined by a particular implementation, allowing multiple information policies to be developed for a single system.

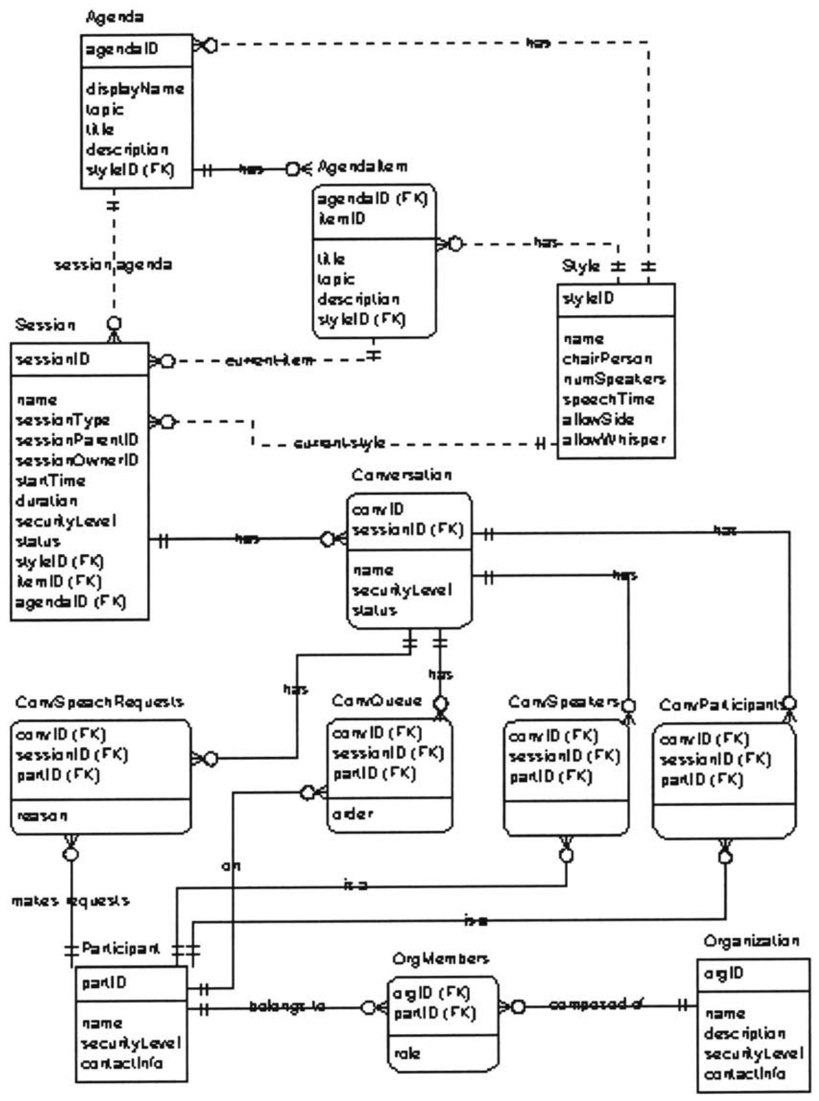


Figure 13 - Database Structure for Collaboration Session Information

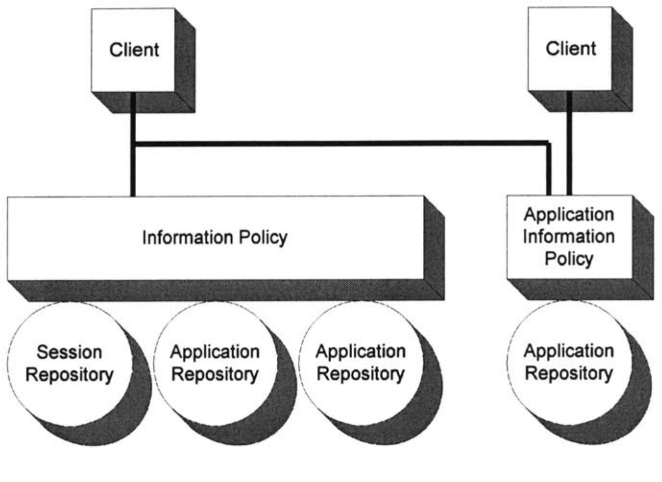


Figure 14 - Information Policy Implementation

When a client application sends a transaction, the information policy determines if the client has the appropriate permissions to process the transaction, then passes the transaction on to the appropriate data repository or refuses the request and suggests alternatives to the client. When clients change their position in the session (speaking, whispering or requesting additional permissions), the information policy changes the state of the client on the system. A session repository is necessary to ensure the collaboration information is stored and logged. As new members enter the system, they will need access to information policy related information such as who is in the session, who is currently speaking and most importantly, and who is in charge of the session. Utilizing a repository, the information policy component is able to make use of the collaboration manager component to notify clients when participants enter, leave or change status in the session.

3.3.1.3 Data Repository

The data repository supports all aspects of the collaboration system. Although designed as a single component, the data repository is actually a set of loosely organized databases. Client applications access their repository through an interface, or repository manager, that directs requests to the appropriate data repository. The information policy component sends approved transactions to the manager to determine which repository to send the request to. Figure 15 shows the placement of the repository manager and the paths for transactions. Note that some transactions bypass the information policy and the repository manager, making direct calls to the repository. This flexibility is designed to allow client applications to bypass all layers for some transactions. For example, a system application may be used to mine data stored on a particular repository, however it has no need to use the collaborative features and can therefore bypass the system.

The implementation of the repository is assumed to be a database system. Database systems have higher performance and greater data persistency than other solutions, such as flat-files. Relational databases are web-enabled and include extensibility through embedded development languages like Java. Most object-oriented data types can be mapped to a relational database for persistency or can make use of an object-relational or object-oriented database. Separation of the data repository from the client access point allows databases to be swapped during a session and multiple databases to work together.

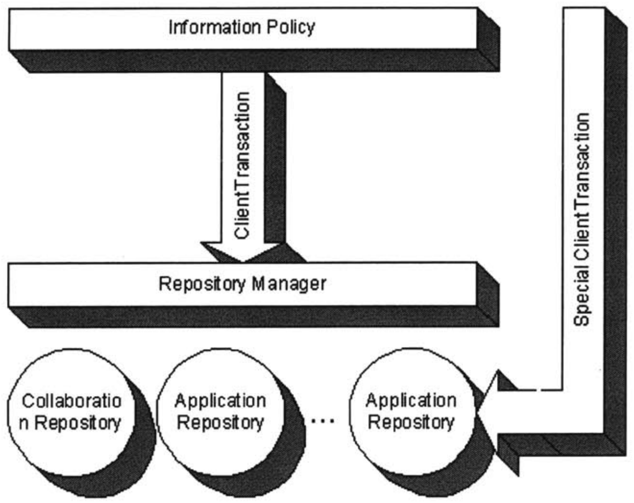


Figure 15 - Repository Manager and Data Repository

To accommodate latecomers, the data repository stores information for at least the duration of the collaboration session. When a participant joins a session late, it is rebuilt to the current state based on information in the repository. It is required to offer the option of persistency beyond the session for replay purposes, however some sessions choose to be "off the record". These collaborations are not stored on the database. Reasoning behind this may be security or uncertainty of the validity of the data. An "off the record" session will operate similarly to the normal session. However, replay of the collaboration at a later time will not be possible. The data repository also provides the means for long-term storage of session information within the system.

The challenge of data storage can be addressed by using a commercially available solution, or by creating a custom system. Many relational database systems provide a scalable, robust solution to the collaboration system. Using an existing data storage solution leverages the power of third party products and expertise. Most commercially available database systems provide access via many development languages, such as ODBC, Java JDBC and C++.

3.3.1.4 Collaboration Manager

Once a message arrives in the collaboration manager, it is sent to the appropriate client. The collaboration manager component is responsible for handling a single collaboration session; therefore, the whole system has several managers. Each session runs on its own manager to isolate itself and minimize security risks and fatal errors. If one session crashes on the system, others should not be affected. Within each manager component is the essential information to efficiently run a synchronous collaboration including participant and session properties. Within the collaboration manager, there is a collection of participants in the session who wish to receive notification when changes are made to the repository. When a new participant comes online and attempts to register with the collaboration manager, a check is done with the data repository to ensure the participant is allowed in a particular collaboration session. The collaboration manager also maintains a list of open conversations and their participants.

As a new message enter the collaboration manager from the repository, it is passed through various filters to determine the type of message and target conversation or participant. Determining who gets the message within the server minimizes network traffic and the load required by the client application. Some messages will need to be sent to all participants and all conversations, therefore, each collaboration session has a single conversation containing all participants. A message entering this conversation is sent to all participants registered with the collaboration manager. Some participants may not be actively monitoring the session conversation, requiring a session wide event to be sent to all participants regardless of the conversation they are actively monitoring to ensure successful message delivery. As an example, consider a side conversation involving two mission planners in a large collaboration session. While in the side conversation, an important decision is presented, requiring a response from everyone. The planners in the side conversation may not be monitoring the main conversation and may slow down the session unless they are notified. Sending the event message to all conversations allows the system to alert the side conversation to the important topic underway in the main conversation. They can then choose to ignore the event, end their side conversation, or just pause it for the time being.

The collaboration managers are also responsible for synchronizing with each other during an active collaboration sessions. Each collaboration manager maintains a list of the available collaboration managers. If the whole system is under stress and needs new collaboration engines, or any client goes out of range of the current system and needs to activate nearby collaboration engines, the collaboration manager of the new collaboration engine will register to the system, and every existing collaboration managers will be notified of the new engine. Similarly, whenever a collaboration manager leaves a session either because of a decent exit or because of timeout, the system will mark it as unavailable.

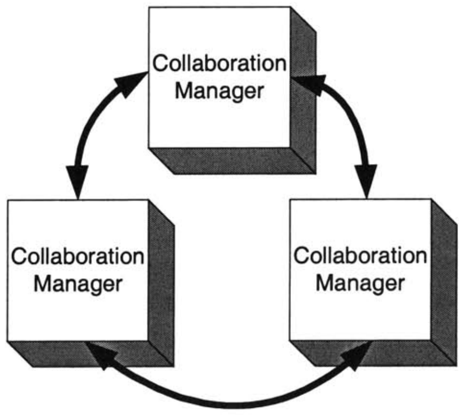


Figure 16 – Synchronization among Collaboration Managers

As the status of each collaboration manager changes, its corresponding data repository will be synchronized as needed. As pointed out in the requirement analysis part, static information, dynamic information, and volatile information will be replicated or updated. Synchronization schemas fall into two categories: full replication and incremental replication. The choice of synchronization schema is contingent upon server status, information volatility and networking capability. Figure 16 illustrates the loose synchronization schemas among three collaboration managers.

3.3.1.5 Custom Components

To address the issues of scalability and future requirements, custom implementations of any component can be plugged into the system. Customizable components can perform specific functionality not in the existing system allowing them to work with other components at the interface level. In addition, custom collaboration managers can work in systems with default managers to handle the same collaboration session in different capacities. For example, consider a client application that requires secure communication on a wireless network using encryption through a special operations planning application using direct networking instead of the Internet (see Figure 17). This would require a custom collaboration manager capable of encrypting the message and sending it over the alternative network. Moreover, custom client applications may work with standard implementations of the data repository and information policy, requiring only the custom collaboration manager. Another feature of the system shown in Figure 17 is the inclusion of the default collaboration manager, which serves clients on the Intranet or Internet while the custom manager serves those agents whose transmissions must be encrypted.

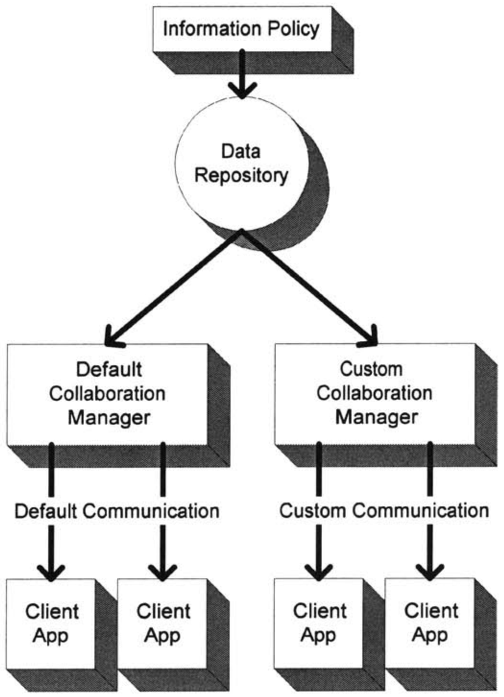


Figure 17 - Custom Implementation of System Components

3.3.2 Technology Details

To implement each of the components, various technologies were used. Communication via the Internet can be through many channels. Some technologies such as CORBA [24], allow advanced programming techniques to be used such as object-oriented design while others rely on simple messages of byte-data from point to point. The messaging system that the collaboration system uses for information dissemination is of key importance and has only begun to be addressed by industry. As the system incorporates new devices, new technology quite different than typical desktop computers will be used to achieve the same functionality, as these new devices will have greatly different capabilities. Thus the design of the system presented in this paper takes into consideration future requirements by using industry standards and flexible technologies designed for integration with new developments.

3.3.2.1 Networking Technologies

For the prototype solution, various networking technologies were investigated. The primary development language was the Java programming language that included three ways of communication via the Internet: socket messaging [25], Remote Method Invocation (RMI) [26] and CORBA [24]. Socket communication which consists of sending an array of bytes to a specific computer port, was abandoned based on previous problems encountered at MIT such as unreliable transactions [2]. Java RMI allows Java objects to communicate via the Internet as though they were on the same computer [26]. The primary limitation of RMI is that it only works with objects written in Java, removing the possibility for client applications written in other languages to utilize the collaboration environment. Finally, CORBA (Common Object Request Broker Architecture) was chosen. As specified by the Object Management Group, CORBA [24] defines an open networking solution that allows objects written in different languages to communicate via a network as thought they were written in the same language and running on the same machine. The Java implementation of CORBA allows object-oriented principles, such as inheritance to be used, giving the system full scalability. CORBA is primarily used in the collaboration manager component of the system for distribution of information to the participants in a particular collaboration session.

3.3.2.2 Messaging System

Alerting participants to information that has changed within the session is the job of an event distribution, or messaging system. Participants register with the collaboration manager for events they are interested in and the manager will in turn alert them when those events take place. As noted by Piehler [27], the publish-subscribe mechanism can be modified to create a system that is flexible and scalable and can handle simple and complex situations. Platt in [28] discussed the Microsoft solution to event notification using COM (Component Object Model). This solution involves proprietary technology that will only work with Microsoft software, eliminating alternate operating systems and devices. Java's JINI technology includes a specification for distributed event notification based on its RMI technology and the built in event system in the Java language [29]. Currently the JINI Distributed Event specification is not implemented, existing only as a specification, however the design of JINI proves promising for systems that include various hardware devices operating with one another in various network configurations [30]. Java has also introduced a new standard for RMI and CORBA, allowing objects of each type to communicate seamlessly via the IIOP channel CORBA uses. This technology proved to be the most versatile, and was chosen to enable a client application written in any CORBA compliant language to communicate with the system that is primarily written in the Java programming language. The final solution made use of concepts of Piehler's model over a CORBA network (Figure 18). Piehler's model was transformed to mimic the structure of a collaboration involving sessions and conversations within the session.

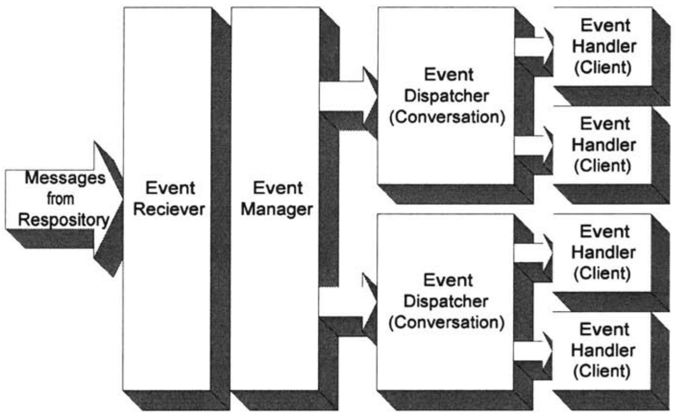
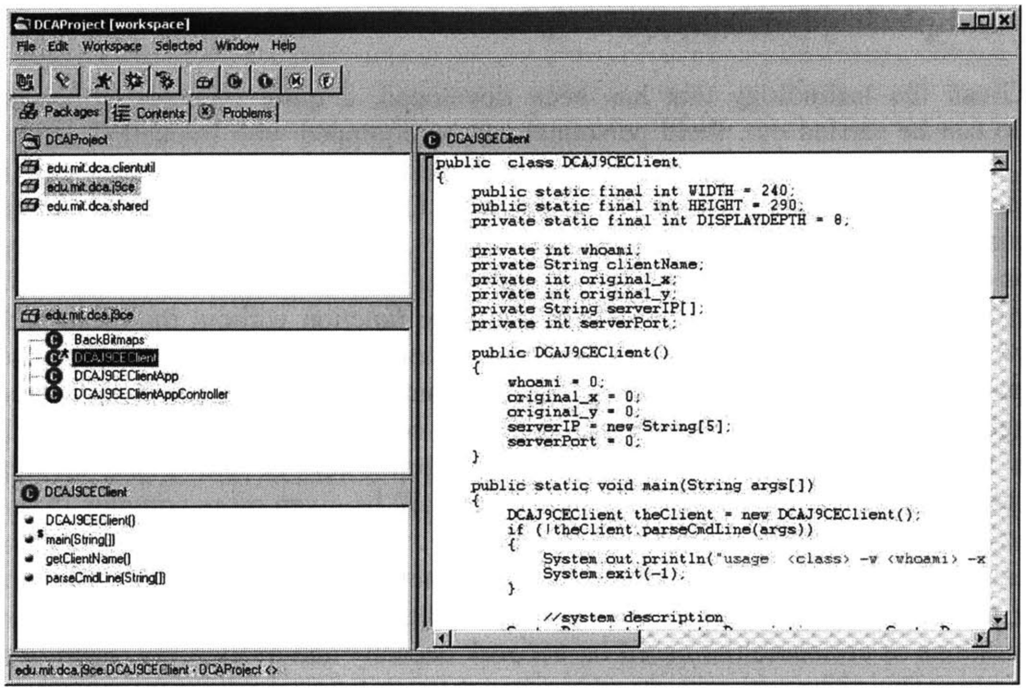


Figure 18 - Implemented Messaging System

3.3.2.3 Windows CE and Palm Devices

PDAs like Windows CE and Palm Pilots are useful collaborators because its mobility and portability. In this research, Java programs were developed as client applications for Windows CE devices, including Compaq iPaq 3630 and CASIO Cassiopeia E-125. The development environment is IBM's VisualAge Micro Edition 1.3 [31], which features the J9 virtual machine. J9 efficiently executes Java bytecodes or can use just-in-time (JIT) or ahead-of-time (AOT) compilation, and fully support Java Native Interface (JNI) native methods. IBM boasts of a number of attractive points for its development environment, such as high performance garbage collection routines, realtime extensions based on the Realtime Specification for Java, capability to remotely maintain and update applications, server support for connected devices via TCP/IP, additional class library support for the unique needs of embedded systems, and tools provided for code configuration, optimization, and reduction tasks.

Figure 19 shows a screen shot of the source codes for MIT-Draper project. The program runs on Windows CE as shown in Figure 9. The same program can be executed on Palm devices without any modification so long the IBM J9 virtual machine is set up in the Palm device.

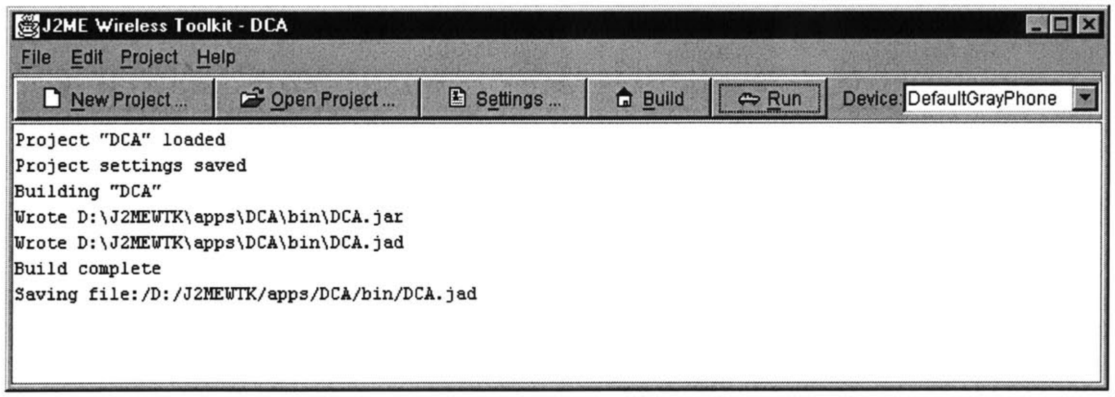


```
public  class DCAJ9CEClient
{
    public static final int WIDTH = 240;
    public static final int HEIGHT = 290;
    private static final int DISPLAYDEPTH = 8;

    private int whoami;
    private String clientName;
    private int original_x;
    private int original_y;
    private String serverIP[];
    private int serverPort;

    public DCAJ9CEClient()
    {
        whoami = 0;
        original_x = 0;
        original_y = 0;
        serverIP = new String[5];
        serverPort = 0;
    }

    public static void main(String args[])
    {
        DCAJ9CEClient theClient = new DCAJ9CEClient();
        if (!theClient.parseCmdLine(args))
        {
            System.out.println("usage: <class> -w <whoami> -x
            System.exit(-1);
        }

            //system description
```

Figure 19 – IBM VisualAge Micro Edition Development Environment

3.3.2.4 Java-Enabled Phone and Pager

The capabilities of Java-enabled phones handheld devices are increasing rapidly as technology enables smaller and more powerful devices to perform functions similar to those found in desktop solutions. In this research, Java programs were developed using Sun Microsystem's Java 2 Platform Micro Edition Wireless Toolkit. The J2ME Wireless Toolkit is a set of tools that provides Java developers with the emulation environment, documentation and examples needed to develop MIDP compliant applications targeted at mobile information devices such as cellular phones and two-way pagers [32]. The current version of the product is based on the J2ME Connected Limited Device Configuration (CLDC) 1.0 and the Mobile Information Device Profile (MIDP) 1.0 technical specifications. Figure 20 shows the interface of the development environment, and the running program is shown in Figure 10.



```
Project "DCA" loaded
Project settings saved
Building "DCA"
Wrote D:\J2MEWTK\apps\DCA\bin\DCA.jar
Wrote D:\J2MEWTK\apps\DCA\bin\DCA.jad
Build complete
Saving file:/D:/J2MEWTK/apps/DCA/bin/DCA.jad
```

Figure 20 - Java 2 Platform Micro Edition Wireless Toolkit

3.3.3 Sample Implementation

Given the technology that has been developed, a quite complex collaboration scenario can be carried out. Field personnel can be equipped with PocketPC and phone acting as data collectors. When something interesting is discovered they can input locations and change existing location information. Other collaborators can be connected via desktop computers and represent mission control, monitoring field agent activity and perhaps entering information regarding scheduling of their activities. In addition, wireless LAN enabled computers can also perform the same function without the location fixity required by the desktop computer. These collaborators could move around freely within the networked zone, perhaps monitoring several rooms with mission planners. Moreover, field agents that required more contact could be equipped with web-enabled cellular phones that allow them to communicate with mission control and access application information. These may represent field agents that will be given more complex directives and they may be asked to report their findings verbally, and not through a common application. As presented, this sample scenario shows the power of the collaboration system presented, and introduces some potential pitfalls such as the need for dynamic configurations of collaborators based on network capacity and availability. Each of these shortcomings should be anticipated and handled through future work on the system.

