

# Adaptation of CAIRO Meeting Environment Toward Military Collaboration Efforts

By

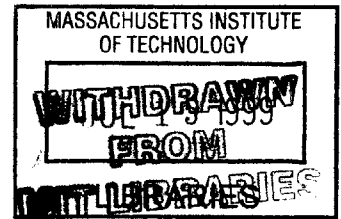
Luke Tan-Hsin Fu

Submitted to the Department of Electrical Engineering and Computer Science  
in Partial Fulfillment of the Requirements for the Degrees of  
Bachelor of Science in Electrical Engineering and Computer Science  
and Master of Engineering in Electrical Engineering and Computer Science  
at the Massachusetts Institute of Technology

May 24, 1999

[Date 1999]

Copyright © 1999 Luke Tan-Hsin Fu. All Rights Reserved.



The author hereby grants to M.I.T. permission to reproduce and distribute publicly paper and electronic copies of this thesis and to grant others the right to do so.

Author \_\_\_\_\_

Department of Electrical Engineering and Computer Science  
May 24, 1999

Approved by \_\_\_\_\_

5/24/99

John J. Turkovich  
Charles Stark Draper Laboratory  
Technical Supervisor

Certified by \_\_\_\_\_

Rimosky Pena-Mora  
Assistant Professor of Civil and Environmental Engineering  
Supervisor

Accepted by \_\_\_\_\_

Arthur C. Smith  
Chairman, Department Committee on Graduate Theses

# Adaptation of CAIRO Meeting Environment Toward Military Collaboration Efforts

By

Luke Tan-Hsin Fu

Submitted to the  
Department of Electrical Engineering and Computer Science

May 24, 1999

In Partial Fulfillment of the Requirements for the Degrees of  
Bachelor of Science in Electrical Engineering and Computer Science  
and Master of Engineering in Electrical Engineering and Computer Science

## Abstract

During the course of tactical operations for military missions, many types of collaboration among individuals take place to assess situations, plan missions, and monitor the execution of operations. The process of coordination not only requires concurrent control in the placement and movement of units in a shared space but also concurrent control in multiple shared views that represent the situation. A prototype system was developed by taking two tactical planning applications and incorporating them into the collaboration model employed by CAIRO (Collaborative Agent Interaction control and synchRONization). The main focus of this research lies in the formalization of an architecture that supports military collaboration scenarios. This architecture lays the groundwork for development of a robust collaboration system that incorporates distributed databases.

Thesis Supervisor: Feniosky Pena-Mora

Title: Assistant Professor of Civil and Environmental Engineering

Technical Supervisor: John J. Turkovich

Title: Group Leader of Information Technology

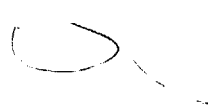
## ACKNOWLEDGMENT

May 24, 1999

This thesis was prepared at The Charles Stark Draper Laboratory, Inc., under IR&D Project 18570.

Publication of this thesis does not constitute approval by Draper or the sponsoring agency of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.

Permission is hereby granted by the Author to the Massachusetts Institute of Technology to reproduce any or all of this thesis.

 (author's signature)

First and foremost, I would like to thank Feniosky Pena-Mora for the assistance and encouragement he has provided me over the past year. I truly appreciate the confidence he had in me, a total stranger to him, in supporting me for this research position. In my one year under his tutelage, I have learned a great amount in the field of collaboration technology.

I am also extremely grateful to all of the individuals at Draper Laboratory who have assisted me during my learning process and made my stay here an enjoyable one: Milt Adams, Peter Scheidler, Mike Hutchins, Charlie Strauss, Tom Wells, and Harold Andrews. Of course, I am especially thankful to my supervisor, John Turkovich, for all of his insights, suggestions, and general compassion. Also, thank you, John, for taking such careful consideration in reviewing my thesis. I have never seen someone read a document so meticulously, and I applaud that.

Finally, I would like to express my gratitude to my family, who have supported me and provided me with guidance throughout my life. I would also like to thank my sister here at MIT, May Tse, for all the Almond Jello she made me during my thesis breaks. Last but not least, I am extremely grateful to all of my friends that have taken part in my trials and tribulations throughout my college years. Special thanks to my friends Robert Lin and Anne Park for getting me through this last term in my times of need. And even more thanks to my roommate Norman Tsao for staying up with me on the last night and to my masseuse Stefani Okasaki for saving my bad back.