Chapter 4

Special Mechanisms for Multi-server Collaboration Systems

This chapter presents two important mechanisms specially designed and implemented for smooth and efficient operation of multi-server collaboration systems. These mechanisms are characteristics of multi-servers, thus not applicable to single-server system. Besides, the chapter also presents a mechanism to manage heterogeneous platform interfaces, such as PC, laptop, Window CE, Palm, phone and pager. This mechanism is suitable for both single-server and multi-server collaboration systems.

4.1 Performance and Reliability Improvement Mechanism

Performance and Reliability Improvement Mechanism (PRIM) is designed and implemented as a software package to provide a configurable, reliable data transmission and storage system. Hereby the servers running in the multi-server collaboration system are called collaboration engines, which not only assume the normal tasks of the central server in a single-server system, but also undertake the jobs of synchronizing with each other and assisting in the handoff process. Moreover, these engines acting as PRIM controllers to configure and implement the PRIM mode of the distributed application repositories, co-ordinate and conduct the redundancy.

4.1.1 Techniques for PRIM

For ensuring high data availability and excellent system performance, PRIM uses the following three important techniques, *data striping*, *data mirroring* and *error recovery*. These terms are adapted from the RAID literature to collaboration systems. RAID, standing for **R**edundant **A**rray of **I**nexpensive **or** Independent **D**isks, is the term used to describe a storage systems' resilience to disk failure through the use of multiple disks and by the use of data distribution and correction techniques [33].

4.1.1.1 Data Striping

Data striping is a method of mapping data across the physical drives in an array to create a large virtual drive. In multi-server collaboration systems, application data is subdivided by collaboration engines into consecutive segments or *stripes* that are written sequentially across the network into the array of different application repositories. Each stripe has a configurable size.

4.1.1.2 Data Mirroring

Data mirroring is a simple fault-tolerant method where the mirroring array consists of multiple sets of data stored on two or more repositories with all the data fully backed up. For the PRIM implementation, the collaboration engine writes the application data to different application repositories in parallel. Although most data mirroring implementations involve two sets of data (hence the term *mirror*), three or more sets can be created in PRIM for increased reliability in the collaboration system.

If an application repository failure occurs in a mirroring array, subsequent read and write operations are directed to the surviving application repositories. A replacement repository is then rebuilt using application data from the surviving repositories, whenever that engine is going back to the session. This rebuilding process has some impact on the collaboration engine's performance because all application data must be read and copied from the surviving repositories to its own repository.

4.1.1.3 Error Recovery

Error recovery is especially important for data striping, because the collaboration engine needs to read data from separate application repositories. If one application repository is not available, either because of physical failure or network loss, the collaboration engine can rebuild the application data from other workable repositories. Typically, exclusive OR (XOR) function of data and parity information on the remaining application repositories is computed to *regenerate* the data on the failed repository.

4.1.2 Core Ideas of Implementing PRIM in Collaboration Systems

The striped array of repositories can offer improved read and write performance compared to an individual application repository, as the collaboration engines can access the striped application repositories in parallel. However, the stripe size should be matched to the type of collaboration services or tools that are running:

- For input/output (I/O) intensive or transactional collaboration services, larger stripes are preferable since multiple concurrent requests are made for small data records. If a stripe on an individual application repository is large enough to contain an entire record, that application repository can respond independently to these simultaneous data requests.
- For application-intensive collaboration services, smaller (byte-level) stripes are more appropriate since large data records are stored. If a given data record extends across several application repositories in the array, the contents of the record can be read and written in parallel, improving the overall data transfer rate.

In collaboration systems, the storage decision is fundamentally a trade-off between data striping and data mirroring. Data mirroring offers fault-tolerance. However, as service levels are increased, and the number of repositories in the multi-server system increases, the differences between one PRIM configuration and another becomes important. In this case, time to rebuild an unsynchronized repository, risk of data loss through the write hole, parity verification, and physical security become key considerations. In this research, several PRIM configurations are proposed and designed for collaboration systems. Not surprisingly, all the configurations are a trade-off between data striping and data mirroring with some error-recovery mechanisms.

4.1.3 Typical PRIM Configurations

This section describes in detail typical PRIM configurations. To describe the different configurations, this research uses a terminology similar to the RAID scheme.

4.1.3.1 PRIM 1: Mirroring

PRIM 1 is simply mirroring. Each collaboration engine has its own application repository, called principal repository and has at least one repository as its mirror repository. During a collaboration session, each engine reads and writes data to the principal repository and mirror repository simultaneously. This ensures that both repositories have the same snapshot of data and applications at any time. One engine's principal repository can be configured as another engine's mirroring repository, and vice versa. If the principal repository is unavailable, engines can still run by relying on the mirror repository.

Figure 21 illustrates a PRIM 1 configuration, where *engine A* uses *repository A* as its principal repository, *B* as its mirror repository, and *engine B* uses *repository B* as its principal repository, *A* as its mirror repository. Both *repository A* and *repository B* contain the same AutoCAD® drawing. When the repositories are working as planned, each collaboration engine is able to get the drawing from its principal repository to serve its own clients. Each engine is also responsible for synchronizing the drawing in the mirror repository. If, for some reason, *repository A* fails, *engine A* can still get the up-to-date drawing from its *repository B*, and *engine B* does not need to synchronize *repository A*. Whenever *repository A* goes back to work, *engine B* will immediately rebuild *repository A* from *repository B*, and *engine A* will resume using *repository A* as its principal repository.

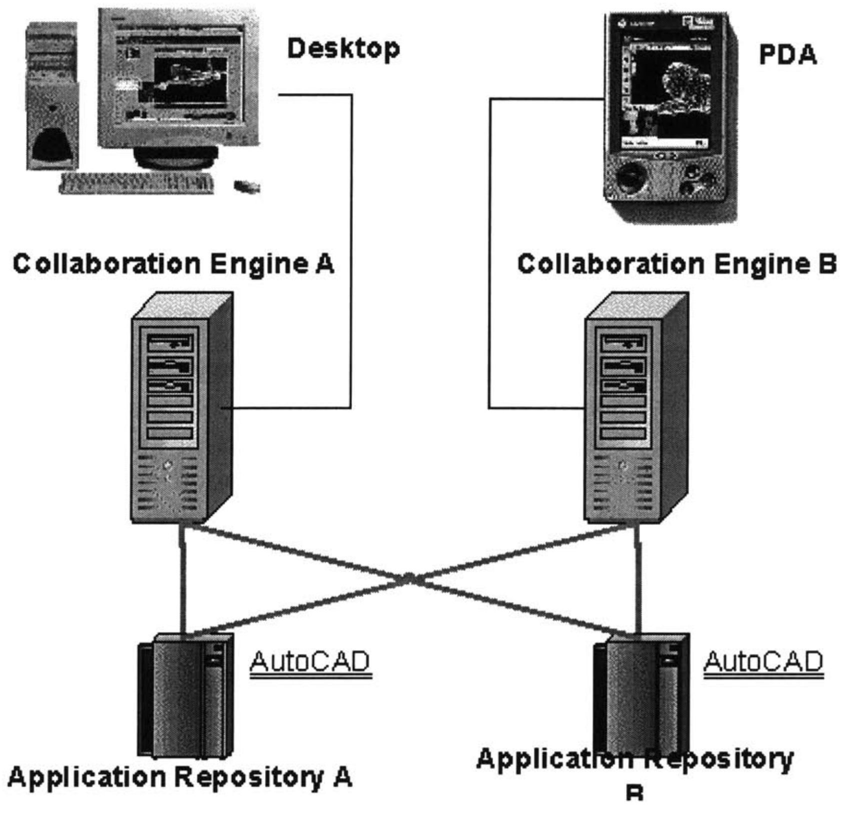


Figure 21 - PRIM 1 Concept Illustration

PRIM 1 offers high data availability because at least two complete sets of data are stored. It is also easy to implement for multi-server collaboration systems. PRIM 1 is better suited for small databases or other small-scale systems that emphasize reliability. PRIM 1 is the preferred configuration when the collaboration session involves limited number of clients and data communication traffic is low. For example, PRIM 1 will be useful when the collaboration services are text chat, whiteboard, and real-time polling.

4.1.3.2 PRIM 2: Principal + Simple Striping

In PRIM 2 configuration, each collaboration engine has its own principal application repository, and a number of other application repositories to which the engine distributes data. These repositories are called striped repositories. An engine has two choices when reading data, it can either read data from its principal repository, or read it from several striped repositories in parallel and assemble the final data. The exact method to use is decided by the repository working status and networking status. When the engine writes data, it not only writes data to its principal repository, but also splits the data and writes them separately to striping repositories. The flexibility of PRIM 2 is that several engines can share both principal repositories and striped repositories.

Figure 22 illustrates a PRIM 2 configuration. Both *engine A* and *engine B* use *repository A* as the principal repository to store an AutoCAD® drawing. Meanwhile, they use *repository C* and *D* as their striped repositories, in which the first and second halves of the AutoCAD® drawing are stored, respectively. Whenever *engine A* or *engine B* changes the drawing, it not only updates the drawing in *repository A* but also splits the drawing data and writes them to *repository C* and *D*. Whenever *engine A* or *engine B* reads the drawing, it may read from *repository A*, or from *C* and *D* in parallel.

When the principal *repository A* becomes unavailable, the collaboration engines will try to rely on repositories *C* and *D*. Meanwhile, the engine will try to keep the system operating in PRIM 2 mode in the following two ways: it will attempt to seek another engine from which to build a principal repository, or it will promote one stripped repository (like *C*) to principal repository, and make *repository D* and another repository from other engines become new stripped repositories. When *repository A* goes back to work, the system will recover *A* as the principal repository. If, either stripped repository *C* or *D* fails, engines can either simply work with *repository A*, or find a repository from other engines to build it as a stripped repository. As described before, the failed repository will be rebuilt when it recovers.

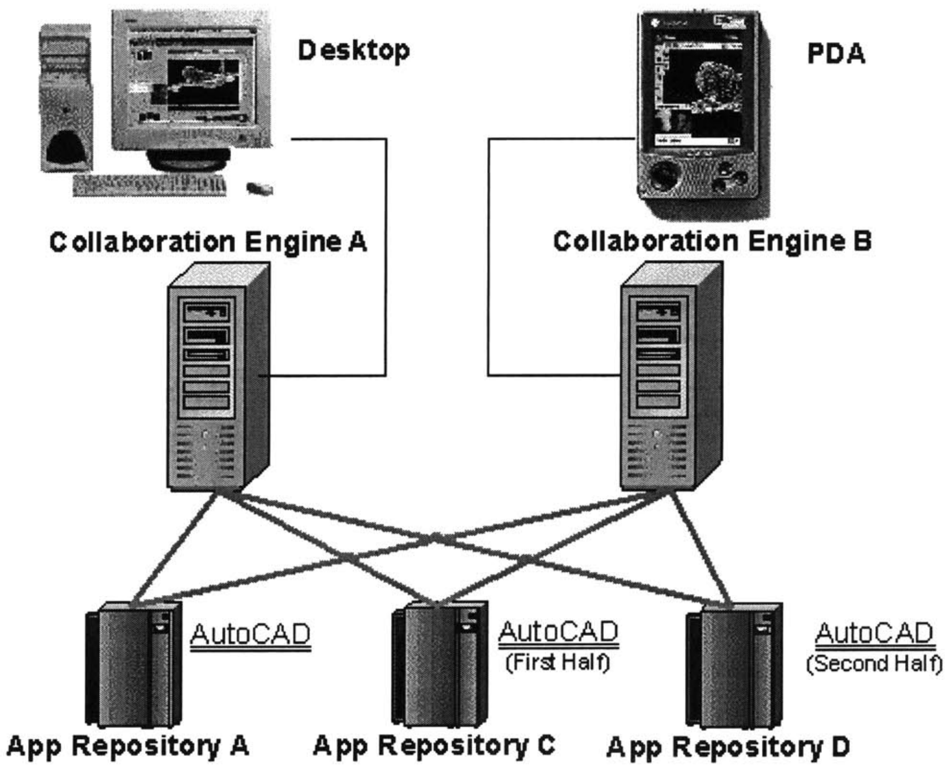


Figure 22 - PRIM 2 Concept Illustration

PRIM 2 offers both data performance and fault-tolerance. Since the data can be transferred in parallel to and from mirror repositories, PRIM 2 offers higher performance than a single repository mechanism. PRIM 2 is suitable for collaboration sessions that involve large amount of data transfer but not much data transactions, for example, playback of old collaboration sessions.

4.1.3.3 PRIM 3: Principal + Striping with Parity

PRIM 3 resembles PRIM 2 in the way that engines have principal repositories and distribute data to striped repositories. Different engines can share the same set of principal repositories and striped repositories. The difference is that in PRIM 3, an extra repository is used to store the parity information for error recovery. Thus when one striped repository fails or is not accessible, the engine can rebuild data and use other repositories, although the system "downgrades" to PRIM 2. Note that reading process for PRIM 3 is the same as PRIM 2.

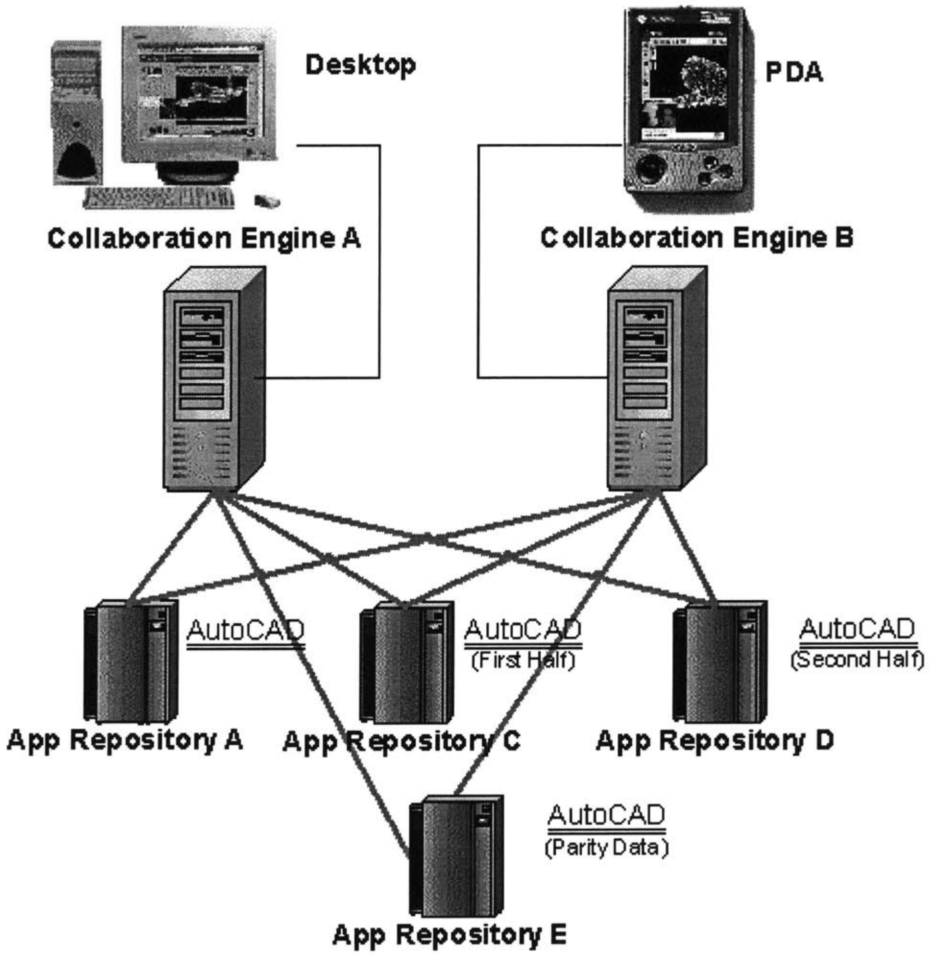


Figure 23 - PRIM 3 Concept Illustration

Figure 23 illustrates a PRIM 3 configuration. In this case, *repository E* is used to store parity data calculated from the data in repository *C* and *D*. Whenever *engine A* or *engine B* changes the drawing, it not only updates the AutoCAD® drawing in *repository A*, splits the drawing data and writes them to *repository C* and *D*. In addition, it also calculates the parity data and writes it to *repository E*. Whenever *engine A* or *engine B* reads the drawing, it may read from *repository A*, or from *C* and *D* in parallel. Similar to PRIM 2, if any repository becomes unavailable, engines will not only continue working with the remaining repositories, but also actively seek repositories from other engines to keep the system operating in PRIM 3 mode. For example, if the principal repository A fails, engines will try to find a repository from other engines and build it as the principal repository. If that does not work, engines will try to promote one stripped repository to principal repository. Similarly, if a stripped repository or parity application repository fails, engines will try to find a repository from other engines and build it as a stripped repository or parity application repository. In this way, the system will be capable of operating in PRIM 3 mode to provide non-stop services to clients.

PRIM 3 offers both data performance and strong fault-tolerance. It offers the same reading performance as PRIM 2, but its fault-tolerance is much better than that of PRIM 2. PRIM 3 is suitable for collaboration sessions that involve either large amount of data transfer or are transactional heavy, for example, application sharing and text chat.

4.2 Software-Based Handoff Mechanisms

For a multi-server collaboration system, wired or wireless servers and/or clients make the network topologies more flexible and dynamic. Due to the mobility of wireless servers and clients and constraints of dynamic wireless networks, the connectivity of the system components (servers and clients) becomes unstable and fragile. This in turn leads to frequent loss of service offered by the servers. Hence, the collaboration system needs to control the connections between system components and handle the service handoff between servers to maintain the collaboration session, service continuity and information consistency. In this context, service handoff relates to the transfer of collaboration service between the collaboration engines and dynamic re-allocation of client-server connections.

To improve the quality of service (QoS) in a multi-server collaboration system, this section presents a software-based service handoff mechanism. The mechanism is applicable to complicated distributed collaboration environments combining both wired servers with wired or wireless clients (shown in Figure 24) and mobile servers with wired or wireless clients (shown in Figure 25).

4.2.1 Dynamic Wireless Networking Constraints

The mobility of the components (servers and clients) in the distributed environment and the dynamic of the wireless network greatly influences and constraints the quality of service (QoS) in a collaboration system. The network topologies are changing when mobile clients or mobile servers change their locations. The mobile client may go out of the service coverage of its server or the mobile server may be too far away from its clients. To provide the best quality of service, the system should try to optimize the clients' access by dynamically assigning each client to the best available server during the collaboration session. The system constantly monitors the movement of clients and servers, and makes necessary changes in client-server connections, based on the conditions of client locations, server locations, and network capabilities.

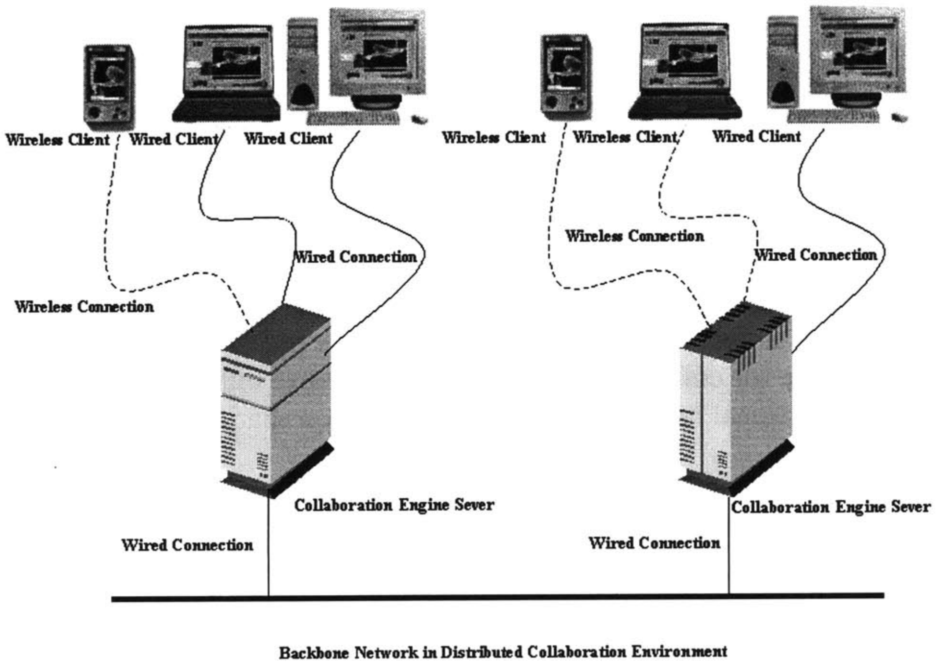


Figure 24 - Wired Servers with Wired or Wireless Clients

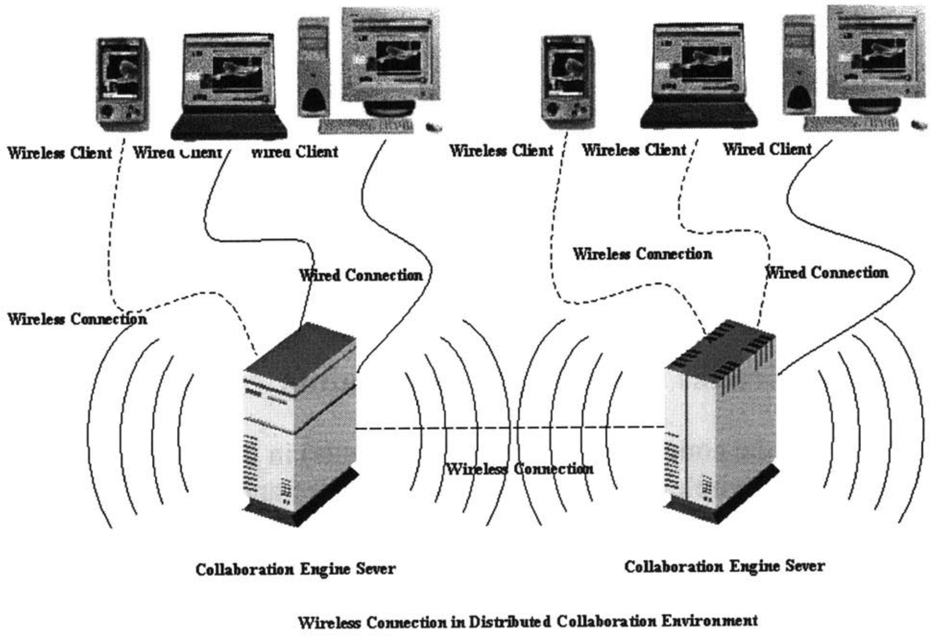


Figure 25 - Wireless Servers with Wired or Wireless Clients

Figure 26 shows a group of distributed collaboration servers (static or mobile) around the world providing collaboration services for multiple clients (static or mobile). The mobile clients can automatically search for the best available server for services at anytime during the collaboration session, no matter whether the servers or clients are in wired or wireless connection (shown in Figure 27). Such a distributed collaboration system can protect data consistency and ensure QoS in an interaction session.

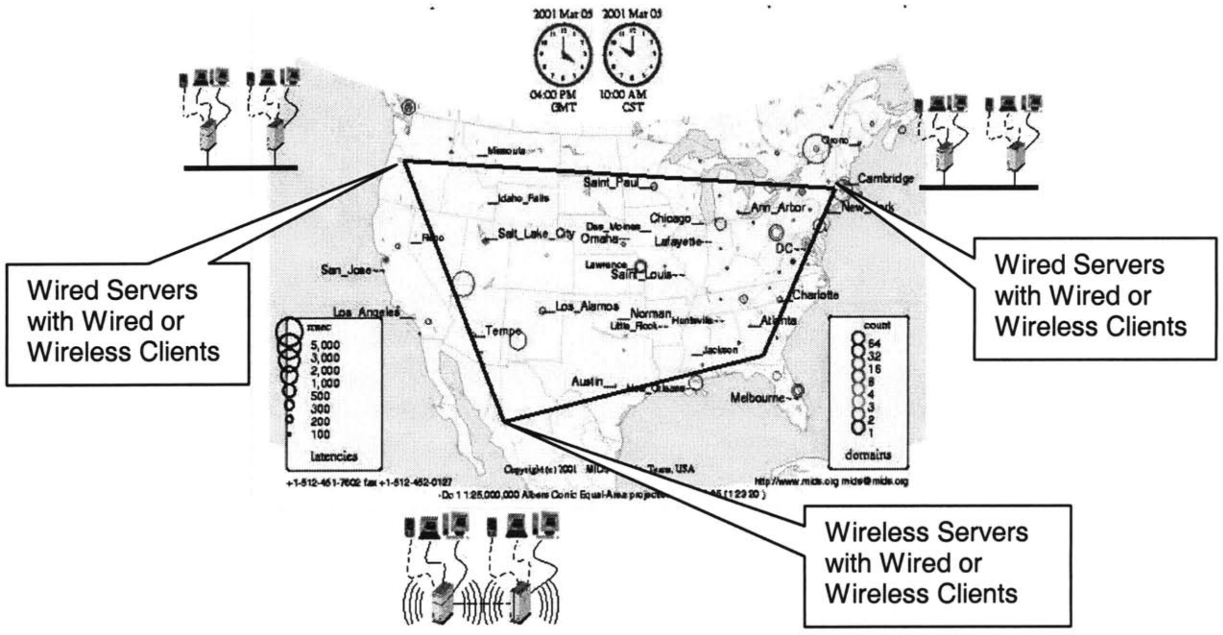
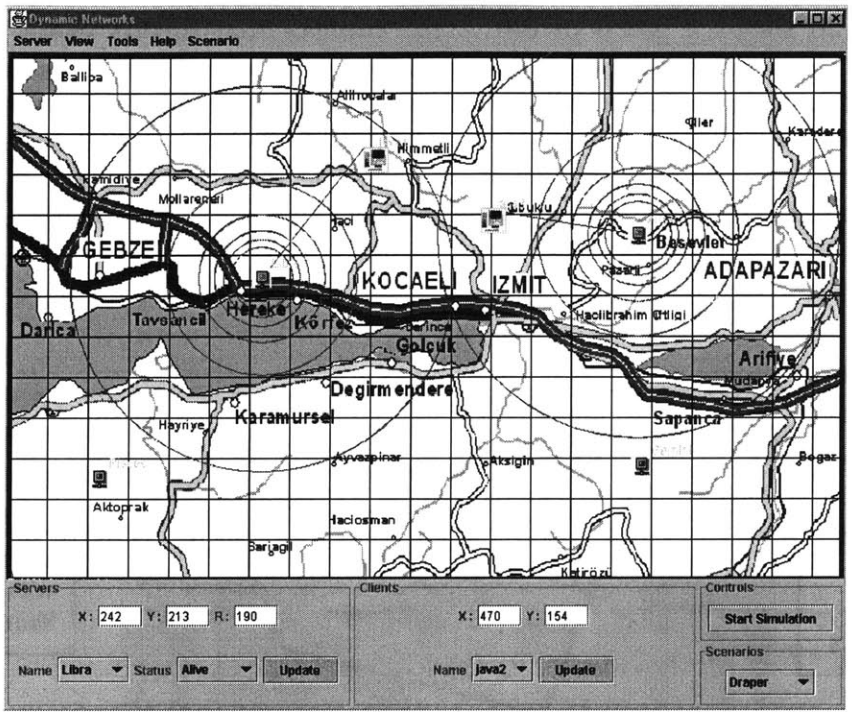

Wired Servers with Wired or Wireless Clients
Wired Servers with Wired or Wireless Clients
Wireless Servers with Wired or Wireless Clients
2001 Mar 05
04:00 PM GMT
10:00 AM CST
latencies
domains

Figure 26 – Multi-Server Collaboration System with Wired and Wireless Networking



Servers: X: 242 Y: 213 R: 190 Name: Libra Status: Alive
Clients: X: 470 Y: 154 Name: java2
Scenarios: Draper

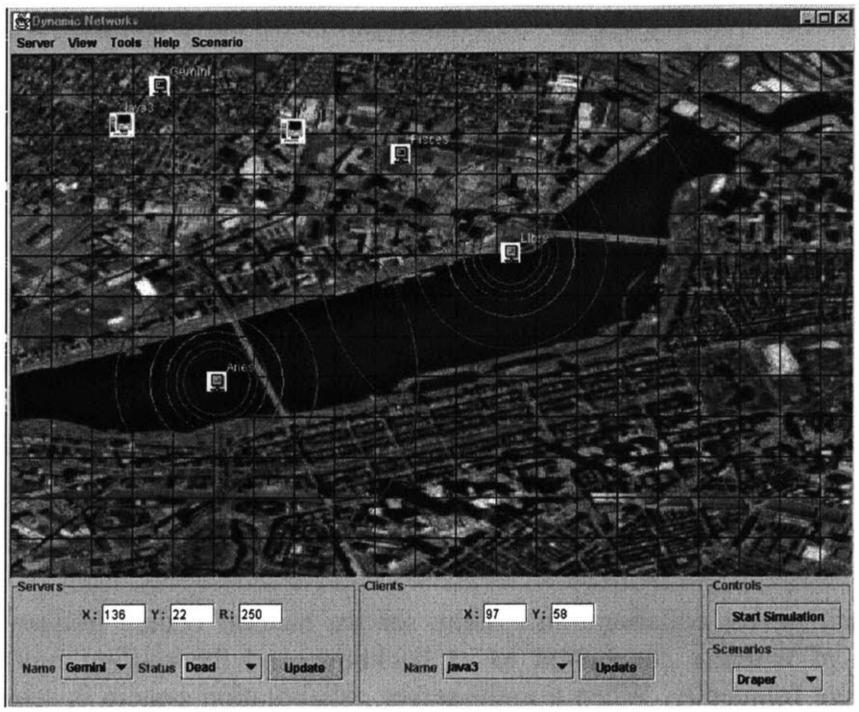
Figure 27 - Mobile Clients Searching for the Available and Best Mobile Server

4.2.2 Service Cell and Service Handoff

Network protocols and applications must deal with limited bandwidth, high latencies, sporadic high bit-error rates and temporary disconnections typical to communication over wireless links. The core issue of providing collaboration services in the distributed environment over a mobile wireless network is QoS support in the presence of changing network connectivity.

4.2.2.1 Service Cell

Wireless LAN technology use cells, called microcells, similar to the cellular telephone system to extend the range of wireless connectivity. The signals that can be carried by each wireless Access Point are limited by its power output. Individual microcell overlap to allow continuous communication within wired network. At any time, a mobile PC equipped with a WLAN adapter is associated with a single access point and its microcell, or area of coverage. Access points handle low-power signals and handoff for users as they roam through a given geographic area.



X: 136 Y: 22 R: 250

Name Gemini Status Dead

X: 97 Y: 58

Name java3

Draper

Figure 28 - Collaboration Server's Service Cell

In a multi-server collaboration system, due to each collaboration engine's network conditions, it has its own limited service coverage, which is called service cell. The service cells of different collaboration engines may overlap with others (shown in Figure 28). In case that a collaboration engine is connected with other engines and clients by a wireless tower, the power of the tower transmitter determines the range of service cell. Similar to microcells in the wireless LAN, the use of service cells in the collaboration system offers increased spatial efficiency and service quality thereby by ensuring the rate of service handoffs between the servers and clients. Unlike wired networks, the communication components in wireless networks, i.e. mobile clients and mobile servers, change their connectivity dynamically in an active collaboration session, henceforth service handoff happens frequently during any collaboration session.

4.2.2.2 Service Handoff

Service handoff refers to the transfer of collaboration service between the collaboration servers and dynamic re-allocation of client-server connections for wired or wireless networks.

The service handoff takes place, when

- A mobile client moves from one service cell to another.
- A mobile collaboration server moves far away from its clients, making the clients out of reach of its service cell.
- A collaboration server cannot provide services to its current clients due to some hardware, software or network problems. In this case, service handoff

occurs between the service cells from this server to the next neighboring or best available server.

- The system connects the clients to the best server available at anytime during a collaboration session. This scenario occurs as the system monitors the QoS of the services to the clients and tries to optimize the clients' access to information and services.

During the service handoff, the collaboration server of each service cell needs to hand over the connection responsibility and service continuity within the collaboration session of that mobile component. The next section highlights the key features of the proposed software-based service handoff as a solution to the problems identified earlier in this section. The mechanism of the proposed software-based service handoff can be applicable to different network connectivity conditions.

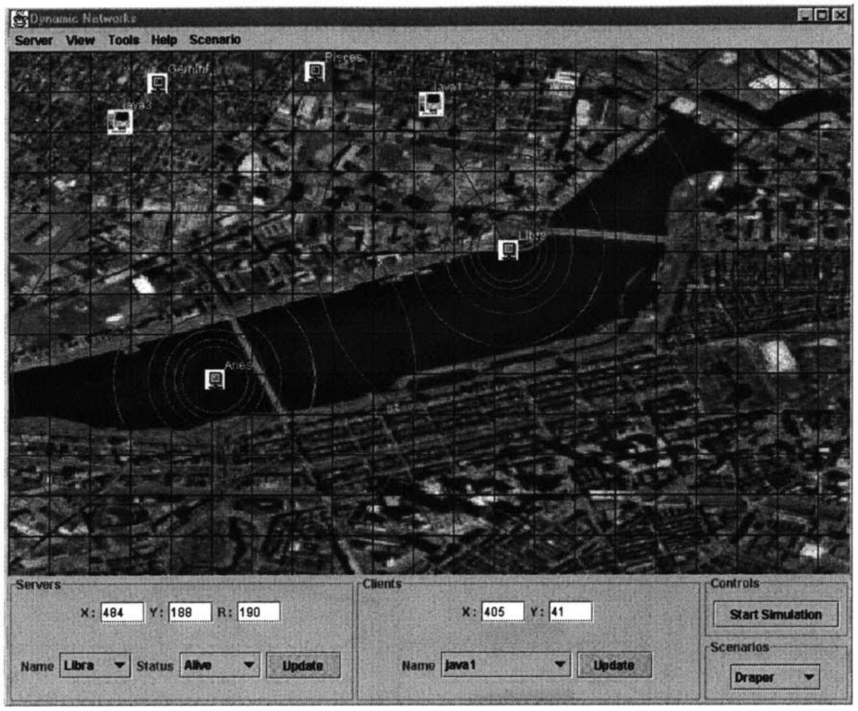
4.2.3 Service Handoff Solution

In a multi-server collaboration system, service handoff occurs frequently because the interface of wireless connection to its backbone and the routing function of the network change frequently. When a multi-server collaboration system is setup, a virtual connection tree forms, which is a collection of collaboration servers and wired network switching nodes and links. The root of the tree is a fixed switching node of the wired network (mobile access points or collaboration servers), and the leaves of the tree are mobile clients. Each collaboration server maintains a routing table of the whole connection tree, including information of other collaboration servers, data repositories and client applications. The collaboration server broadcasts its services and monitors the clients within its service cell, such as collaboration servers Libra and Aries shown in Figure 28.

4.2.3.1 Network Location and Connection Establishment

Figure 29 illustrates that mobile client *java1* is registered to the nearest available collaboration server *Libra* when it entered the service cell of *Libra*. When a mobile client enters the service cell of one collaboration server, a connection is established between the mobile client *java1* and the collaboration server *Libra*. In order to re-establish the connection, the mobile client must be located based on the following two main functions:

- Searching: Client queries the entire network to look for the nearest collaboration server and sends a request for service.
- Registration: Client is responsible for its own registration at the available collaboration server, providing its device type, location, networking capability, and other information.



X: 484 Y: 188 R: 190

Name Libra Status Alive

X: 405 Y: 41

Name java1

Draper

Figure 29 - Collaboration Connection Establishment

Once the collaboration server gets the request from the mobile client within its service cell, it launches several functions of the collaboration server:

- Registration: Collaboration server registers the client into its client list and updates the routing table of the connection tree.
- Relocation: Collaboration server records the location and tracking the movement of the clients (within its service cell).
- Monitoring: Collaboration server monitors the quality of service it provides to the mobile clients (within its service cell) based on self-leaning algorithms by collecting the following information: the client's networking capability, the signal strength received by the client, and response time from the client.
- Transferring: If the location of the mobile client is out of the collaboration server's service cell, or and the qualities of the service the mobile client is served is not smooth enough, the original collaboration server transfers the service to the next available or best available collaboration server.
- Re-allocation: Collaboration server re-allocates the client-server connections. The server configures and controls the transmission and storage mode over the distributed data repositories for this mobile client. This configuration is based on to premise of optimizing the quality of service while monitoring the requested client services, the server conditions and the network conditions.

- Notification: Collaboration server notifies all the other collaboration servers to update the routing table and the new location of the mobile unit.
- Broadcast: The collaboration server keeps broadcasting its available services for all the mobile clients within its service cell.

Figure 30 illustrates the sequence diagram for the whole process of connection establishment and transfer.

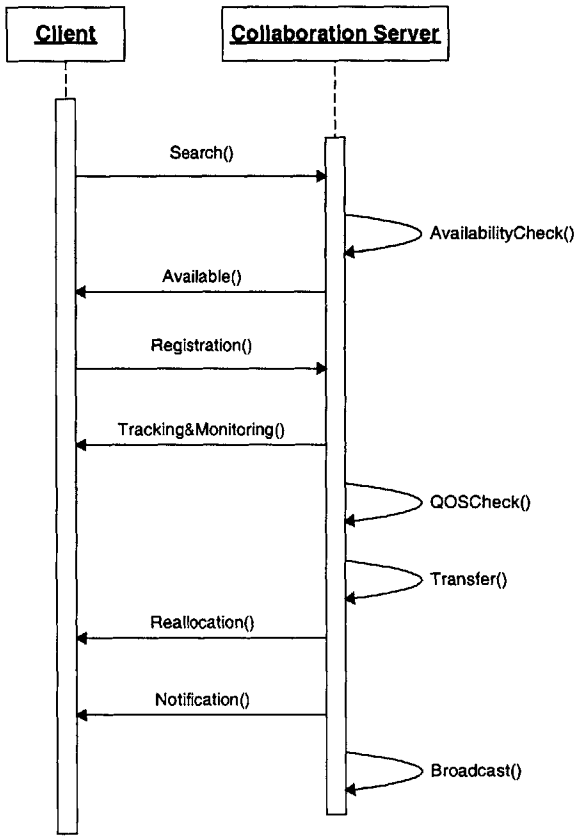


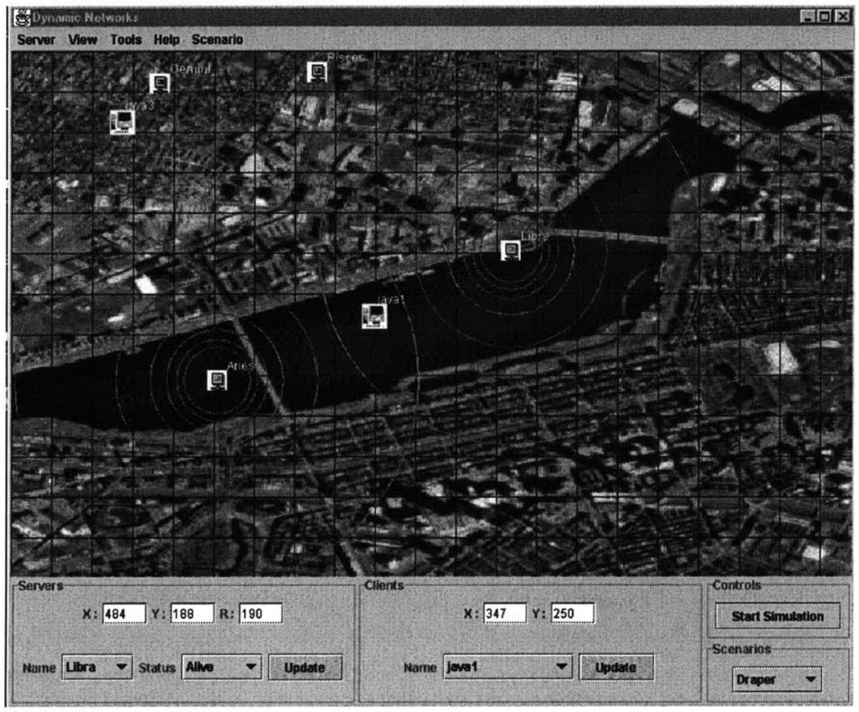
Figure 30 – Sequence Diagram for Connection Establishment and Transfer

4.2.3.2 Soft Service Handoff

The multi-server collaboration system uses soft service handoff to achieve seamless handoffs among collaboration servers. A soft handoff is essentially a “make before break” connection. The connection between the mobile client and the collaboration servers are established and can be maintained indefinitely. Two or more collaboration servers near the mobile client might be involved in the handoff. When the mobile client leaves the original service cell, there is no drop off between or among service cells, so as to keep the continuity of the service and consistency of the collaboration session. The soft service handoff processes are similar for mobile clients and mobile servers.

As an illustrative example, consider the following scenario with a wired-connected server and wireless clients in the environment. When a mobile client moves into the overlapped area of two collaboration servers' service cells, it keeps searching for the nearest available collaboration server and prepares for registration. The two collaboration servers involved in the service handoff monitor the mobile unit's location, the signal strength received by the mobile unit and the response time from the mobile unit. The collaboration system then makes the decision which server will serve the mobile client, and whether a handoff will occur.

In Figure 31, when mobile client *java1* moves into the overlapped area of collaboration servers *Libra* and *Aries*'s service cells, *Libra* and *Aries* keep monitoring *java1* simultaneously. Because of the shorter distance to *Libra* and better quality of service from *Libra*, no handoff needs to take place.

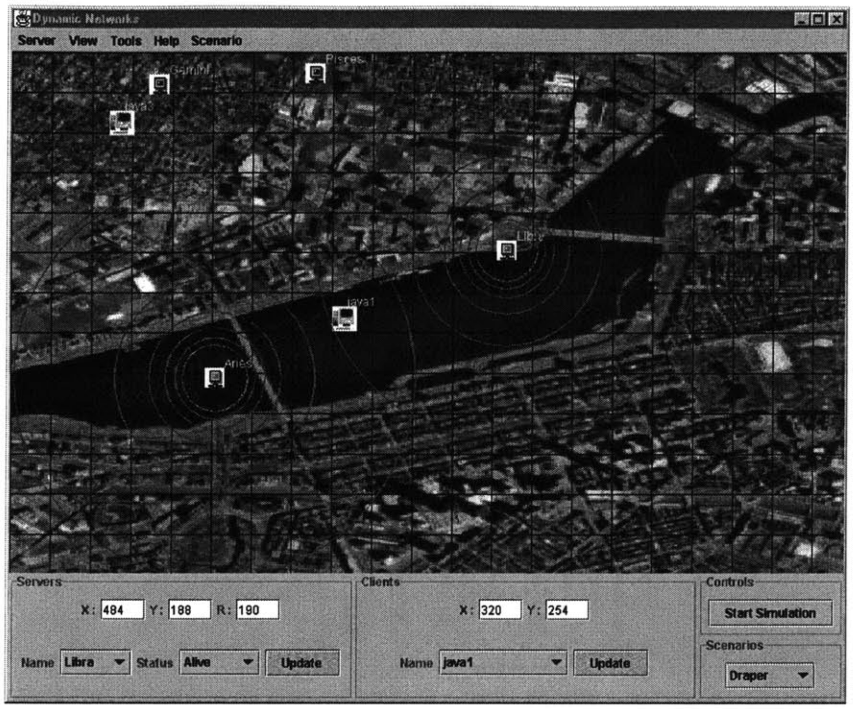


Servers — X: 484 Y: 188 R: 190 — Name: Libra — Status: Alive

Clients — X: 347 Y: 250 — Name: java1

Scenarios: Draper

Figure 31 - Mobile Client Located in the Overlapped Area of Two Service Cells



Servers

X: 484 Y: 188 R: 190

Name: Libra Status: Alive

Clients

X: 320 Y: 254

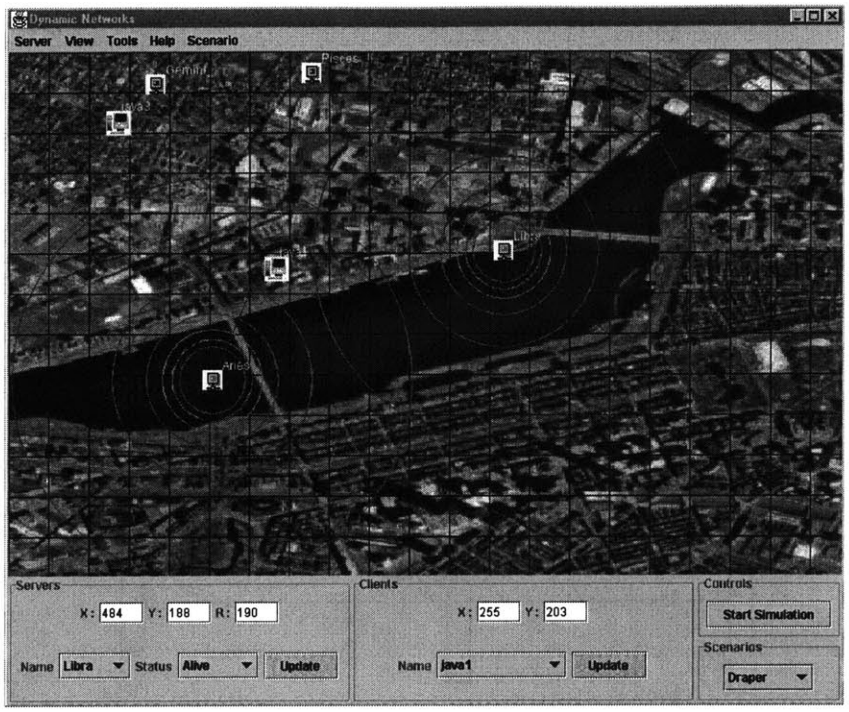
Name: java1

Controls

Scenarios: Draper

Figure 32 - Handoff Occurred Between Two Collaboration Servers

Once the mobile client *java1* gets more and more closer to the collaboration server *Aries* (shown in Figure 32) or leaves the service cell of the collaboration server *Libra* (shown in Figure 33), the system will be able to find out that the collaboration server *Aries* can provide better quality of service to the mobile client, hence the handoff between *Libra* and *Aries* takes place.