Luke T. Fu  
May 1999



# Contents

- 1.0 Introduction ..... 10
  - 1.1 Importance of This Research..... 10
  - 1.2 Objectives of This Research..... 11
  - 1.3 Hypothesis of This Research..... 13
  - 1.4 Benefits of This Research ..... 14
  
- 2.0 Background..... 15
  - 2.1 The CAIRO System ..... 15
    - 2.1.1 The Architecture..... 16
      - 2.1.1.1 Collaboration Manager ..... 16
        - 2.1.1.1.1 Media Drivers ..... 17
        - 2.1.1.1.2 Message Server ..... 18
      - 2.1.1.2 Forum Server ..... 18
        - 2.1.1.2.1 Chairman Meeting ..... 19
        - 2.1.1.2.2 Freestyle Meeting..... 19
        - 2.1.1.2.3 Lecture Meeting..... 20
      - 2.1.1.3 Name Server ..... 20
    - 2.1.2 The Features ..... 20
      - 2.1.2.1 The Interface..... 21
      - 2.1.2.2 The Status Panel ..... 23
      - 2.1.2.3 The Menus..... 23
      - 2.1.2.4 Side-Talk ..... 25
      - 2.1.2.5 The Agenda Tool ..... 27
      - 2.1.2.6 The Agent..... 27
  - 2.2 Military Applications ..... 29
    - 2.2.1 Geospatial View ..... 29
    - 2.2.2 Temporal View..... 30
  - 2.3 Summary..... 32

  
- 3.0 Methodology..... 33
- 3.1 Literature Review..... 33
  - 3.1.1 Meeting Protocols ..... 34
  - 3.1.2 Cross-Platform Compatibility..... 35
  - 3.1.3 Database Integration..... 35
- 3.2 Implementation ..... 36
  - 3.2.1 Familiarization with the CAIRO system..... 36
  - 3.2.2 Development of a Prototype ..... 36
- 3.3 Results ..... 37
  - 3.3.1 Definition of Requirements ..... 37
  - 3.3.2 Formalization of an Architecture..... 37

3.4 Summary .....	38
4.0 Literature Review .....	39
4.1 Meeting Protocols .....	39
4.1.1 University of Illinois at Chicago Research .....	39
4.1.2 DiCE .....	41
4.1.2.1 Establishing a Conference .....	41
4.1.2.2 Controlling the Progress of a Conference .....	43
4.1.2.3 The Architecture .....	43
4.1.2.3.1 Collaboration Management .....	44
4.1.2.3.2 Media Transmission Control .....	45
4.1.2.3.3 User Interface .....	45
4.1.3 Angelo System .....	46
4.1.4 MITRE Research .....	47
4.2 Cross-Platform Compatibility .....	49
4.2.1 Java Collaborative Environment .....	49
4.2.2 SHASTRA .....	51
4.2.3 Problems with Personal Digital Assistants .....	52
4.2.3.1 Resource Limitations .....	52
4.2.3.1.1 Display Limitations .....	53
4.2.3.1.2 Memory and Computational Power Limitations .....	55
4.2.3.2 Network Connectivity .....	55
4.3 Database Integration .....	56
4.3.1 SHASTRA .....	56
4.3.2 Synchronous Collaboration Environment .....	57
4.3.3 M-RAM System .....	58
4.4 Summary .....	59
5.0 Implementation .....	61
5.1 Initial CAIRO Modifications .....	61
5.1.1 Expressions .....	62
5.1.2 Main Room .....	63
5.1.3 Casual Contact .....	64
5.2 Development of the Prototype .....	65
5.2.1 Temporal View Integration .....	66
5.2.2 Geospatial View Integration .....	66
5.3 Summary .....	67
6.0 Results .....	69
6.1 Database Notification vs. Event-Passing .....	69
6.1.1 Definitions .....	70
6.1.2 Comparison of Issues .....	71
6.2 Requirements .....	71
6.2.1 Meeting Protocol .....	71
6.2.1.1 Meeting Styles .....	71
6.2.1.2 Command Hierarchy .....	72
6.2.2 Cross-Platform Compatibility .....	74
6.2.3 Database Integration .....	75

6.3 Proposed Architecture.....	76
6.3.1 The Architecture.....	76
6.3.1.1 Participant.....	77
6.3.1.2 Application.....	77
6.3.1.4 Session Server.....	78
6.3.1.4 Name Server.....	78
6.3.1.5 Domain Databases.....	79
6.3.2 Possible Resource Allocations.....	79
6.3.2.1 Sessions Using Independent Resources.....	81
6.3.2.2 Sessions Using Shared Resources.....	84
6.3.2.3 Hierarchical Sessions.....	86
6.3.2.4 Contingency Meetings.....	88
6.4 Summary.....	91
7.0 Conclusion.....	92
Bibliography.....	94