Servers X: 484 Y: 188 R: 190
Name Libra Status Alive
Clients X: 255 Y: 203
Name java1
Scenarios Draper

Figure 33 - Handoff When Mobile Client Out of Service Cell

To complete the handoff, the following steps need to be completed:

- The original collaboration server *Libra* first notifies *Aries* of the handoff and transfers all the system information about the mobile client *java1* to *Aries* for registration.
- Once getting the handoff information, collaboration server *Aries* establishes a new connection with *java1* and registers it into its system, updating the routing table.
- *Aries* notifies the system to update routing table.
- *Libra* transfers the required services and session information about the mobile client, and control of the data repositories related to the collaboration session.
- *Aries* takes over the services for the mobile client and controls the corresponding data repositories after getting required information from *Libra.*
- After the entire transfer is done, *Libra* cuts off its connection with *java1* and stops the service for the mobile unit, but keeps monitoring the mobile client in its service cell.
- Handoff process is completed.

The handoff process is illustrated in sequence diagram Figure 34.

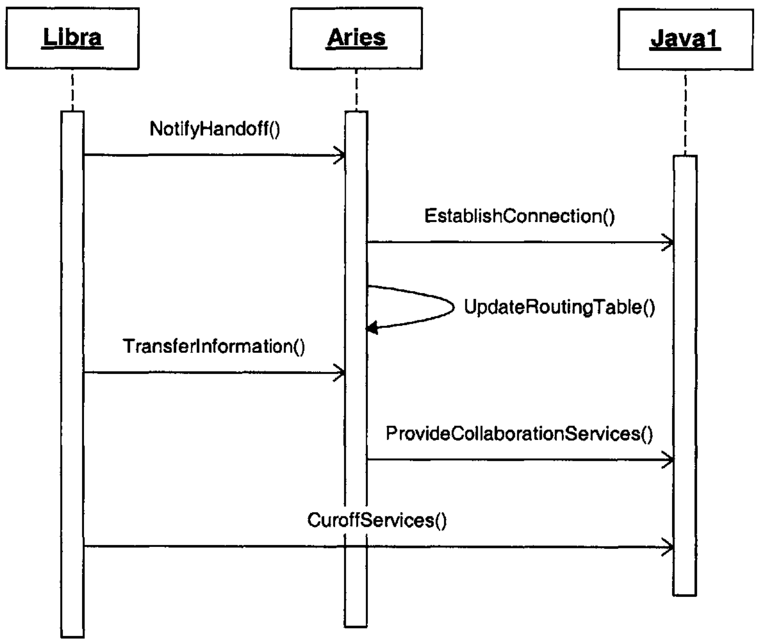


Figure 34 – Sequence Diagram for Soft Services Handoff

In conclusion, the soft service handoff provides a mechanism for smooth transfer between collaboration servers. By ensuring data persistency and information consistency among the servers of the distributed collaboration system, the mechanism provides a seamless service for mobile users in a multi-server collaboration system.

4.3 Mechanism to Manage Heterogeneous Platform Interfaces

With recent advances in operating systems and computer networks, the current marketplace is filled with a number of computing platforms (desktop computer, laptops, handheld devices running Windows CE®, handheld devices running Palm OS®, cellular phones, and pagers) connected to the Internet with a number of different network options (wired or wireless). Keeping this in mind, it is envisaged that in a collaboration session, distributed participants would use these heterogeneous devices and network options to run collaboration services or tools like text-chat, whiteboard, audio-video, real-time polling, and application sharing. Current implementations of collaboration systems lack proper mechanisms for identifying individuals and their actions in a controlled manner during a collaboration session. Moreover, since participants have computing platforms with different viewing screens and resolutions (colloquially called screen real estate), they will not be able to view the same area of the collaboration services or tools like whiteboard and shared applications. Both constraints inevitably lead to an increase in participants' overhead during the collaboration session.

To reduce participants' overhead and increase overall efficiency of collaboration session, this research presents the following three sub-mechanisms and the underlying concepts:

- Controlled-Participant-Identification (ISWYD, **I See What You Do**)
- Port-of-View (ISWYS, **I See What You See**)
- Focus-of-Attention (IMYSWIS, **I Make You See What I See**)

Controlled-Participant-Identification mechanism involves showing a short and concise participant-related message with a participant's pointing device (for example, mouse pointer for a desktop computer). This mechanism is triggered whenever the participant is controlling the collaboration session or working with a particular collaboration service or tool in the session. For brevity, this mechanism is abbreviated as ISWYD (I See What You Do).

Port-of-View mechanism involves revealing on a device with larger screen real estate, a portion of the screen or the entire window that is currently being viewed by a participant with smaller screen real estate. For brevity, this mechanism is abbreviated as ISWYS (I See What You See).

Focus-of-Attention mechanism involves synchronizing participants' viewports with the viewport of the participant in control of the session. The synchronization process is jointly determined by the position of the participants' pointing devices and the screen real estate of the other participants. This mechanism is abbreviated as IMYSWIS (I Make You See What I See).

4.3.1 Constraints of Collaboration Systems From the Perspective of User Interfaces

The ultimate goal of a collaboration system is to provide geographically distributed participants with the same bandwidth and multiple channels of information as face-to-face interactions while providing additional functionalities, such as record-and-playback, real-time polling and application sharing. However, there is a need to appreciate the tremendous complexities involved in trying to reach this objective. The most important of them are highlighted below:

- The nature of software that enables collaboration is enormously complex and multi-dimensional in scope. Mapping real life scenarios such as physical settings, procedures and presence to a software process that would be easy enough for participants to comprehend and use, is extremely difficult.
- Computing power, screen real estates and bandwidth limitations impose significant constraints on the richness of the communication media that can be offered reliably, and with all available functionality.

In this regard, to gain a sense of "telepresence" in the computerized interaction, current collaboration systems need to be improved to enable distributed participants to be alert of the following information:

- Who is currently controlling the session?
- Who is currently controlling specific pointing devices?
- What platforms are other participants using?
- What are the other participants focusing on?
- Which portion of the screen or window are the other participants viewing?

Based on visual cues, the above information can be gathered if the participants are physically collocated during a face-to-face interaction. However, for computer-enabled collaboration, these cues are missing and need to be shown explicitly to provide multiple channels of information to the end user. To complete the sense of "telepresence", the collaboration systems also need to be aware of different interaction styles.

4.3.2 Interaction Styles

Collaboration systems should support the following basic interaction styles:

Controlled-Style

In a controlled-style interaction, the participant in control determines who will be acting on the collaboration services or tools. Any participant who hopes to act on the collaboration services or tools has to request for permission from the participants in control, who will decide whom to grant the meeting floor based on certain algorithms.

Free-Style

In a free-style interaction, all participants have the meeting floor and thus can act freely during the collaboration session.

4.3.3 Controlled-Participant-Identification (ISWYD)

In a controlled-style interaction, only the individuals with permission have control of the floor and by extension control the pointing device used by the computing platform. Therefore, it is essential that each interaction participant know who has the permissions to act on the collaboration services and tools at any particular moment. To relieve the participants' overhead of trying to memorize who has the meeting floor, it is suggested to label the pointing device with the speaker's name, type of computing platform and network connection.

In a free-style interaction, all participants have the floor to control the pointing devices freely. In addition, as the number of participant increases, it becomes virtually impossible for each participant to know what is happening. Controlled-Participant-Identification mechanism is introduced to address this specific problem. Whenever a participant is moving the mouse or pointing device, a small information bar with the controller's name, the type of the platform and other short messages is displayed in the location of the pointing device. This information bar is called cur-bar in this research, and it can be customized in two aspects as described below.

4.3.3.1 Behavior-Customizable

Whether the cue-bar is displayed or not can be predetermined by the participant who sets up the collaboration session. Furthermore, actor (speaker) and observers (receivers) can jointly customize the cue-bar.

In a controlled-style interaction, there are three possibly customizable behaviors, with the corresponding logic shown in the activity diagram Figure 35.

- If the participant in control mandates that the cue-bar be displayed, other participants will see this bar all the time; therefore, the actors/speakers' actions will be shown to all the other participants.
- If the participant in control mandates that the cur-bar not be displayed, other participants will never see this bar; thus all actions are fully anonymous.
- If the participant in control only recommends the cue-bar, participants can customize the cue-bar behavior. If a speaker does not want other participants to know what she is doing, she could turn off the cue-bar. Similarly, if an observer is not interested in what others are doing, he could turn off the cue-bar as well.

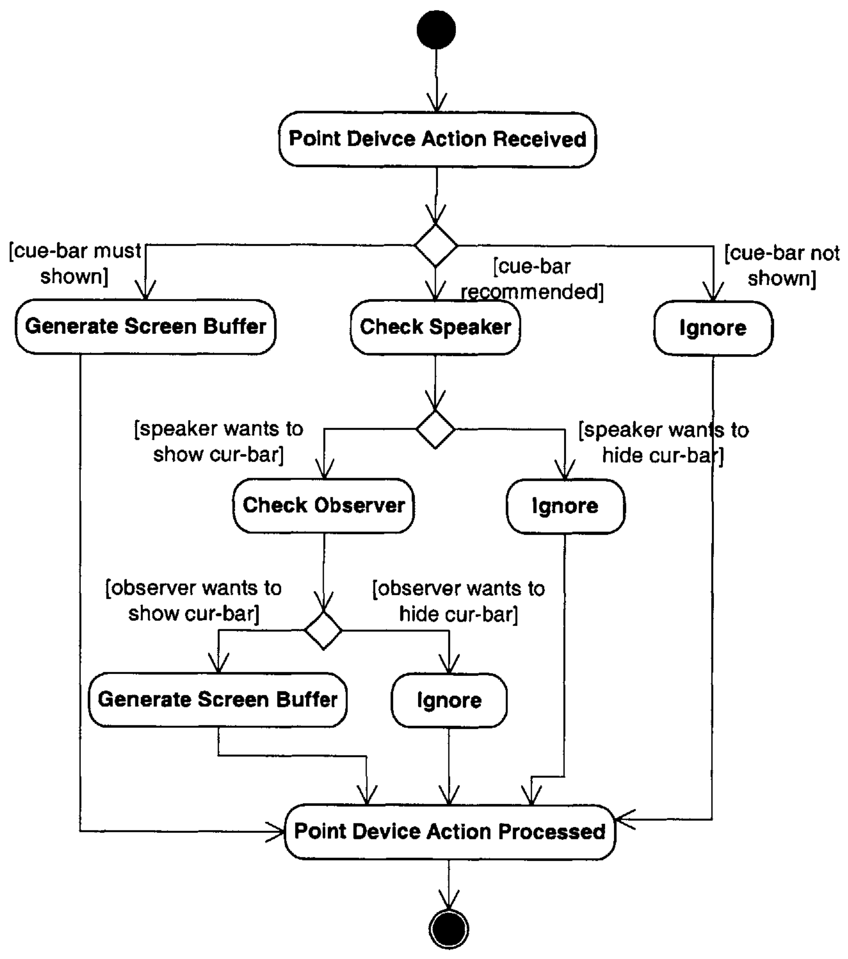

Point Delvce Action Received
[cue-bar must shown]
[cue-bar recommended]
[cue-bar not shown]
Generate Screen Buffer
Check Speaker
Ignore
[speaker wants to show cur-bar]
[speaker wants to hide cur-bar]
Check Observer
Ignore
[observer wants to show cur-bar]
[observer wants to hide cur-bar]
Generate Screen Buffer
Ignore
Point Device Action Processed

Figure 35 – Activity Diagram for Controlled-Participant-Identification

In a free-style interaction, each participant can freely determine the cue-bar behavior. Different from that of controlled-style interaction, if a participant hides the cue-bar, it might either be because the participant is not interested in who is controlling the pointing device, or because the participant does not want other participants to know that she is currently controlling the pointing device.

4.3.3.2 Context-Customizable

Another feature of the cue-bar is that both message content and format are customizable. The message content is composed of several sections, including the participant name or ID, the type of computing platform, the network connection, and other user-defined information. The format for message display on the cue-bar can be flexibly specified during each session. All sections in the message may share the same format, or each section can have its own format. The whole message can be specified as one string, in which the message content placeholders and message format tags are mixed. The syntax for building this string is similar to HTML-based syntax mechanism.

The basic syntax is as following:

```
<font size=n face=fontface color=fontcolor> Message Placeholder </font>
```

In addition, the following attributes can be used to specify message format:

Table 3 - Special Attributes for Cue-Bar Message Format

attributes	*Meaning*
`<b> ... </b>`	Boldface
`<i> ... </i>`	Italics
`<sup> ... </sup>`	Superscript font
`<sub> ... </sub>`	Subscript font
`<s> ... </s>`	Strikeout
`<u> ... </u>`	Underline

Figure 36, Figure 37, Figure 38 and Figure 39 illustrate the concepts of Controlled-Participant-Identification. Consider the following scenario describing a collaboration session involving four participants sharing an AutoCAD® drawing in a collaboration session. *Feniosky* is using a web browser on his computer, while *Chang*, *Sanjeev* and *Mary* are participating in this collaboration session from their Window CE® device, phone and pager respectively. Figure 36 shows that *Feniosky* is in control of the session, and he mandates the cue-bar be shown for each participant. Thus a cue-bar with his name and the platform he is using is displayed for all the participants. When *Chang* controls the session, the cue-bar changes to display *Chang*'s name and his platform, as shown in Figure 37. *Feniosky*, *Sanjeev* and *Mary* will also see a similar cue-bar. Figure 38 and Figure 39 illustrate the scenarios when *Sanjeev* and *Mary* are controlling the session.

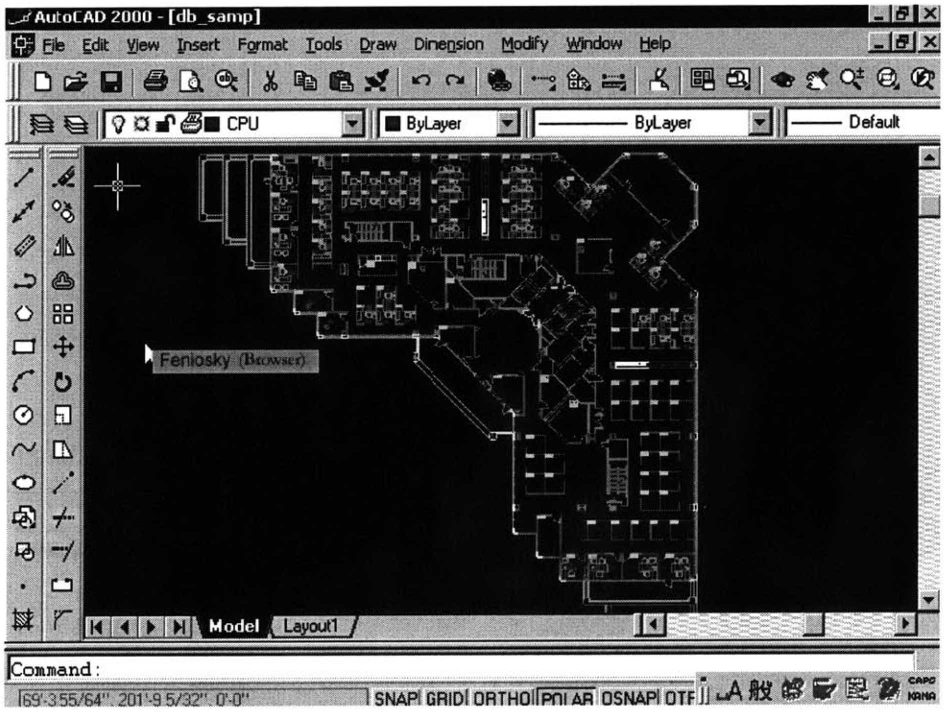


Figure 36 - ISWYD: Participant Feniosky controls the session from a browser

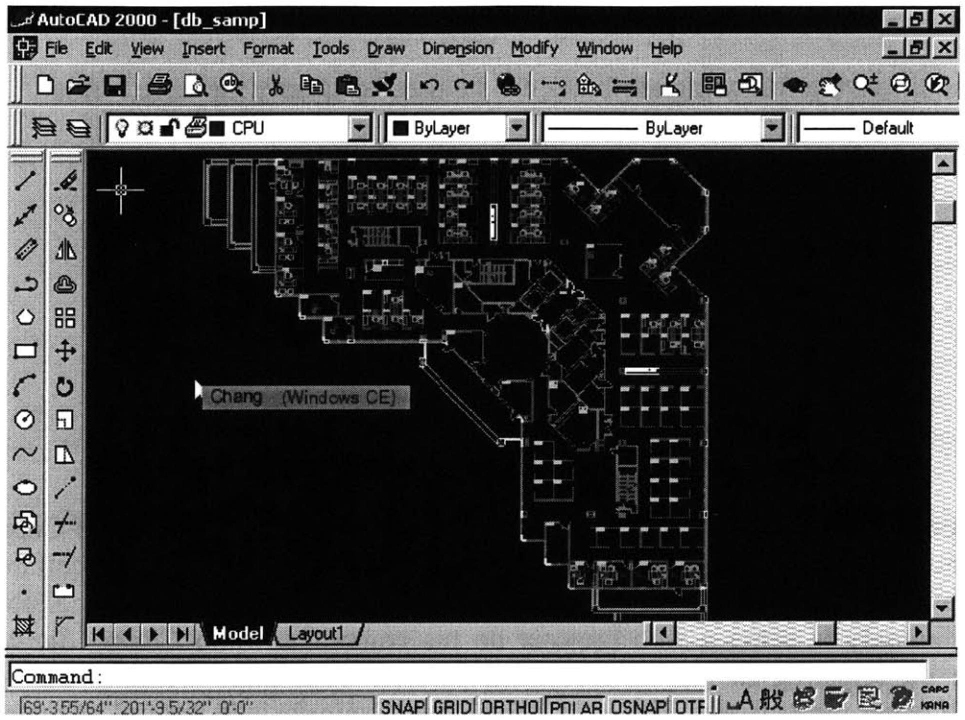


Figure 37 - ISWYD: Participant Chang controls the session from his Windows CE®

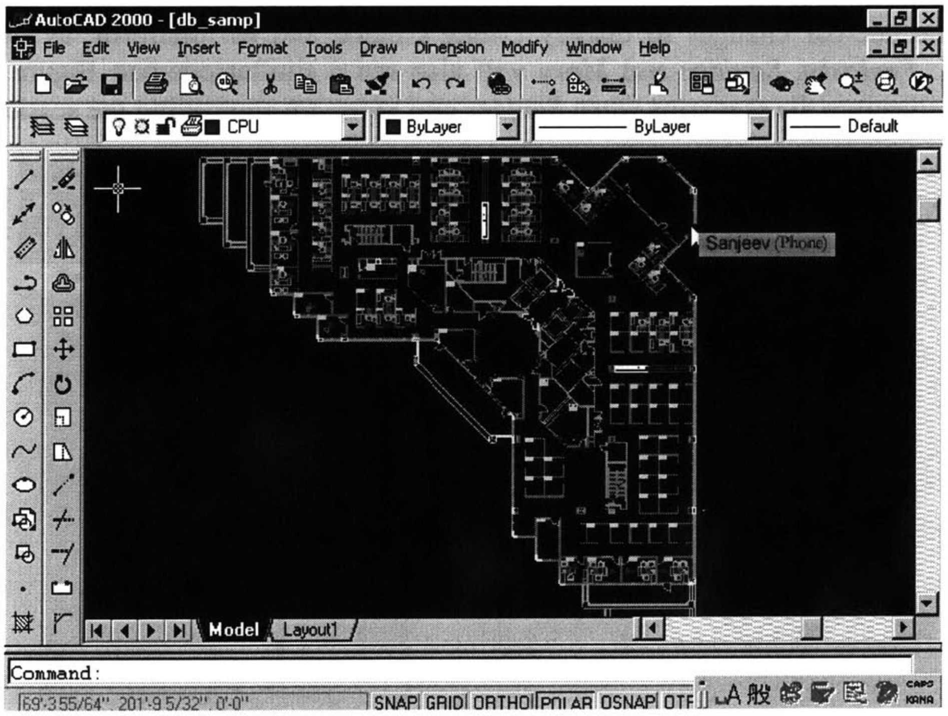


Figure 38 - ISWYD: Sanjeev controls the session using his phone

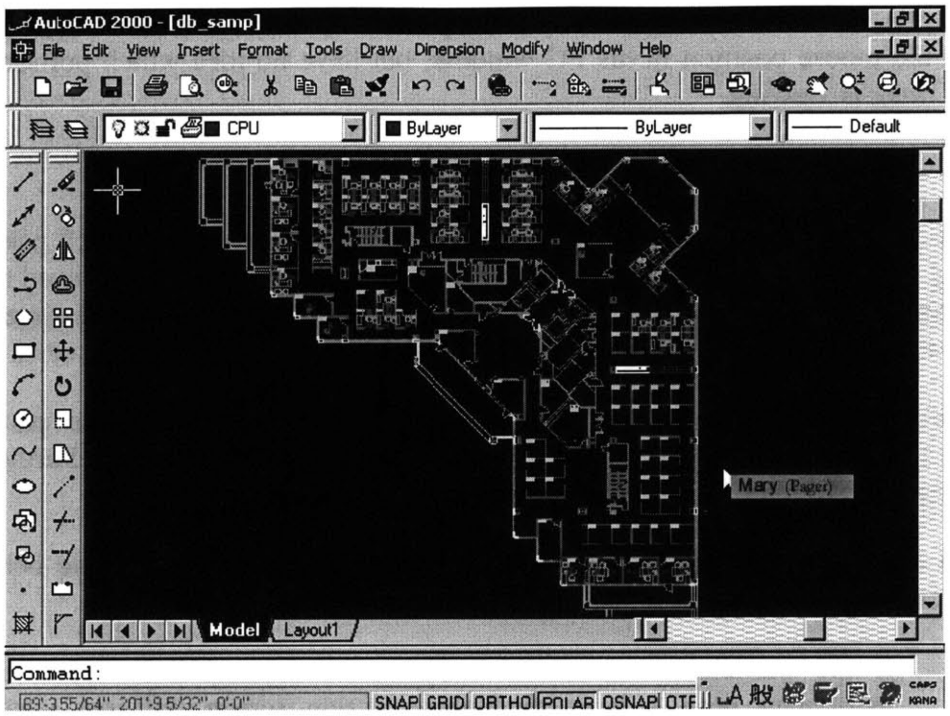

AutoCAD 2000 - [db_samp]
File Edit View Insert Format Tools Draw Dimension Modify Window Help
CPU
ByLayer
ByLayer
Default
Mary (Pager)
Model
Layout1
Command:
SNAP GRID ORTHO POLAR OSNAP OTRACK

Figure 39 - ISWYD: Mary controls the session using her pager

4.3.4 Issues Related To Different Screen Sizes And Resolutions

Typically, participants use computing platforms with viewing screens that have different sizes and resolutions (also referred henceforth as screen real estate). Table 4 shows the typical data for desktop computers, laptops and handheld computers.