# List of Figures

Figure 2-1. Current CAIRO Architecture [Hussein 1995] .....	16
Figure 2-2. CAIRO Collaboration Manager Interface.....	17
Figure 2-3. Forum Server Interface.....	19
Figure 2-4. Name Server Interface.....	20
Figure 2-5. Hallway of Meetings .....	21
Figure 2-6. Status Panel.....	22
Figure 2-7. Menus.....	24
Figure 2-8. Side-Talk.....	26
Figure 2-9. The Agenda Tool.....	27
Figure 2-10. The CAIRO Agent .....	28
Figure 2-11. Geospatial View Interface .....	30
Figure 2-12. Temporal View Interface.....	31
Figure 3-1. Three Dimensions of Design Consideration .....	34
Figure 4-1. DiCE Architecture [Vin, et. al. 1993].....	44
Figure 4-2. Example Air Campaign Planning Scenario [Maybury 1997].....	48
Figure 4-3. JCE System Architecture [Abdel-Wahab, et. al. 1997].....	50
Figure 4-4. SHASTRA Application Architecture [Anupam & Bajaj 1993] .....	52
Figure 4-5. Replacing Table with Formatted Text [Lauff & Gellersen 1997] .....	54
Figure 4-6. Brokered Connection and Communication [Bajaj, et. al. 1995] .....	56
Figure 4-7. SCE System Architecture [Park, et. al. 1996].....	57
Figure 5-1. Snapshot of Expressions Tool .....	62
Figure 5-2. The Main Room .....	64
Figure 5-3. Casual Contact .....	65
Figure 6-1. Basic Structure of Command Hierarchy .....	73
Figure 6-2. Proposed Architecture .....	77
Figure 6-3. Initial State of Sessions Using Independent Resources.....	82
Figure 6-4. Information Flow for Sessions Using Independent Resources.....	84
Figure 6-5. Initial State of Sessions Using Shared Resources .....	85
Figure 6-6. Information Flow for Sessions Using Shared Resources .....	86
Figure 6-7. Architectural Representation of a Collaboration Hierarchy .....	87
Figure 6-8. Architectural Representation of Versioning .....	89
Figure 6-9. Information Flow for Versioning.....	90



# List of Tables

Table 5-1. List of Acknowledged Emoticons..... 63

Table 6-1. Comparison of Event-Passing and Database Notification Models on Selected  
Issues..... 70

Table 6-2. Database Dimensions ..... 80

Table 6-3. Affected Databases for Sessions Using Independent Resources ..... 81

Table 6-4. Affected Databases for Sessions Using Shared Resources..... 85

# Chapter 1

## 1.0 Introduction

As technology moves rapidly into the world of networks and distributed environments, the government is looking forward to the wealth of opportunities that are being uncovered with this new form of communication. By combining the computing power of today's processors with real-time data transmission, virtual collaboration forums over distributed networks will become a powerful method of planning and design. Although there are presently legitimate concerns with bandwidth limitations and security issues with this emerging technology, by the time the virtual environment is implemented, these problems will most likely have been resolved [Ethos 1999, ZDNet 1999].

## 1.1 Importance of This Research

Situation assessment, planning, and execution monitoring of military missions are necessary to insure success. None of these functions can be conducted by a single individual. Command staffs must work together as highly disciplined, collaborating teams to coordinate missions successfully. Teams collaborate most effectively when they meet together in the same room [Lipnack & Stamps 1997]. This way, they can actively share information concerning their ideas, experiences, and opinions in various forms of media. Furthermore, individuals evoke qualitative attitudes such as agreement,

uncertainty, questioning, and so forth about the information they see through the use of gesturing and the use of audible, visual, and physical cues.

Networked computers provide a new opportunity for conducting these meetings. The numbers, kinds, and sizes of computer networks are increasing. These networks are being established to send electronic mail, transfer files, and even conduct video teleconferencing sessions. These three activities are already helping to increase communication among individuals. However, much more can be done to enhance the collaboration experience. Some forms of information that team members may exchange during collaboration sessions include text, maps, imagery, and charts. The ability to communicate using all of these forms of data can provide further insight to missions. Computers provide a mechanism through which these types of data can be distributed in an electronic format.