Table 4 - Typical Screen Size and Resolution of PCs, Laptops and PDAs

Platform	***Screen size***	***Resolution***
Desktop	15", 17", 19", 21"	800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024, 1600 × 1200
Laptop	12.1", 13.3", 14.1", 15.5"	800 × 600, 1024 × 768
Palm OS®	Palm III/V™: 4.7" × 3.2" Palm VII™: 5.25" × 3.25"	160 × 160
Windows CE® (HPC)	Compaq iPaq H3650™: 5.11" × 3.28" HP Jornada 545™: 5.2" × 3.1" CASIO Cassiopeia E-115™: 5.2" × 3.25"	240 × 320
Phone*	2.5" × 4.6"	96 × 128, 96 × 54
Pager*	3.5" × 2.5"	150 × 60

** Screen resolution data excerpted from Sun's J2ME Wireless Toolkit*

The differences among screen real estates introduce a unique constraint to the quality of collaboration session. Using the same scenario, *Feniosky* (with a desktop screen) can view and control the whole AutoCAD® window without scrolling, but *Chang* (using a Window CE® device) can only view a small area of the whole drawing. When

Chang scrolls his window to view a specific portion of the overall drawing, *Feniosky* might get the mouse position in his PC window, but have no idea which portion of the drawing *Chang* is viewing.

4.3.5 Port-of-View (ISWYS)

Port-of-View solution is designed to reveal the portion of window each participant is viewing. For large screens with high-resolution, bounding boxes are displayed to show the corresponding viewport of each smaller screen with low-resolution. The time period to display the bounding boxes could be tuned in advance or set by the participants.

Similar to the Controlled-Participant-Identification mechanism, the participant setting up the collaboration can set the Port-of-View behavior in advance. Furthermore, the actor (speaker) and observers (receivers) can jointly set the behavior.

In a controlled-style interaction, there are three possibly customizable behaviors, with the corresponding logic shown in the activity diagram Figure 40:

- If the participant in control mandates a Port-of-View behavior, each participant with a large or high-resolution screen will see what other participants with small or low-resolution are viewing.
- If the participant in control mandates that Port-of-View behavior not be used, no participant will be able to see what the other participants are actually viewing.
- If the participant in control recommends a Port-of-View, observers can customize this behavior. If an observer is not interested in the information, he could turn off the Port-of-View independently of the speaker decision to show the information or not.

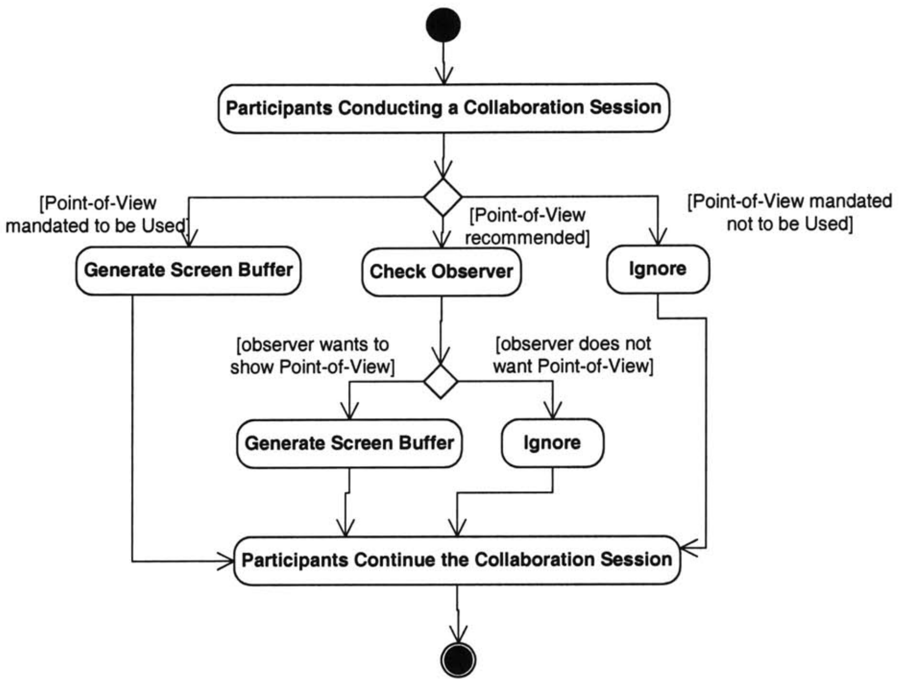
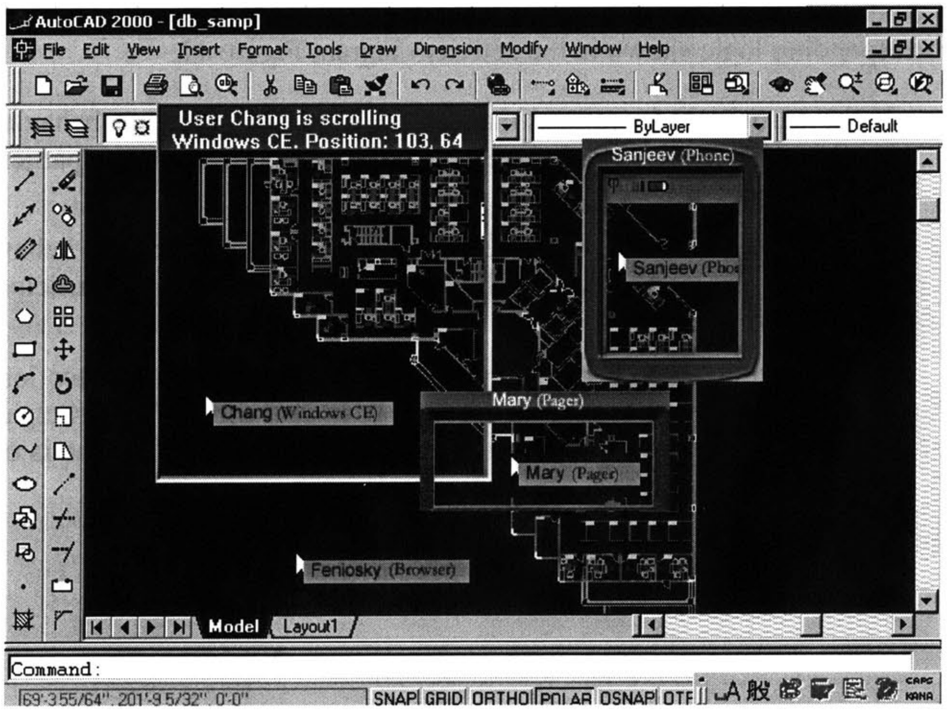

Participants Conducting a Collaboration Session
[Point-of-View mandated to be Used]
[Point-of-View recommended]
[Point-of-View mandated not to be Used]
Generate Screen Buffer
Check Observer
Ignore
[observer wants to show Point-of-View]
[observer does not want Point-of-View]
Generate Screen Buffer
Ignore
Participants Continue the Collaboration Session

Figure 40 – Activity Diagram for Point-of-View

In a free-style interaction, each participant can freely determine the Port-of-View behavior. Different from that of controlled-style interaction, if a participant hides the Port-of-View, it might either be because the participant is not interested in what the others are viewing, or because she does not want other participants to know what she is viewing.



User Chang is scrolling Windows CE. Position: 103, 64

Figure 41 - Port-of-View

Continuing the scenario from the earlier sections, Figure 41 illustrates Port-of-View visible on the screen of *Feniosky*'s desktop. *Feniosky* sees what *Chang* is viewing in his Windows CE® screen, what *Sanjeev* is viewing from his phone, and what *Mary* is viewing from her pager.

4.3.6 Focus-of-Attention (IMYSWIS)

Realizing the constraint introduced by the differences among screen real estates, Port-of-View mechanism can be used to reveal the viewing portions of all participants. However, we need another mechanism to force all participants to focus on the same portion of the window. Continuing the scenario, when *Feniosky* moves the mouse to a new position, *Chang* might not know where the mouse pointer is if that pointer is actually out of range from Chang's screen.

Focus-of-attention mechanism is introduced to address this specific issue. In this mechanism, the screen-scrolling events and events from the pointing devices of all participants are captured and this information is used to synchronize each participants' viewport accordingly. Whenever a participant with larger or high-resolution screen views or manipulates a portion of the window, the mechanism will automatically scroll the window for the participants with smaller screen real estate, so that every participant will see the same area.

Similar to the Controlled-Participant-Identification and Port-of-View mechanisms, the participant setting up the collaboration can set the Focus-of-Attention behavior in advance. Furthermore, the actor (speaker) and observers (receivers) can jointly set the behavior.

In a controlled-style interaction, there are three possibly customizable behaviors, with the corresponding logic shown in the activity diagram Figure 42:

- If the participant in control mandates a Focus-of-Attention behavior, each participant will be forced to view the same region where the participant in control is viewing.
- If the participant in control mandates that Focus-of-Attention behavior not be used, no participant will be able to force other participants to synchronize their viewports.
- If the participant in control only recommends a Focus-of-Attention, other participants can customize this behavior in real-time. If a participant does not want to be forced to view that region, he can stay where he is or freely scroll to other regions.

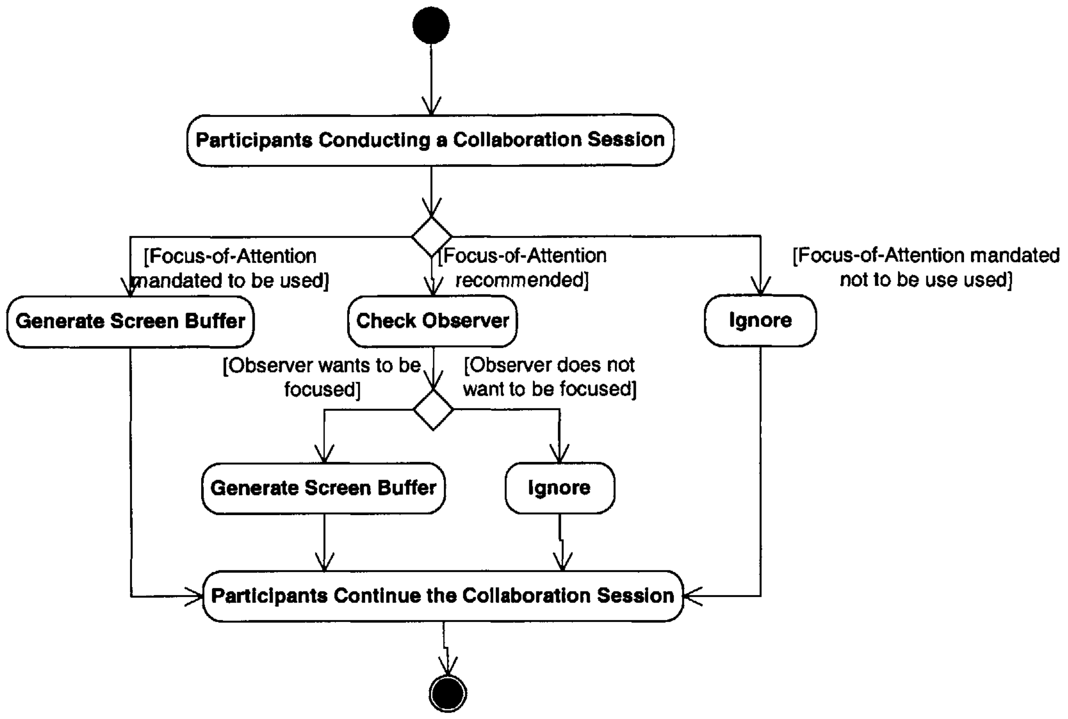

Participants Conducting a Collaboration Session
[Focus-of-Attention mandated to be used]
[Focus-of-Attention recommended]
[Focus-of-Attention mandated not to be use used]
Generate Screen Buffer
Check Observer
Ignore
[Observer wants to be focused]
[Observer does not want to be focused]
Generate Screen Buffer
Ignore
Participants Continue the Collaboration Session

Figure 42 – Activity Diagram for Focus-of-Attention

In a free-style interaction, each participant can freely determine the Focus-of-Attention behavior. Thus, each participant will have the freedom to decide whether to view the same region with other participants, or view her own region of interest.

There are two manifestations of the Focus-of-Attention mechanism. To illustrate the first one, continuing the scenario from earlier mechanisms, Figure 43 illustrates what is happening when *Feniosky* requests a Focus-of-Attention. Screens of all the other participants will be scrolled to a region of the AutoCAD® drawing such that the center of that portion is where *Feniosky* puts his mouse pointer. A second manifestation of the Focus-of-Attention mechanism involves *Feniosky* identifying a particular area in the AutoCAD® drawing as the Focus-of-Attention. In this case, screens of all the other participants will be scrolled to the correct location, with the region of interest shrunk or expanded to fit the particular screen real estate.

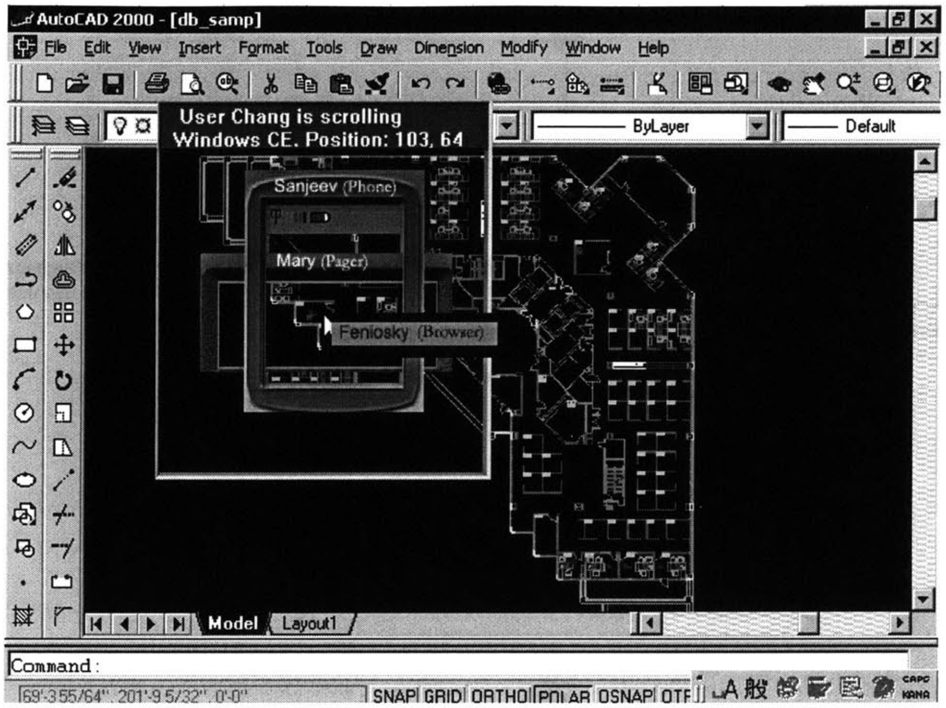

AutoCAD 2000 - [db_samp]
File Edit View Insert Format Tools Draw Dimension Modify Window Help
User Chang is scrolling
Windows CE. Position: 103, 64
Sanjeev (Phone)
Mary (Pager)
Feniosky (Browser)
ByLayer
Default
Model
Layout1
Command:
SNAP GRID ORTHO POLAR OSNAP OTR

Figure 43 - Focus-of-Attention

In conclusion, this research has identified three mechanisms to reduce participants' overhead and increase overall efficiency of a computer enabled collaboration session. Specifically, the research has highlighted the motivation as well as implementation details for the following three mechanisms:

- Controlled-Participant-Identification (ISWYD, **I** See **W**hat **Y**ou **D**o)
- Port-of-View (ISWYS, **I S**ee What **Y**ou **S**ee)
- Focus-of-Attention (IMYSWIS, **I** Make **Y**ou **S**ee **W**hat **I S**ee)

Chapter 5

Making Applications Collaboration-Ready

This chapter discusses an important mechanism to make virtually any applications collaboration-ready. Server-side computing is the suitable solution, in which the applications are running in the server, and clients get the screens from the server. The participants' mouse and keyboard actions from the clients are intercepted and sent to the server, which executes those actions and sends the screen update back to clients.

5.1 Server-Side Computing

Collaboration system should be able to support collaboration services like text-chat, whiteboard, audio-video, real-time voting, and application sharing. This chapter is focused on application sharing, an important means for sharing applications across multiple clients, including desktop client, PDA client, phone client and pager client.

5.1.1 Necessity of Server-Side Computing

Application sharing is crucial for running applications by multiple users upon heterogeneous platforms. Consider the following case: in the fire-fighting session, field agents entered a building and searched for victims. He used a Windows CE to collaborate with the control center, where people use desktop PCs to issue commands and directions. The field agents and the control center collaboratively work on an AutoCAD drawing showing the floor plans of the building. It seems impossible that AutoCAD can be run in the Windows CE device, which has little processing power and limited screen real estate. This assertion is right in the sense that the colossal AutoCAD cannot run on Windows CE, but ignores the fact that the program can be run in a central server. The server will accept stylus and keystroke events from the Windows CE, insert these events into AutoCAD's event queue, which will process them as if the events are generated from a local user. The server keeps polling the whole screen, and sends any update back to the Windows CE client. In this way, the field agent can fully gain control of the AutoCAD drawing, as if he is running such an application.

Server-side computing is necessary for any practical collaboration systems, which should support sharing applications or files coupled with specific applications on multiple platforms. The following situations are to be fulfilled with server-side computing:

- Some applications can only run on limited families of computer hardware or operating systems. For example, Microsoft Office series, like Word, Excel, Powerpoint, can only run on Windows and Mac OS, but not available on UNIX.
- Application files do not reside on all the clients. It is common that a Powerpoint slide is only stored in a client. Hence, other clients might will not be able to view and modify this slide. Even if each client has the same file, they cannot collaboratively work on it. When one client modifies the slide, the other clients will not be able to see it because each client is working on his/her own copy. Therefore, centralized server operation to the slide is a must to ensure that each client will be synchronized with up-to-date information.
- For people who are using their PDAs for collaboration, application sharing is especially important, because normally they do not have the application in hand, either because of the limited power of PDAs, or simply because there is no such software for that platform.

Application sharing will virtually make any application collaboration-ready by employing a technology called server-side computing. Server-side computing can be easily deployed from a server instead of running on the local client machine and thus avoiding a number of problems. Server-based computing does not require applications to be downloaded to client devices. As a result, application performance is neither bandwidth- nor device-dependent.

5.1.2 Advantages of Server-Side Computing

Due to the inherent advantages of server-based computing and the distributed nature of the virtual design teams, the architecture of server-side computing system can be summarized in the following two critical components [34]:

- A multi-user operating system that allows multiple concurrent users to log on and run applications in joint as well as separate, protected sessions on a single server.

- A remote presentation services architecture capable of separating the application's logic from its user interface, so that only keystrokes, mouse clicks, and screen updates travel the network.

This new model of running applications solves a diverse set of challenges [35]:

- **LAN-Based Applications**. Applications based on two-tier client/server architectures are designed for the LAN and are not optimized to run over high-latency phone or WAN connections that run 100 to 1000 times slower than a local segment.
- **Heterogeneous Clients**. Networks usually involve PCs as well as non-Windows systems such as Linux, OS/2, UNIX, or Macintosh. Other networks could include low-cost, fixed function devices, such as terminals or wireless devices such as PDAs.
- **Heterogeneous Software Packages.** Each participant within a collaboration session might use a set of software packages that is specific to his/her won discipline. Therefore compatibility becomes an issue for situations such as visualization.

5.2 High Level Solution

A good alternative to building applications that directly support distributed and shared usage is to add a multi-user extension to existing applications. The principle on which this is strategy is based is: "If all involved applications receive the same input and data in the same order, they will present the same behavior to all users". Therefore in order to convert applications from single-use to support multi-user shared usage, an extension is added to the original application that can access the functionality of the original application, so by catching input from each user and sending the information to the other users, the output is recreated on the other users computers. One of the advantages of this solution is that it can be less complex, and allow for tuning to the circumstances, both with respect to user interface and communication characteristics. It has a great potential, since it also allows for collaboration awareness.

The objective of the multi-user extensions that were developed was to add distributed and shared usage to existing application that had been designed for single users. As explained previously, the principle on which the extensions are based is very simple, by giving all the participants the same input; the output for all will be equal. Each distributed application is listening for a specific set of events and once these events are triggered, they are sent over the network to the other distributed applications, thus obtaining synchronization. In other words, by catching input from each user and sending the information to the others, the output on the other users computer is recreated. The following figure shows basic components of the multi-user extension with a simple case of three users [36].

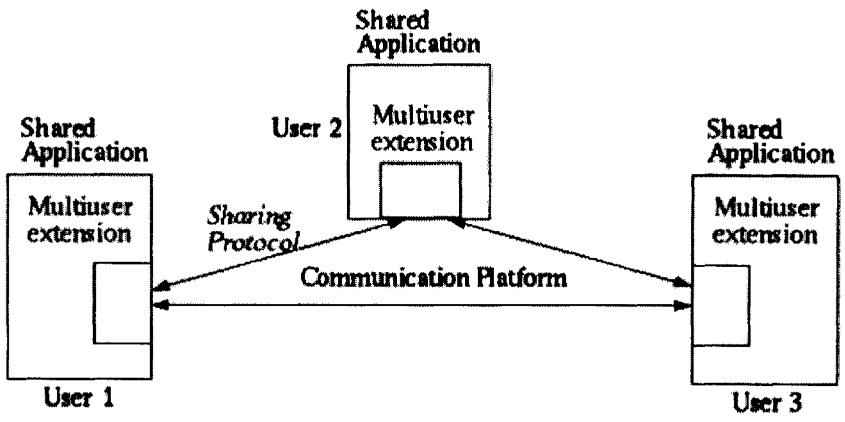


Figure 44 - Multi-User Extension Architecture

One of the advantages of this solution is that it can be less complex to implement, and because a multi-user extension must be developed for each application, it allows for some customizing according to the circumstances. For example the user interface as well as communication characteristics can be designed for each specific application. In this sense it has a great potential, since the application is in a certain way is 'aware' that it is used in a distributed fashion. And new collaboration-specific functionality can be added to match the needs of the distributed team that is using the application. This is an important difference with the Shared X solution explained in the next few sections, where the application is deployed 'as is' with no possibility of modifying its functionality.

5.3 Virtual Network Computing

The Virtual Network Computing (VNC) project is not the only network computing system. Other thin-client systems include those built around the Citrix ICA protocol (for example, Citrix's Winframe [34] and Insignia Solutions' Ntrigue [37]), those built around Microsofts Remote Display Protocol (for example SCO's Tarantella [38], and Datalight's ThinSystem [39] as well as Microsoft's own Windows based Terminal Server) and those built around the X protocol such as Graphon's RapidX [40] and Bridges. The problem with all of these systems is that, unlike X, they use proprietary protocols, so reliable information about them is difficult to obtain. Citrix's ICA protocol is a popular mechanism for remote interaction with PCs, but it appears to be closely tied to the Microsoft Windows GUI, so it may not be an ideal general-purpose remote display protocol. Microsoft has developed its own protocol, T.128 previously known as T.Share, based on the ITU T.120 protocol. The objective of the VNC project is that the protocol, or something similar to it, may become an open cross-platform standard for very thin-client computing.

In the VNC system, server machines supply an entire desktop environment that can be accessed from any Internet connected machine using a simple software NC. Whenever and wherever a VNC desktop is accessed, its state and configuration (right down to the position of the cursor) are exactly the same as when it was last accessed. In addition, VNC allows a single desktop to be accessed from several places simultaneously, thus supporting application sharing in the style of computer supported cooperative work (CSCW).

The technology underlying VNC is a simple remote display protocol. It is the simplicity of this protocol that makes VNC so powerful. Unlike other remote display protocols such as the X Window System and Citrix's ICA, the VNC protocol is totally independent of operating system, windowing system, and applications [41].

5.3.1 The VNC Protocol

The technology underlying the VNC system is a simple protocol for remote access to graphical user interfaces. It works at the frame buffer level and therefore applies to all operating systems, windowing systems, and applications—indeed to any device with some form of communications link. The protocol will operate over any reliable transport such as RS232, firewire, USB, modems and IrDA, anything which gives a reliable two-way connection. At present TCP/IP is the only protocol implemented, because it's convenient, ubiquitous, and easy to route. The endpoint with which the user interacts (that is, the display and/or input devices) is called the client viewer. The endpoint where changes to the frame buffer originate (that is, the windowing system and applications) is known as the server (Figure 45) [41]. VNC is truly a "thin-client" system. Its design makes very few requirements of the client, and therefore simplifies the task of creating clients to run on a wide range of hardware.

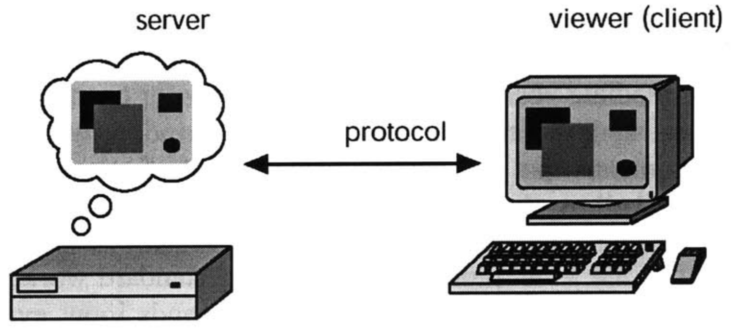


Figure 45 - Thin Client System Implemented by VNC

The display side of the protocol is based on a single graphics primitive: Put a rectangle of pixel data at a given x, y position. At first glance this might seem an inefficient way to draw some user interface components. However, allowing various encoding schemes for the pixel data gives a large degree of flexibility in trading off parameters such as network bandwidth, client drawing speed, and server processing speed. The lowest common denominator is the so-called raw encoding, where the pixel data for a rectangle is simply sent in left-to-right scanline order. All VNC clients and servers must support this encoding. However, the encoding actually used on a given connection can be negotiated according to the capabilities of the server and client and the connection between them. For example, copyrectangle encoding is very simple and efficient, and can be used when the client already has the same pixel data elsewhere in its frame buffer. The encoding on the wire is simply an x, y coordinate. This gives a position in the frame buffer from which the client can copy the rectangle of pixel data. This encoding is typically used when the user moves a window across the screen or scrolls a window's contents. Most clients will support copyrectangle encoding, since it is generally easy to implement, saves bandwidth, and is likely to be faster than sending raw data again. However, in a case where a client cannot easily read back from its frame buffer, the client could specify that it should not be sent data encoded this way. A typical workstation desktop has large areas of solid color and text. One of our most effective encoding takes advantage of this phenomenon by describing rectangles consisting of one majority (background) color and "subrectangles" of different colors. There are numerous other possible schemes. JPEG encoding could be used for efficient trans mission of still images or an MPEG encoding for moving images. A pixel-data caching scheme could efficiently encode multiple occurrences of the same text character by referring to the first occurrence.

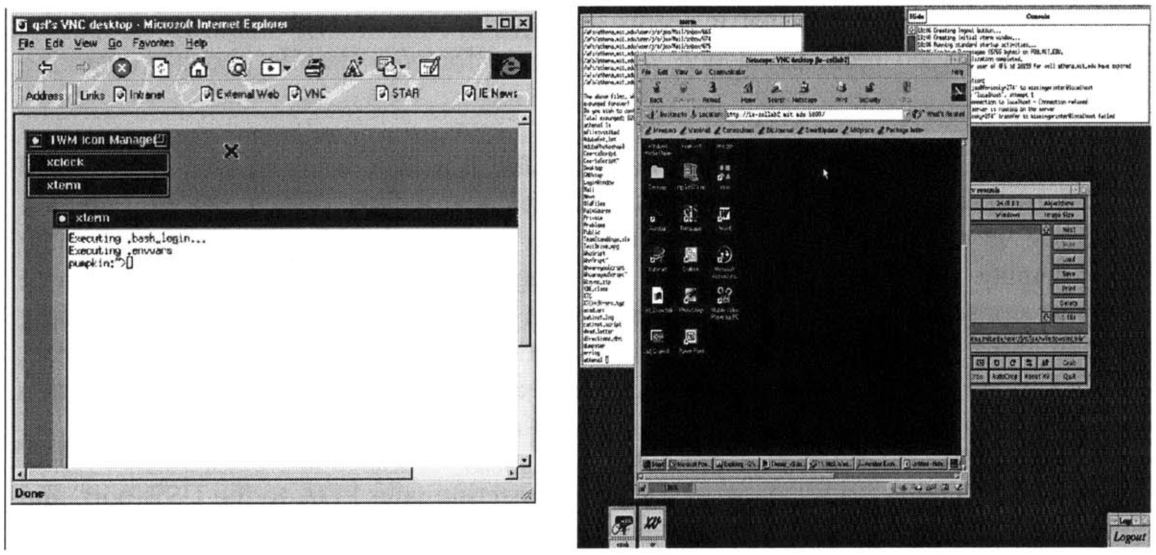
A set of rectangles of pixel data makes a frame buffer update (or simply, update). An update represents a change from one valid frame buffer state to another. In this sense, an update is similar to a frame of video. It differs, however, in that it usually affects only a small area of the frame buffer. Each rectangle may be encoded using a different scheme. The server can therefore choose the encoding most appropriate for the particular screen content being transmitted and the available network bandwidth. The update protocol is demand-driven by the client. That is, an update is only sent by the server in response to an explicit request from the client. All screen changes since the client's last request are coalesced into a single update. This gives the protocol an adaptive quality: the slower the client and the network, the lower the rate of updates. On a fast network, for example, as the user drags a window across the screen it will move smoothly, being drawn at all the intermediate positions. On a slower link—for example, over a modem—the client will request updates less frequently, and the window will appear at fewer of these positions. This means that the display will reach its final state as quickly as the network bandwidth will allow, thus maximizing the speed of interaction.

The input side of the VNC protocol is based on a standard workstation model of a keyboard and multi-button pointing device. The client sends input events to the server whenever the user presses a key or pointer button, or moves the pointing device. Input events can also be synthesized from other nonstandard I/O devices.

To establish a client-server connection, the server first requests authentication from the client, using a challenge-response scheme, the client typically requires the user to enter a password at this point. The server and client then exchange messages to negotiate desktop size, pixel format, and encoding schemes. The client requests an update for the entire screen, and the session begins. Because of the stateless nature of the client, either side can close the connection at any time without adverse consequences.

5.3.2 The VNC Viewer

The viewer was designed to be as simple as possible, as indeed it should be for any thin-client system. It requires only a reliable transport (usually TCP/IP), and a way of displaying pixels (either writing directly to the frame buffer or going through a windowing system). The available viewers are: an X-based viewer (which runs on Solaris, Linux, and Digital Unix workstations), a Win32 viewer that runs on Windows NT and 95, and a Java applet that runs on any Java-capable browser, a Windows CE viewer that runs on SH3 and MIPS processors and a Palm OS viewer that runs on the IBM Workpad, Palm Pilot Professional, Palm III and Palm V. The images in Figure 46 show a variety of X and Windows desktops being accessed from both Java and native X and Windows viewers.


qsf's VNC desktop - Microsoft Internet Explorer
File Edit View Go Favorites Help
Address Links Intranet External Web VNC STAR IE News
TWM Icon Manager
xclock
xterm
xterm
Done
Logout

```
Executing .bash_login...
Executing .envvars
pumpkin:~>
```

Figure 46 - Unix Desktop Within IE and Windows Desktop Within Netscape on Unix

5.3.3 The VNC Server

The VNC server architecture is more complex than the viewer. Because the protocol is designed to make the client as simple as possible, it is usually up to the server to perform any necessary translations (for example, the server must provide pixel data in the format the client wants). Servers have been written for two main platforms, X (that is, Unix) and Windows NT/95/2000. The X-based server was the first one developed. A single Unix machine can run a number of VNC servers for different users, each representing a distinct desktop. Each desktop is like a virtual X display, with a root window on which several X applications can appear. The Windows server is more complex since Windows has fewer places to insert hooks into the system to monitor display updates, and the model of multi-user operation is less clearly defined. The current Windows server simply mirrors the real display to a remote client, which means that only a single desktop is available from any one PC running the server [41].

5.4 Integration With Collaboration Systems

The VNC protocol's simplicity could allow it to be used on a much wider range of hardware. Some devices usually have a highly specialized user interface and typically employ customized physical display devices. This has traditionally prevented such interfaces from being mobile in the VNC sense of the word. But the usefulness of a remote display protocol can be extended so that users could, for example, bring up the controls for their video recorder on a mobile phone as they drive home from work, use a modem to dial a telephone answering machine and reprogram it through a graphical interface, display their car stereo or GPS receiver as part of the dashboard, regardless of the equipment brand installed. At present, such functions require the displaying device to have detailed knowledge of the remote system and to emulate that system's user interface or some alternative interface that it deems appropriate. For example, you would need a driver for your video recorder, which was designed for your mobile phone's operating system.

A much simpler approach would be to use the interface designed for and provided with the remote device, but to interact with it locally. For this, a set of common "phonemes" is needed with which to construct a variety of GUIs. This is the role that remote display protocols such as VNC can play. It is simple enough to implement cheaply in consumer electronics hardware, yet it can be used to describe the building blocks of most modern user interfaces. With standards such as IEEE-1394 Firewire, USB, and IrDA, the physical interface to connect a variety of devices is provided; with a remote display protocol, a standard for plug-and-play user interfaces can be added.

Imagine walking up to any workstation, connecting your PDA to the USB port, and having the PDA applications instantly available on the workstation screen, or plugging your PDA into your car and having the engine management unit display servicing information on the PDA's screen. And imagine that this works for any workstation, any PDA, any car. The engine management example illustrates an important point: A standardized GUI protocol allows devices that have no physical display of their own to provide graphical information when such a display becomes available to them. Your PDA could, perhaps, shrink to the size of a pen if it could access a display and keyboard through an IrDA link. And yet this "microPDA" could still display PowerPoint-style presentations when in the vicinity of an LCD projection panel or a large TV .

This model is very similar to the Web, where services without an I/O capability of their own wait for a user to provide one in the form of a Web browser. The success of this strategy has led to embedding HTTP daemons in printers, switches, routers, and other devices. But to be a Web server, a device must at least have a TCP stack and an IP address. And to be a Web browser requires at least the ability to render fonts and parse HTML. In contrast, a remote display protocol requires only a reliable transport medium and the simplest of display capabilities. And while a page of HTML will generally require the transmission of fewer bytes than its remote display protocol equivalent (i.e. the VNC protocol), the latter is infinitely more flexible [41].

5.4.1 Annotation

An important aspect outlined in the requirements is the enhancing the collaboration efficiency and minimizing collaboration overhead. One of the key issues is to provide awareness of other members' actions and to provide an additional communication channel while sharing an application, as we have already argued in Chapter 4. An application launched through the collaboration system allows its users to communicate graphically on top of the shared data through the annotation board.

The annotation functionality can be seamlessly integrated into application sharing service by adapting VNC protocol so that the VNC server will draw the annotations like pointers, labels, texts and underscores in its screen buffer, and send the buffers to each client; in this way, all the clients will be able to view the annotations whenever a participant who has permission and is annotating.

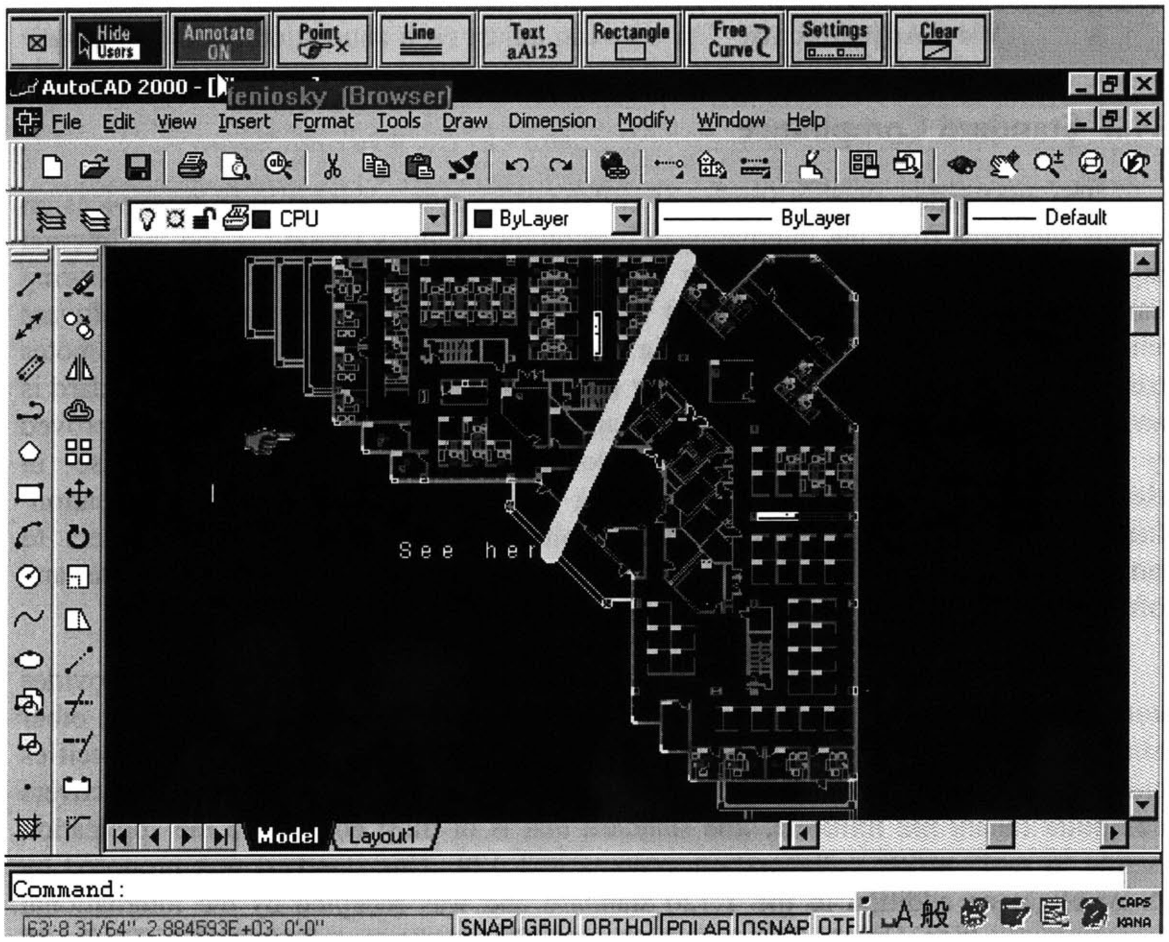


Figure 47 - Annotation Interface

The annotations capabilities including text, underlining, pointers and labels are shown in Figure 47. The desired features of the annotation board are:

- **Multi-Document Interface.** The annotation board contains a multi-document interface such that a user can work on more than one annotation document at a time. A document can be created, opened or saved at any time. Also, figures can be cut/copied/pasted between annotation documents. The user can browse through multiple annotation documents with a tab.
- **Variety of Figures Available.** A selection of figures comes with the whiteboard. A user can easily construct rectangles, lines, arrows, curves, free-hand drawings, images and text within each whiteboard document.
- **Figure Properties Modification.** A user can alter a figure's properties at any time. For instance, a rectangle could be modified such that its color is red and its outline is a thick, blue, dashed line. The appearance of text could also be adjusted by changing its font, font size, and color. Figures can also be grouped/ungrouped along with being ordered to appear in front of or behind other figures.
- **View Adjustments.** The user can adjust each annotation document's view such that it can be scrolled horizontally or vertically and zoomed in or out.

5.4.2 Standard Compliance

The application sharing service of collaboration systems must base its services on various standards from the International Telecommunications Union [42] and the Internet Engineering Task Force [43] in order to guarantee users participating in conferencing session interoperability over different types of networks, connections and software vendors. The International Multimedia Teleconferencing Consortium [44] promotes interoperability and is a good resource for ITU specifications. The ITU T.120 standard is made up of a suite of communication and application protocols developed and approved by the international computer and telecommunications industries for use in teleconferencing applications. By complying with the T.120 standards, Application-sharing service provided by multi-server collaboration systems can assist users to participate in conferencing sessions over different types of networks, connections and software vendors.

The T.120 standard covers document conferencing (data sharing) and provides protocols for establishing and managing data flow, connections and conferences; other standards in the suite cover specific facilities for data conferencing. T.126 specifies multipoint still image and annotation for whiteboard applications. T.127 covers multipoint binary file transfers. The standard that is of most concern to the application sharing in multi-server collaboration systems is T.128. This standard was proposed by Microsoft as an addition to the T.120 standard and was accepted by the International Telecommunication Union, Telecommunication Standardization Sector (ITU-T). T.128 specifies the program sharing protocol, defining how participants in a T.120 conference can share local programs. Specifically, T.128 enables multiple conference participants to view and collaborate on shared programs. The application sharing services must be T.128 compliant in the case that users might need to share their desktop or applications that are installed on the local hard drive instead of those available on the application server.

5.4.3 Security

The T.120 data conferencing systems appear difficult to accommodate with firewall packet filters and mayor features of this put the sharing server and all its resources, including its LAN, in serious jeopardy. The standard ITU T.120 services appear to not pose much risk to the sharing or the client systems. The shared whiteboard, which is part of the standard, allows exchange of information but no change of information can be made by a remote user to a shared system or other remote user. The file transfer standard will allow changing the destination host but it is low risk since the received files can be assigned to a specific directory. If senders could target some other directory they could overwrite critical system files. This is the danger posed by the collaborative application-sharing feature.

The feature that poses the most serious risk to network security is the collaborative application sharing. If remote users can take control of the application, the risk here is that any remote user has access to the full power offered by the application. If the program can write to the hard drive (e.g. Word or Netscape) then the remote user can save files to disk, possibly over writing critical system files. Other scenarios include hostile macros (like existing mail-borne macro viruses), downloading and executing destructive ActiveX or other applications with a web browser, inserting command.com into Word to get a command prompt and downloading network sniffers to harvest passwords. This gives remote users access to unprecedented destructive power on the shared system. From a Word document it is possible to remotely be able to get a command prompt and delete files from the sharing Windows NT system in about 10 seconds, probably not enough time for a network administrator to notice [45].

There may be ways of deploying a collaborative application sharing that reduces the risk it poses to the user and the enterprise. These rely on network-oriented approaches to hide the users from the attacks rather than trying to solve the inherent vulnerabilities of the deployment model.

5.4.3.1 Virtual Private Networks

The term "Virtual Private Network" (VPN) has gotten a lot of press recently and is marketed as the panacea for all networking ills; this is far from the truth. A VPN securely connects two or more disjoint networks over an intervening network. It might be used to connect different branch offices over the Internet, for example. The presumption is that the intervening network is hostile and by extension that the joining networks are secure.

If the networks in question are not secure, then the VPN only connects one unsecured network to another. If, for example, you use a VPN to connect a well-protected network to one with no protection whatsoever, you have just merged the unsecured net into the secure one, presenting all the vulnerabilities of the latter to the former. VPNs are useful only if you are connecting two secured networks, or at least two networks with the same security posture.

With some commonly available VPN-type tools such as Checkpoint's SecuRemote or MS PPTP, the network security improves but the application vulnerability remains unchanged. The reason is that this solution is an attempt to solve an application-level problem with a network-oriented solution. Note also that these solutions are PC-only, thus limited in its application range [45].

5.4.3.2 Encryption

Third party gateways exist which may help improve reliability, robustness, scalability and security. However, without end-to-end security it is possible for crackers on the remote end to infiltrate a "friendly" network or machine and simply use the gateway as a bridge inside a firewall and attack protected machines as described earlier. The gateway simply requires crackers to first break into machines we allow to connect to our gateway, without offering a complete solution.

A comprehensive security solution would require secure authentication, cryptography for all communication channels, and fine-grained control over the actions remote users can execute on the shared machines. This needs to happen all the way up to the application layer.

VNC used a random challenge-response system to provide the basic authentication that allows you to connect to a VNC server. This is reasonably secure; the password is not sent over the network. Once connected, however, traffic between the viewer and the server is unencrypted, and could be snooped by someone with access to the intervening network. Therefore, due to the cryptography requirement, it is important to "tunnel" the VNC protocol through some more secure channel such as SSH.

SSH normally just provides you with a 'Secure SHell', a login window to a remote machine. All traffic is encrypted between the two machines using public key encryption techniques. SSH has another advantage. It can compress the data as well. This is particularly useful if the link between the client and the server is a slow one, such as a modem, but even on a faster network it can help make up for the fact that the encryption takes a certain amount of time and so can slow the link down a little.

Fine-grained control over the users actions is the third component necessary to guarantee security. This was not implemented for this research but the most general solution would be to enforce read-only access to the server hosting the applications.

5.4.3.3 Corporate Networks and Firewalls

Unfortunately, the complexity of the implementation as well as compliance with the T.126 standard makes proxying the application's network traffic through a firewall very difficult. Further, the application sharing feature allows remote access to the shared user's desktop, as explained earlier, allowing arbitrary functions to be run such as file writing, deleting, launching other applications and downloading cracker tools. This application gives excessive control to remote connections making the user's desktop machine and networks resources highly vulnerable, as a result this type of applications are very unpopular among network administrators and security consultants.

5.4.4 Concurrency Control

In order to avoid consistency problems (and confusion) when multiple authors work in the Collaborative application sharing, the system should offer several levels of concurrency control:

- **Event notification.** Users need to be notified of changes made to objects by other users.
- **Fine-grained notification.** The event handler must be able to distinguish between write operations on the application (content) and on other attributes such as annotation.
- **User-controlled locking.** In addition to implicit locking within the scope and control of transactions, users must be able to lock objects during a long update that outlasts several short transactions.
- **Shared locking.** Locked objects must always remain readable, so users can get a common view of the information space.
- **Fine-grained locking.** While one user is updating the application (content) of a node, other users should still be able to annotate on it.
- **Persistent collaboration information.** Locks, events, and other information about the collaboration between users must be stored in a database. Clients need to be able to recover from server crashes and therefore need to have persistent collaboration information.

5.4.4.1 Locking

User-controlled Locking

User-controlled locking should be *shared, fine-grained* and *persistent.* While a user is writing a (long) node, the ongoing work will be saved frequently so as to not lose a lot of information in the event of a system crash. Meanwhile the user may wish to keep the node locked for the entire duration of the editing session.

Shared Locking

Allowing locked objects to be read enables users to browse an application containing locked objects. The only conflicts we wish to avoid are write-write conflicts. Being able to read the ongoing work of other writers can be helpful, for instance to view how one writer has created an object another writer wishes to use.

Fine-Grained Locking

Databases typically provide locking at the object level. Locking at the attribute level reduces the risk of conflicts. While one writer is adding to a node another writer may write to another node.

Since fine-grained locks may be held for only a short time (there's only so much updating one can do on a small part of an object) one may consider queuing lock requests instead of denying them.

Persistent Locking

Locks should be stored on both the client and the server side. When a server goes down and recovers, the clients should be able to reclaim the locks they had before the crash. When a client is restarted after a crash it should be able to retrieve the locks from the server.

5.4.4.2 Notification Control

Notification control allows users to be notified of important actions on the shared applications performed by other users. When you view a node currently locked by another user, your view should be updated each time that user saves updates to that node.

Event notification is usually asynchronous. Synchronous notification requires polling, which is very inefficient. When a node changes all readers receive an event, and will update their display asynchronously.

Individual users may wish to subscribe to events (or not). This places selection overhead on the server. For every operation the server must check which client applications have subscribed to that event. This overhead can be placed with the client by making the client ignore unwanted events (and letting the server send them anyway).

Fine Grained Notification

Notification of all operations on objects and attributes in objects should be supported, including notification of lock, unlock, delete and write operations on a find-grained level. Notification can only be reliably implemented at the same granularity level as locking.

Persistent Notification

Event information should be kept persistently in the shared workspace, for the same reason as persistent locking. If either the server or the client crashes, both should be able to recover the event subscriptions.

5.4.4.3 Transactions

Transactions in database systems serve three purposes:

- They are logical units that group operations comprising a complete task;
- They are atomicity units whose execution preserves the consistency of the database;

- They are recovery units that ensure that either all the steps enclosed within them are executed or none.

Short Transactions

Short transactions are needed for saving editing work while continuing a long editing session. Data is locked in different granularities to prevent other users from performing conflicting operations on the same set of data. In databases short transactions use locks to avoid read-write and write-write conflicts. In a shared workspace only write-write conflicts must be avoided.

Long Transactions

User-controlled locking combined with short transactions is a better solution to long updating sessions than long transactions. The logical, atomicity and recovery units are much smaller in user-controlled locking. User-controlled (shared) locking does not completely prevent other users from getting to resources over long periods of time. In user-controlled locking, users or applications must explicitly lock needed resources. Users must be aware of the multi-user situation.

Chapter 6

Making Participants Collaboration-Savvy

This chapter discusses the mechanism to make participants collaboration-savvy. The basic tool is a self & cross assessment. A web-based Questionnaire Service Provider is designed and implemented for this purpose.

6.1 Questionnaire Research

Questionnaire is an important means for people to gather information about other persons' beliefs, attitudes, behaviors, feelings, perceptions, motivations, or plans. Specially, in a collaborative environment, people need to collect information from the group to support group decision-making as well as evaluate the health of group collaboration. In this thesis the questionnaire for collaboration system is called **collaborative questionnaire**.

In information era, computers should greatly facilitate the questionnaire. **Electronic questionnaire** is such a new means of data collection. Meanwhile, database and network, the two underlying technologies for information technology, develop in an astounding speed, therefore **electronic collaborative questionnaire** draw more and more attention from researchers.

This chapter first describes the traditional questionnaire for social research, and advocates the wide application of electronic questionnaire (online questionnaire, or computer-aided questionnaire). Then it tries to give some ideas on characteristics of collaborative questionnaire and its electronic version. Finally it attempts to provide a framework for computer-aided collaborative questionnaire.

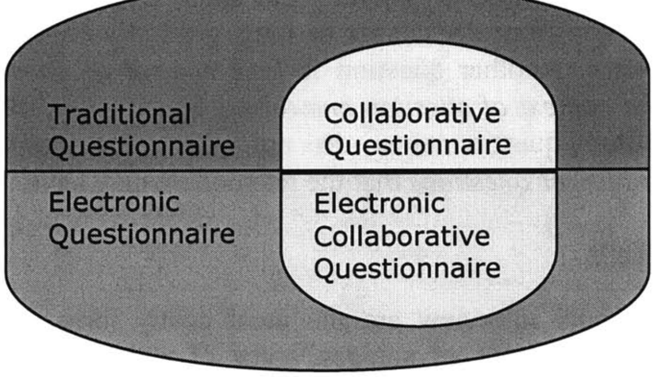

Traditional Questionnaire
Collaborative Questionnaire
Electronic Questionnaire
Electronic Collaborative Questionnaire

Figure 48 - Questionnaire Schema

Figure 48 shows the rough schema on questionnaires. Traditional questionnaire and electronic questionnaire have different modes of data collection. Collaborative questionnaire is the questionnaire specially used in collaborative environments, while electronic collaborative questionnaire is defined to be the electronic questionnaire for collaboration systems.

6.1.1 Traditional Questionnaire and Electronic Questionnaire

Traditionally, data can be gathered with a structured questionnaire in three modes: written questionnaire, personal interviews and telephone interviews [46]. Each mode has specific advantages and disadvantages, which are roughly listed in the following:

<u>**Written Questionnaire**</u>

Advantages *Low cost* is the primary advantage of written questionnaire, whether they are mailed or handed out. A second advantage of written questionnaire is the avoidance of potential *interviewer bias*, which is hard to minimize in telephone and personal interviews. A third advantage is that written questionnaire may place less pressure for *immediate response* on the subject. This factor may be important when the subject has to look in personal records for the information to answer a question. Reponses to attitude questions may also benefit if the subject takes ample time to consider each question carefully rather than giving the response that springs immediately to mind. Besides, written questionnaire are sometime credited with another advantage --- giving respondents a greater *feeling of anonymity* and therefore encouraging open responses to sensitive questions.

Disadvantages The written questionnaire has important disadvantages, particularly in the *quality of data* that can be obtained. At least two considerations are involved. The first is the *response rate*, the chief index of data quality in a survey because it defines the extent of possible bias from nonresponse. A low response rate calls into question any conclusions based on the data. The second aspect besides response rate is the *accuracy and completeness of responses* to questions. A personal interview or telephone interview makes it easier to build rapport between interviewer and respondent, motivating the respondent to give full and accurate answers. On this dimension of data quality as well as response rate, written questionnaires generally fall short. Besides, there are other minor disadvantages. Written questionnaire cannot be long, generally no more than 12 pages or 125 individual responses. Another question is *lack control of question order* and *the inability to control the context of question answering*, specifically, the presence of other people. Finally, written questionnaires do not allow an interviewer to *correct misunderstandings or answer questions* that the respondent may have.

Personal interview

Advantages Personal interview are the most costly form of data collection in general, but they offer important advantages, some of which are shared by telephone interviews. The ability of the interviewer to notice and correct the respondent's misunderstandings, to probe inadequate or vague responses, and to answer questions and allay concerns are important in obtaining complete and meaningful data. The interviewer can control the order in which the respondent receives the questions, which is not possible in written questionnaires. And in general the interviewer can control the context of the interview, including the possible biasing presence of other people. Some advantages are specific to the personal interview, such as visual aids, which can be provided and are useful in a number of contexts. The most important advantage, though, is in data quality. Personal interviews can attain a response rate as high as 80 percent. Moreover, a face-to-face interview can best establish rapport and motivate the respondent to answer fully and accurately. Personal interviews also allow the greatest length in interview schedules.

Disadvantages The primary disadvantage of personal interviews is their high cost, which depends heavily on the geographic overage required by the study. Also, related to the potential rapport between interviewer and respondent is the possibility of large interviewer effects. For example, the interviewer's expectations or personal characteristics (such as race or sex) can influence responses.

Telephone Interview

Advantages Telephone interviews permits a high response rate, on the average just 5 percentage point lower than personal interviews. Telephone interviews do not impose strict limits on interview length, although typically they do not extend much over an hour. All the other advantages of personal interviews, except the ability to use visual aids, are also available in telephone interviews. For instance, the interviewer can notice and correct the respondent's misunderstandings, to probe inadequate or vague responses, and to answer questions and allay concerns. The interviewer can control the order in which the respondent receives the questions, and can control the context of the interview. Telephone interviews also have several advantages over personal interviews, such as substantially low cost, and supervision of interviewers.

Disadvantages The biggest challenge of telephone is that selection of representative interviewees seems to be difficult. For example, telephone directories may yield inadequate samples of phone subscribers. Besides, telephone interviews do have a few more serious **disadvantages**. Interviewer effects are possible although smaller than with personal interviews. No visual aids or cues are possible. And, Too complex questions are impossible to ask.

Electronic Questionnaire

This mode of questionnaire heavily utilizes the Internet and database technology to facilitate questionnaire design, presentation, respondents' invitation, responses storage and result reporting and analysis. It has almost all the advantages of the traditional modes while getting rid of their disadvantages.

Advantages The biggest advantage of electronic questionnaire should be its extremely low cost for the whole process. The questions can be saved and reused, and the results can be dynamically generated. The organizers (or interviews) do not need to own a web server or DBMS to conduct a survey; rather, they can utilize market-available service providers. The participants (or interviewees) only need a standard browser to take any questionnaire. The second advantage is the flexibility to control question order and the ability to control the contexts of question answering, since the software should be able to arrange questions or contexts according to the specific answers given by each respondent. Besides, electronic questionnaires have the following advantages: There is not no potential interview bias. The respondents have a great feeling of anonymity. There are abounded means to provide visual aids, like illustrations, graphs, even audio and video.

Disadvantages Both organizers and participants must have access to Internet to organize and take a questionnaire. The organizer might need to think about good ways to motivate the respondents.

Table 5 summarizes the advantages and disadvantages of these four modes of data collection.

Table 5 - Summary Comparison of Different Data-Collection Methods

Dimension of Comparison	Written Questionnaire	Personal Interview	Telephone Interview	Electronic Questionnaire
Cost	Low	High	Moderate	Lowest
Data quality				
Response rate	Low	High	Moderate to high	??
Respondent motivation	Low	High	High	Low, but should have solutions
Interviewer bias	None	Moderate	Low	None
Sample quality	Low, unless high response rate	High	Moderate to high	Highest
Possible interview length	Short	Very long	Long	Longest
Ability to clarify and	None	High	High	High

probe				
Ability to use visual aids	Some	High	None	High
Speed	Low	Low	High	Highest
Interviewer supervision	-	Low	High	-
Anonymity	High	Low	Low	High
Ability to use computer assistance	None	Possible	High	Whole process
Dependence on respondent's reading	High	None	None	High
Control of context and question order	None	High	High	Moderate

6.1.2 Procedures for Effective Collaborative Questionnaire

The following outline of procedures are recommended for traditional questionnaire [46].:

- Choose modes of data gathering;
- Determine the specific content area to be covered by the questionnaire. Possible factors including the purpose of the questionnaire, related topics and expert/social scientist consulting;
- Determine whether to make use of existing questions or scales;
- Write questions, paying attention to question wording;
- Put together all questions into a questionset, paying attention to question sequences;
- Circulate the draft questionnaire to experts and consultants for comment and suggestions and revise them accordingly;
- Pretest the questionnaire in a small yet representative respondents, so as to identify unforeseen problems on existing questions, to see whether it is necessary to add or eliminate questions; to try to convert open-ended questions to close-ended ones.
- Analyze the pretest results are make necessary changes;
- Begin the actual administration of the questionnaire;
- Analyze the final results.

For electronic collaborative questionnaire, the steps might be simplified to the following:

- Determine the specific content area;
- Determine the class of the questionnaire: polling or survey;
- Design questions, first concentrate on the contents, then focus on the presentation;
- Conduct the questionnaire;

- Collect and analyze responses.

In all these steps, computer-aided tools might be involved to increase the efficiency.

6.1.3 Collaborative Questionnaire

Collaborative questionnaire might be considered as one kind of questionnaire, but it has its own characteristics. The most outstanding one is that for collaborative questionnaire, *sample quality* is not a factor at all, because the group members are automatically considered to be the respondents. As a result of eliminating the issue of sampling, another troubling factor *response rate* also ceases to be a problem. Therefore the focus of collaborative questionnaire is not how to sample respondents and reach respondents, but how to design good questionnaire, not only its contents and context, but also its appearance and style.

Electronic collaborative questionnaire is the electronic version of collaborative questionnaire, which can be divided into two classes: **polling** and **survey**. These two classes of questionnaire are similar in the aspect that the organizer collects and analyzes responses to certain questions from many participants, but they have significant differences.

From the viewpoint of questionnaire scale, polling is light-weighted. In a polling, the organizer just posts simple questions, which are easy to answer. The questions are not only simple themselves, but also have plain organizations. For example, the polling questions generally are not grouped into categories, and no question has sub-questions. No dynamic question order control is needed. On the other hand, survey is somewhat heavy-weighted. It might contain a larger number of questions, which are logically and systematically grouped. Some questions might have sub-questions to make the question organization more clearly. Generally the dynamic question order control or context control is needed, in which the questions are displayed according to the respondent's answers to some judging questions.

From the viewpoint of questionnaire presentation, a polling might just appear in a window/screen. But a survey might be displayed in several windows/screens, each with consistent and impressive presentations.

From the viewpoint of synchronous degree, polling is more synchronous, since the organizer might hope to know the results instantly. Survey is not so synchronous; instead, it might have a survey period, which means the responses collected are effective in specified time duration. Figure 49 is a rough illustration of these two classes of collaborative questionnaire.

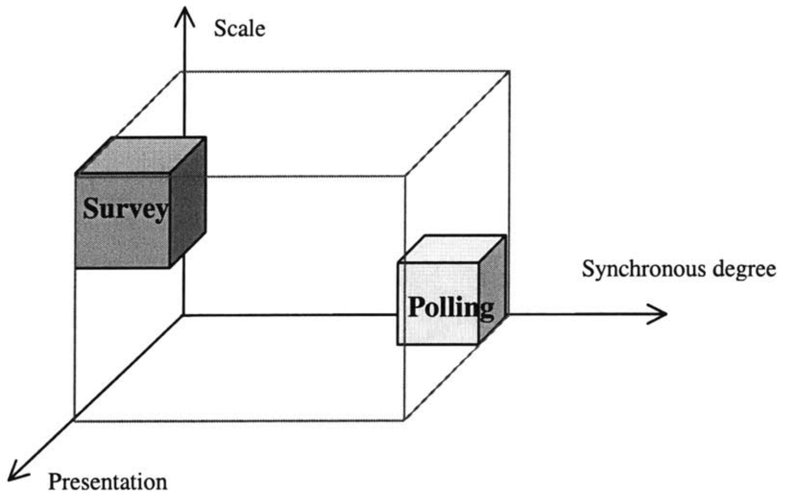


Figure 49 - Rough illustration of two classes of collaborative questionnaire

6.2 Questionnaire Service Provider

To develop a web-based QSP (**Q**uestionnaire **S**ervice **P**rovider), Internet and DBMS are the basic infrastructure. Besides, artificial Intelligence, knowledge-based reasoning are needed to build an agent for monitoring question order and context control. The computer-aided collaborative questionnaire should provide the following facility:

1. An infrastructure for designing questions, inviting participants, answering questions, and generating reports.

2. An agent for aiding organizers to determine questionnaire class.

The QSP developed for this research has provided robust support for three classes of collaborative questionnaire. The difficulty lies in the software agent development, because especially large amount of data/practice should be collected to train such an agent.

6.2.1 Technical Solution

We employ a three-tier Client/Server architecture as illustrated in Figure 50. What each user needs is a standard web browser, like Internet Explorer and Netscape Navigator. The browser will explain and display HTML (HyperText Markup Language) pages and ASP-generated HTML pages. Users are able to fulfill the common tasks involved in questionnaire management, generation, participant invitation, and survey reporting. The business logic is located in server applications running on a server. In this case, the web server, Microsoft IIS (Internet Information Server), is used to enforce the logic. The server application opens connections to the DBMS (Database Management System), which is Microsoft SQL Server 2000. IIS can be running on the same server as SQL Server, or it can connect across the network to SQL Server.

OLE DB, Microsoft's system-level data access interface to both relational data sources and non-relational data sources, exposes a collection of COM (Component Object Model) interfaces to system programming. In this system, OLE DB Provider for SQL Server is used to interact with SQL Server 2000. ADO (ActiveX Data Objects) is Microsoft's new high-level programming interface built on top of OLE DB Providers. ADO is used in ASP codes to open connection, retrieve recordsets and manipulate data.

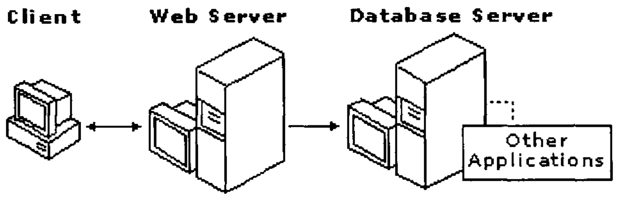


Figure 50 - Three-Tier Client/Server Architecture

6.2.2 Database Design

Figure 51 illustrates the conceptual model. Only important entities are shown here.

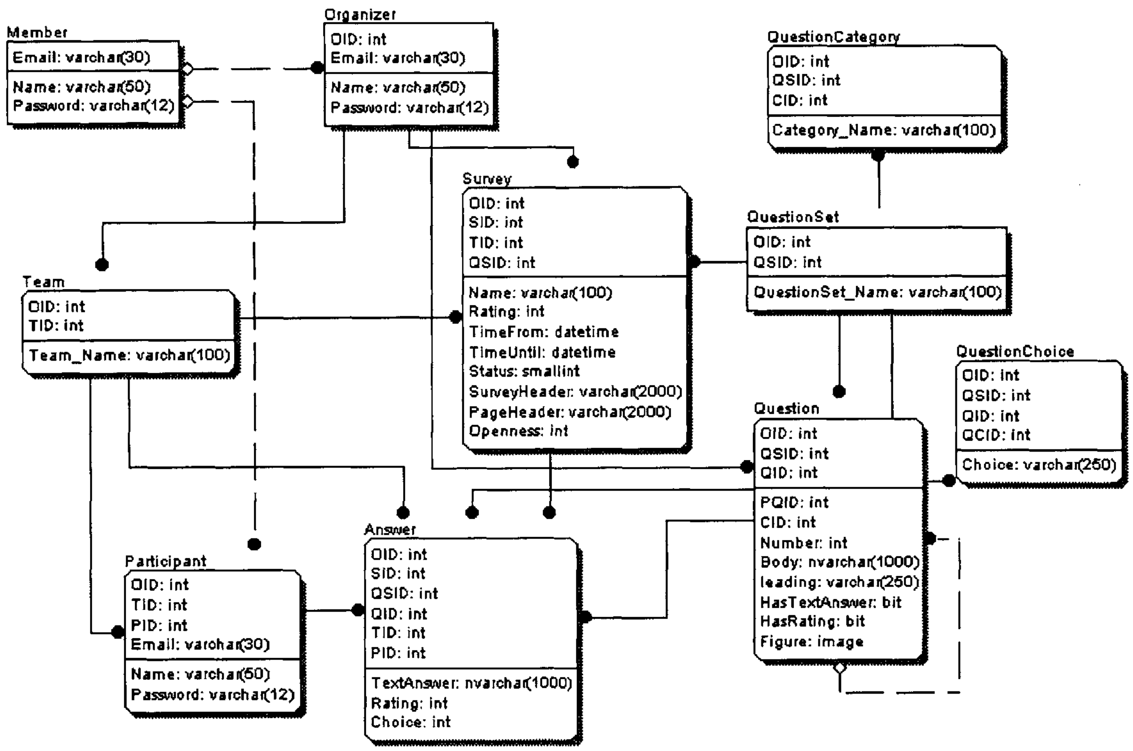


Figure 51 - ER Diagram Of Conceptual Model

Table 6 lists the main relationships between entities. Note we do not include the Organizer, because all the entities (excluding Organizer) have many-to-one relationship with Organizer.

Table 6 - Relationships Between Entities

Pair entities	*Relationship*
Participant & Team	One participant might be affiliated to many groups, while one group contains at lease one participant
Question & (sub) Question	One question might have many sub-questions, while one sub-question should belongs to just one question
Question & Category	One question should belong to one category, while one category might have many corresponding questions
Question & Question Set	One question might appear in many question sets, and one questions set should contain at least one question
Survey & Team	One Survey is taken by exactly one team, while one team might take various surveys (using different question sets)
Survey & Question Set	One survey uses exactly one question set, while one question set might be used in various surveys (taken by different teams)
Answer & Participant	One answer belongs to exactly one participant, while one participant can have many answers (in different surveys and answering different questions)
Answer & Survey	One answer belongs to exactly one survey, while one survey has many answers (answered by different participants, who answers a lot of questions)
Answer & Question	On answer belongs to exactly one question, while one question might have different answers (answered by different participants in different surveys)

We do not give the physical model here, because it is relatively easy to create a physical model on condition that a well-defined conceptual model is at hand. Since relational DBMS does not support many-to-many relationship, we need to replace one many-to-many relationship with two one-to-many relationships, because.

6.2.3 Entity analysis

Besides the main tables (corresponding to entities) displayed in Figure 51, a number of temporary tables are also created in the database. However, to keep the thesis concise, only brief analysis of important entities are given here.

Member

Each user of the system, either an organizer or a participant, must be a registered member. Each member is uniquely identified by his email address, which is used when a user sign up as a new member and later sign in the system.

Each member must have a non-empty password for security.

A member might have two roles. He/she might be an organizer, administering his own surveys. Besides, he/she might be affiliated to a number of teams, taking multiple surveys issued by other organizers.

Organizer

An organizer is a powerful member, who can fully manage all his/her surveys, questionsets, and teams. We only record an organizer's ID and Email in the table Organizer.

An organizer can create a number of teams; for each team, he/she invites people to join via emails automatically generated by the system. The system will check to see whether the invited email belongs to a registered member or not.

- If this email has not been registered, the system will send an email to that address with encrypted string in URL such that the receiver can register and join the team by clicking the URL in the email client program.
- If this email belongs to a registered member, the system will check to see whether this member is already a participant of the target team; if not, the system will send an email to that address with encrypted string in URL so that the receiver can join the team simply by clicking the URL.
- An organizer can create a number of questionsets; for each questionset, he can create questions. Each question belongs to certain category, so that the organizer has to build a list of categories before he can create new question for the questionset.

An organizer can create a number of surveys. Each survey has two basic components: the team and the questionset. See later part for details.

Team

A team is composed of a number of participants. Each team should belong to exactly one organizer, who specifies the email addresses of those people who ought to join this team. The system will take care sending emails and post-processing.

Participant

A participant has not much to do. Each user can sign up as a member, but not able to join any team before he receives an invitation email from certain organizer. After joining, he might also receive email from organizers to invite him to take surveys.

Generally the email message contains a URL, which has an encrypted string (QueryString in HTML). Simply clicking the URL, the browser will take him to proper HTML page. After IIS receives such a request, the system will automatically register a new member, add a member as a participant to certain team, or prepare surveys for the participant to take. Actually a participant needs not to login the system frequently; he just checks his emails!

QuestionSet

A questionset is composed of a number of questions, which are ordered by their numbers. Generally these numbers do not need to be unique, nor do they need to be increased one by one. But the system strongly recommends such as good practice, and does provide a functionality to check the validity of numbers for all questions in one questionset.

Since each question should belong to certain category, a question set also contains several question categories.

There is a constraint to the questions concerning the categories: all questions should be grouped into categories, and all questions belonging to the same category should be ordered continuously. In other words, the first group of questions belongs solely to one category, say, category A, and the second group of questions belongs solely to another category, say, category B, and so on. The questions those are orders later should belong to a category that does not appear for the questions those are orders before; in this case, category A. The system provides a mechanism to check for this rule.

QuestionCategory

Question categories are associated with a questionset, since different questionset has different nature of questions.

Question

Table Question is vital to the whole system. It not only contains the question body, but also carries information on question presentation (how to show the question in the browser) and answer specification (how to limit user to submit valid answer only).

Currently each question might have unlimited number of sub-questions. Note a sub-question itself cannot be a parent-question. Column PQID stores the ID for the parent-question, thus PQID can not be the ID of the question whose PQID is not null.

Column CID, Number, Leading and Body are related to question content and presentation, where CID is used to point to the question category to which the question belongs, and Number is used to arrange questions when displaying them in the browser. Note the system enforce a rule to group the questions in one questionset into categories.

Column Leading and Body store information content of the question. HTML codes are accepted so that attractive questions presentation can be achieved, although some limitations do exist to ensure the correct presentation of questions.

Column Leading is especially useful when the organizer need to display some extra information before the question body. Typically this feature can be used in the question that begins a new category, so that information specific to this category is shown before all questions belonging to the category are presented.

Concerning to the answer specification, current version supports three kinds of answers: comment, rating, and multiple choice. For comment, participants need to type his answers; for rating, participants need to select a number ranging from 1 to maximum rating, say, 5; for multiple choice, participants should choose one from a number of choices.

These three kinds of answer specification can be combined in one question. So it is possible to design such a question that needs the participant to choose a rating number, select a choice, and then write some comments. Another feature worthy of nothing is that both parent-question and sub-question can have any combinations of these three specifications.

Column HasTextAnswer and HasRating are used to specify whether the question accepts comments and rating. For multiple-choice, another table QuestionChoice is used to store all possible choices for one question. See QuestionChoice for details.

Note column HasRating only specifies whether the question needs the user to choose a rating from a range. It does NOT record what is the maximum rating (the minimum is always 1). In the real world, the maximum rating should be the same for all questions in one questionset. In our system, the maximum rating is stored in table Survey as a parameter of the survey.

QuestionChoice

Table QuestionChoice is used to store all choices for multiple-choice questions. Because of the master-detail relationship between table Question and QuestionChoice, one multiple-choice question can present unlimited choices for the participants to choose from.

Survey

A survey has two basic components: questionset and team. Besides, there are several important columns in table Survey.

Column Rating specifies the maximum rating number for all rating questions in the survey. See column HasRating in table Question for related information.

Column SurveyHeader records the information that should be shown before the participants begin to answer questions. Generally the purpose, some notes that the participants need pay attention to, privacy statement, and so on, should be displayed. Column PageHeader records the information shown in each page of the survey, because questions are displayed in multiple pages. Both SurveyHeader and PageHeader accept HTML codes so that they might seem attractive and rich in contents.

Each survey should have its duration. Column TimeFrom and TimeUntil record the beginning date and ending date for a survey, respectively.

Each survey has a status recorded in column Status. Status may be "pending", "current", or "finalized", which is important in that it controls actions that can be taken to the survey. Before the date of TimeFrom, the status of the survey is "pending", and the organizer is free to modify its information, including adding, removing the possible participants, and changing questions. Once the date passes TimeFrom, the status of the survey changes to "current", and emails will be sent to participants to invite them to take this survey. URL with encrypted string is embedded in the email so that all the participant need to do is to click this URL in his email client program. During the effective period of the survey, the system will monitor to find whether there are participants who receive the email but have not taken the survey. Besides, the organizer can generate report on the survey at any time, although it is possible that not all participants have answered questions when he generates such an "incomplete" report. After the date passes TimeUntil, the status of the survey changes to "finalized", and no participant is allowed to take this survey. At that time, the organizer can produce final reports on the survey.

There is a constraint on organizer's ability to change survey information according to its status. When the survey's status is "current" or "finalized", the organizer cannot make changes to the survey, its questionset and its team.

Answer

Table Answer stores all the answers that all participants have given in all surveys. To uniquely identify each answer, the IDs of organizer, survey, question and participant are just enough (can not be less), because the questionset and team info can be deducted from the survey. But for the purpose of easy statistics, the IDs of questionset and team are still recorded.

Table Answer has three columns storing textanswer, rating number and choice chosen. See "Question" for related information.

6.2.4 Functionality analysis

The following storyboard gives the functionality of this system.

Member

Current Member can login the system using his email and password. After successfully login, the system will bring the member into his Information Center. In this Center, a number of hyperlinks is presented to the member for him to proceed. Besides, a hyperlink "Change Profile" will bring the member to change his name and password. This way, both organizers and participants are responsible for their own profiles, and an organizer does not need to manage all his participants, which is a tremendous task sometimes.

New incomers can sign up as a member, and immediately become an organizer if he hopes. Current version does not support the member to explicitly register as a participant, unless he receives an invitation email from certain organizer.

Member is an organizer?

If the member is an organizer, the system will present three main hyperlinks in his Information Center: "Quick Start", "Series Survey", and "Administer My Questionnaire". See later part for detailed information.

Meanwhile, the system will check the following information from the database:

- Is there any survey that should start now? In other words, is there any survey whose column TimeFrom is equal to or larger than the current date? If so, the system will prompt the organizer to start this survey and change its status to "current". Once the survey is started, an invitation email will be sent to all participants in the corresponding team. An encrypted string will be calculated to represent the action the participant will take, and this string will be embedded in the email. Simply stating, if the email is sent to whoever is not a member of the system, the system will register him as a member of the system, make him a participant of the team, and take him to the survey automatically, once the receiver clicks the hyperlink in the email. If the email is sent to whoever are already a member of the system but not a participant of the team, the system will make him a participant of that team and bring him to the survey automatically, once the receiver clicks the hyperlink in the email. If the email is sent to whoever is already a participant of the team, the system will just bring him to the survey.
- Is there any survey whose status is "current"? If so, the system will check whether there are any participants who have not taken the survey, and will prompt the organizer to send another invitation email. This time the system will use the old encrypted string rather than creating a new one.
- Is there any survey that should end now? In other words, is there any survey whose column TimeUntil is larger than the current date? If so, the system will prompt the organizer to end this survey and change its status to "finalized". Emails will be sent to all participants of the corresponding team both as a thank-you letter and a reminding note that they can not change their responses since then.
- When the organizer builds a team, he fills in the email addresses of those people who need to join the team. The system will generate encrypted strings and send emails to those addresses according to their current membership. In the Information Center, the system will check how many email receivers have take actions. For those who have not responded, the system will prompt the organizer whether to send a reminding email or not.

If the member is not an organizer, the system will provide a hyperlink for him to register as an organizer.

Member is a participant?

Each member might be a participant of multiple teams. The system will provide hyperlinks for him to go to different teams. If there are surveys that he can take, the system will show all these surveys in the Information Center. Actually most participants do not need such a step, because he should receive an email which invites him to take specific survey. However, later if he hopes to modify his answers, he can go to the survey from the Information Center.

If the member is not a participant at all, he has no way to explicitly register as a participant of certain team. The only way to achieve this is from the invitation email he receives.

Organizer

Remember in the Information Center, the system provides several main hyperlinks for the organizer: "Quick Start", "Series Survey", and "Administer My Questionnaire".

Quick Start

If an organizer hopes that he can quickly build a questionset, a team, a survey, and begin to collect information as soon as possible, "Quick Start" provides a very good starting point. The System will act as a wizard to guide him to build questions after finishing some necessary yet simple steps.

First, the system will automatically generate a survey name, and ask the organizer whether to change them or not. Then the system will present all available questionsets and teams for the organizer to choose from. If the organizer just chooses an existing questionset and an existing team, then a new survey will be automatically built with the specified name, designated questionset and team, with TimeFrom as the current date, TimeUntil as one year-later, and Status as "current". Immediately invitation emails are automatically sent out to all the participants.

If the organizer chooses to create new team, or new questionset, the system will automatically generate a questions set name, and/or a team name and ask for the organizer whether to change them or not; if not, the system will bring the organizer to create questions, and/or add participants. After he creates all questions and/or manage participants in the team, the system will built all entries in table Survey, QuestionSet, Question, Team, Participant, and send out invitation emails.

Series Survey

In most cases, it is useful to reuse questionsets and teams. For example, if the organizer presents the same questions to the same team in different period, he can easily identify the evolution or progress of what the participants think about certain subjects. Alternatively, if the organizer gives the same questions to different tams in the same period, he then has the information sources to compare the ideas of different teams on the same issue.

"Series Survey" is created to satisfy the first purpose. The same questionset can be used to test the same team in continuous periods. For example, a series of monthly surveys use the same questionset, and are taken by the same participants, but in different months. The organizer just chooses the desired questionset and team, and specifies whether this is a weekly, bi-weekly, or monthly survey. The system will create all entries in the database and take care the email notification issues in different periods.

Administer Questionnaire

Here the organizer can fully manage surveys, question sets and teams that have been created and maintained, including those created in the "Quick Start" step and all the series surveys.

Survey

The system will retrieve all surveys from table Survey and present them. The organizer may add/delete/modify information for each survey, according to the current status of the survey.

For each survey, the system provides functions for previewing and reporting. If the organizer clicks 'Preview', he can get the same presentation of the questionnaire as what the participants will view when they take such survey.

If the organizer clicks 'Report', the system will generate a report with the following format:

Survey Name:
Survey Time:
Number of Participants:
Number of Questions:
...

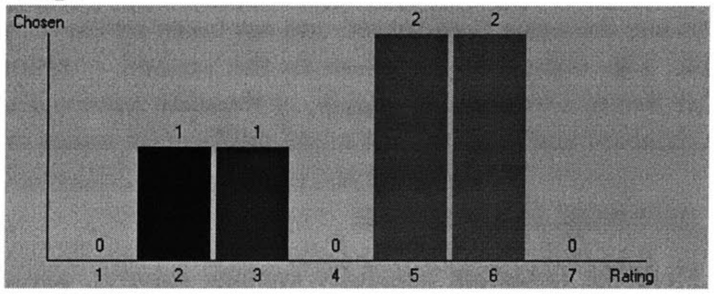
Question	*Participant*	*Response*
Question 1	Participant 1	...
	Participant 2	...
	...	
Question 2	Participant 1	...
	Participant 2	...

For rating questions, statistics information (minimum, maximum, average, mode, standard deviation, and variance) and statistics graph will be displayed (Figure 52). Statistics graph is shown in a Java applet, with parameters passed from HTML codes, which are dynamically generated in ASP.

79 **Working together the team creates solutions that I could not create working alone.**

Listing here are all available ratings.

Answer 1: 5
Answer 2: 6
Answer 3: 3
Answer 4: 5
Answer 5: 6
Answer 6: 2



Statistics Max: 6 Min: 2 Avg: 4 Mode: 6 Std. Dev.: 1.64 Variance: 2.70

Figure 52 - Statistics and Graph for a Rating Question

For series surveys, the organizer can generate statistics on the answers that are given by the same participants to the same rating questions but in different periods. Note we only support rating questions at this version. The simple format is shown as follows:

Team Name:
Participants in the team:
Question Set Name:
Survey1 Name:
Survey1 Period1:
Survey2 Name:
Survey2 Period2:
...

Question	*Number of participants who answers this question*		*Average*	*Minimum*	*Maximum*
Rating Question 1	Survey 1				
	Survey 2				
	Survey 3				
Rating Question 2	Survey 1				
	Survey 2				
	Survey 3				
...					

Also, statistics graph is displayed to show the trend or progress of the participants' ideas on the same subject.

QuestionSet

The system will retrieve all questionsets from table QuestionSet and present them. The organizer may add/delete/modify questions in these questionsets, but the system will check whether there are any current surveys using a questionset; if so, it will be read-only and can not be modified.

A "Check" function is provided for each questionset, to ensure that all questions are grouped into categories. See table "QuestionSet" for details.

Team

The system will retrieve all teams from table Team and present them. The organizer may invite new participants to join a team by typing the emails. But he cannot drop a member if there is at least a current survey or finalized survey is taken by this team.

Participants

For a participant, the system will list all the teams to which he is affiliated in the Information Center, and remind him which surveys are current so that he should take.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

A solid model for a multi-server collaboration system has been established, building on the research done at MIT and Draper Laboratory. Four components for a collaboration system have been identified and their functionality has been established. The chosen solutions will enable the system to be configured in many ways and allow access from any collaborator regardless their network or device limitations.

Five important mechanisms, aimed to improve the performance and scalability of the multi-server collaboration system, to improve the efficiency of collaboration either by handling heterogeneous interfaces or by collecting survey data for collaboration health assessment and enhancement, to make virtually any application ready to be used for multiple participants, are discussed in details. These mechanisms are:

- Mechanism for improving performance and reliability
- Mechanism for seamless handoff of clients among multiple servers
- Mechanism for managing heterogeneous interfaces carried by collaborators
- Mechanism to make applications collaboration-ready
- Mechanism to make participants collaboration-savvy

7.2 Future Works

As newer technologies emerge and different trends emerge, the multi-server collaboration system will be augmented and upgraded to meet the needs. Collaborative efforts are comprised of individuals to solve problems, selecting participants based on the best configuration to solve the problem at hand. The framework is designed to handle these changes and anticipate as much as possible, future developments and needs.

The three mechanisms proposed in chapter 4 are discussed in details, but not fully implemented. Server synchronization algorithms are designed and implemented for both full replication and incremental updating, but only in the database level. Special efforts need to be put to build algorithms for more generic application data, such as Microsoft Word documents, or AutoCAD drawings. Software-based handoff mechanisms are implemented as "semi-transparency", where the clients have to know at least one server to contact before it can join a collaboration session. The automatic notification or alerting of server-availability should be introduced to make servers and services transparent to each client. For the mechanisms to manage heterogeneous interfaces, Controlled-Participant-Identification and Port-of-View are implemented based on VNC adaptation. Future work might be required to make those mechanisms independent of the software that enables server-side computing.

Another area that will be pursued to a greater extent is the introduction of autonomous agents, in the form of sensors or robots, to the collaboration. Sensors are being used in civil structures for stress and fatigue monitoring. They are also used as weather monitors and security systems. Opening sensor networks to a collaboration network where several clients have access to sensor information will empower all participants in a collaboration session. For instance, a participant could get information about the state of a building in a rescue mission prior to entering it.

The inclusion of autonomous robots into the collaboration will provide the participants with information in places that they could not get to as well as increase the mobility requirements of the system. When agents are in the field, network conditions are questionable, depending on location. The collaboration components will need to work locally on such agents to allow point-to-point collaborations to take place between the agents when they are in communication range with one another, but not in range of the system as a whole. Dynamic network configurations will hamper the existing system, as it relies on large servers to handle the collaboration. Portable versions of each component need to be designed and developed. These smaller components will work in a scaled down environment, coming online, as they need to.

With a more complete system in place, system-scale testing plans will be developed and implemented. A sample scenario, presented from Draper, involves several mobile participants connected to the collaboration session through mobile devices such as PocketPCs, Laptops and Java-enabled phones and pagers. These participants are connected to participants located in more permanent locations using multiple desktop computers. The goal of the mission is to direct the field personnel to locate stranded individuals and rescue them in a given amount of time. The fixed position participants act as mission-control, providing field operatives with information such as maps, directives and descriptions of their overall objective. Two-way communication is necessary to ensure the field agents are moving towards their directive. This scenario, when fully functional will include various tests of the system, such as simulated network disruptions, and the use of autonomous agents to investigate certain aspects of the operation. The current prototype has fulfilled the previous requirements for a distributed team without the support for network disruptions and inclusion of fully autonomous agents.

References

[1] http://www.disasterrelief.org/EarthWatch/

[2] Hussein, K.: 'Communication Facilitators for a Distributed Collaborative Engineering Environment' Ph.D. Thesis MIT, 1998.

[3] http://www.cs.unc.edu/~dewan/colab.html

[4] Hania Gajewska , Jay Kistler, Mark S. Manasse, and David D. Redell. Argo: A System for Distributed Collaboration.
http://research.compaq.com/SRC/argo/argopapermm94.html#caleng

[5] The Charles Stark Draper Laboratory, Inc. http://www.draper.com. May 2000

[6] The DaVinci Initiative, Massachusetts Institute of Technology.
http://ganesh.mit.edu

[7] Feniosky Peña-Mora and Justin W. Mills: Component-Based Architecture for Online Collaborative Disaster Relief Mission Planning Environments, accepted for publication in the International Journal of Computer Integrated Design and Construction, August 2001.

[8] Draper Laboratory Inc.: Geospatial Application. December 1999

[9] Draper Laboratory Inc.: Temporal Synchronization Application. December 1999

[10] Fu, L.: 'Adaptation of CAIRO Meeting Environment Towards Military Collaborative Efforts' S.M. Thesis MIT, 1999

[11] Cisco Systems, Inc. 'Deploying H.323 Applications in Cisco Networks'
http://www.cisco.com/warp/public/cc/cisco/mkt/ios/tech/mmcm/tech/h323_wp.htm. May 2000

[12] Anumba, Chimay. ADLIB: Agent-Based Support for the Collaborative Design of Light Industrial Buildings. http://helios.bre.co.uk/adlib. March 1999.

[13] Maher, M. http://www.arch.usyd.edu.au/~mary. May 2000.

[14] Ellis, C: 'CTRG Groupware and Workflow Research'
http://www.cs.colorado.edu/~skip/ctrgOview.html. May 2000.

[15] Center for Innovative Facility Engineering Research Areas. CIFE:
http://www.stanford.edu/group/CIFE/Research/index.html. June 2000.

[16] WebEx, Inc Interactive Platform, April 10, 2000
http://www.webex.com/home/tech_overview.html. May 2000.

[17] Garrett, J., et al: 'Wearable Computers for Bridge Inspection'
http://www.ce.cmu.edu/~sunkpho/mia/overview.htm. May 1998.

[18] Siegel, Jane. Carnegie Mellon University: 'Collaborative Onsite Wearable System'
http://www.cs.cmu.edu/afs/cs.cmu.edu/user/hcii/www/Research/Projects/CollaborativeOnsiteWearabl.html. May 2000.

[19] Michael Stonebraker eds, Distributed Database System, Readings in Database Systems, 3rd edition.

[20] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Grady Booch. Design Patterns. Addison-Wesley Pub Co, 1st edition, 1995

[21] USGS: Earthquake in India on January 26, 2001, Magnitude 7.7.
http://neic.usgs.gov/neis/eqhaz/010126.html

[22] Douglas Herbert. Aid officials assess quake damage.
http://www.cnn.com/2001/WORLD/europe/01/26/quake.relief/index.html, Jan. 2001.

[23] CNN staff. Order collapses as India quake survivors seek food, water.
http://www.cnn.com/2001/WORLD/asiapcf/south/01/30/india.quake.05/, Jan. 2001

[24] Object Management Group: "The Common Object Request Broker: Architecture and Specification" BEA Systems Inc. May 1999.

[25] Sun Microsystems: "The Java Tutorial. Trail: Custom Networking"
http://web2.java.sun.com/docs/books/tutorial/networking/index.html. May 2000.

[26] Sun Microsystems: "The Java Tutorial. Trail: RMI"
http://web2.java.sun.com/docs/books/tutorial/rmi/index.html. May 2000.

[27] Piehler, M.: "Adapting Observer for Event-Driven Design" C++ Report. June 1999.

[28] Platt, D.: "The COM+ Event Service Eases the Pain of Publishing and Subscribing to Data" Microsoft Systems Journal. September 1999.

[29] Sun Microsystems: 'The Jini Architecture Specification' Sun Microsystems. November 1999.

[30] Sun Microsystems: 'Why Jini Technology Now?' Sun Microsystems. January 1999.

[31] IBM: IBM VisualAge Micro Edition 1.3. May 2001

http://www.embedded.oti.com/

[32] Sun Microsystems: JavaTM 2 Platform Micro Edition, Wireless Toolkit http://java.sun.com/products/j2mewtoolkit/

[33] Overview of RAID Levels, http://www.twincom.com/raid.html, October 2000.

[34] Citrix Systems Inc: Server-based Computing White Paper, 1999. ftp://ftp.citrix.com/doclib/SBCWP.PDF

[35] Solari, J. An Application Service Provider Infrastructure for Shared Workspaces in Internet-Based Collaborative Design. S.M. Thesis, MIT, 2000.

[36] Schefström, D. Internet and Distributed Multimedia Course, Center for Distance-Spanning Technology, Luleå University of Technology, Sweden, 1999.

[37] Insignia Solutions Inc. Products white paper. 2001. http://www.insignia.com/products/white.asp

[38] Tarantella, Inc. Tarantella white paper: Tarantella Enterprise 3 Technical Overview, May 2001.
http://www.tarantella.com/whitepapers/pdf/technicaloverview.pdf

[39] ThinSystem, Inc. Datalight white paper: RXE Theory of Operation, 2000. http://www.datalight.com/eval/rxe.pdf

[40] GraphOn Corporation. http://www.graphon.com/Products/BrdgsWndws.html

[41] Tristan Richardson, Quentin Stafford-Fraser, Kenneth R. Wood & Andy Hopper, Virtual Network Computing, IEEE Internet Computing, Vol.2 No.1, Jan/Feb 1998.

[42] International Telecommunication Union (ITU). http://www.itu.int

[43] Internet Engineering Task Force (IETF). http://www.ietf.org

[44] International Multimedia Teleconferencing Consortium, Inc. (IMTC). http://www.imtc.org

[45] Shenton, C. NetMeeting Security Concerns and Deployment Issues, National Aeronautics and Space Administration, white paper, 1998.

[46] Charles M. Judd, Eliot R. Smith, Louise H. Kidder. Research Methods in Social Relations, sixth edition. Harcourt Brace Jovanovich College Publishers, 1991.