## **1.2 Objectives of This Research**

The Collaborative Agent Interaction control and synchRONization (CAIRO) system is a distributed meeting environment that was developed at the Massachusetts Institute of Technology specifically for civil engineering project development [Benjamin 1998]. This system was created as part of the Da Vinci initiative whose goal is to “explore computer support mechanisms for enhancing distributed engineering design change negotiation” [Pena-Mora, et. al. 1996]. One of the primary goals of the CAIRO endeavor is to add structure to the collaboration process in the virtual space by defining various protocols of interaction that are commonplace in the physical space. The term “physical space” refers to the tangible world where individuals meet with one another face-to-face. On the other hand, “virtual space” represents the artificial environment that can be created and reached through computers. The functions of the virtual environment typically imitate the functions of the real world. In this respect, meeting protocols common to the physical world that control meetings are replaced by computer code that attempt to duplicate these protocols in an electronic environment.

Imagine a scenario in which different collaboration control strategies are employed during different phases of mission planning. In the initial phases, participants may want to adopt a “brainstorming” control structure, which allows them to interact simultaneously without restrictions. However, as a plan solidifies, a chairman control strategy, where individuals may only express their opinion after acknowledgement by the chairman, may be better suited to dealing with changing details. Finally, once a mission plan has been determined, staffs can brief the plan to their commanders about the plan using a lecture style meeting protocol. If these types of meeting rules can be replicated on the computer, computer collaboration efforts can be more effective.

The CAIRO initiative has taken the meeting protocols mentioned above and implemented them in its distributed meeting environment. Its collaboration scheme is based on an event-passing paradigm [Hussein 1997] to coordinate interactions among participants. This original CAIRO system contains no mechanism to utilize data from databases. The new collaboration system is designed incorporate both the passive nature of databases as a means of storing persistent information and also their active nature to notify subscribing users of information changes. This research looks at how to apply the database notification scheme into the CAIRO collaboration architecture. This new notification model, using an underlying distributed database structure, replaces the original CAIRO event-passing model. Many of the features presently using flat files will also be stored in the database to take full advantage of the new database design. Moreover, CAIRO will assume the role of a meeting facilitator using an information policy that defines and controls the flow of information among databases. This information policy not only defines the protocols for the meeting but also the availability of information to participants.

A working prototype is intended to demonstrate the collaboration enabling capabilities of the CAIRO meeting environment within the context of two different tactical planning applications: the Geospatial view and the Temporal view. This prototype will employ the event-passing model, which is central to the original CAIRO system. A sample scenario will be created to simulate the types of situations that may

occur during a mission planning meeting. These sample scenarios are intended to provide insight to the various kinds of interaction that need to be managed during collaborations. From this example, an architecture that incorporates the intended underlying database design is developed.

Ultimately, repeated iterations on this initial prototype and the architecture will be performed to refine the system into the desired product. New control structures and information policies will evolve as necessary to meet the military's demands. Furthermore, the architecture is expected to enable additional application views as they evolve to be assimilated with relative ease.

From the work described above, a robust collaboration system is shaped, allowing for tactical mission planning to occur both among distributed units and within independent isolated units. In other words, participants may choose to work as part of a collaborative group or alone in their own private space. The completed system enables full hierarchical group structuring, agenda building, meeting control structures, and a detailed information policy to facilitate this interaction process.

### **1.3 Hypothesis of This Research**

By developing and using a prototype of a military collaboration system, experience can be obtained for generating a roadmap for next generation systems. Based on the advantages and disadvantages of the prototype, a new set of requirements can be developed that define needed changes. This study produces a blueprint that documents the design path for a next generation system. Future developers can then follow this blueprint in creating the next version of the system.

The architecture for the new collaboration system will retain the important features of the current CAIRO system. In particular, CAIRO possesses many features that relate interaction norms in a meeting to a virtual environment. However, the ability to access databases needs to be incorporated to allow for storage and notification.

## **1.4 Benefits of This Research**

The successful development of this collaboration system will greatly enhance the effectiveness of military missions where members of command staffs and expert advisors cannot all physically meet at the same location. First of all, this new system will enable completely distributed meetings to occur. Military staff can thus undertake situation assessment, planning, and execution monitoring without transporting all the staff members to one central location. Staff members will be able to collaborate with other staff members from locations wherever their physical presence is most valuable. Thus, dealing with unexpected issues from remote locations will become easier. In addition, it will be feasible for associates to participate in multiple meeting sessions simultaneously. This way, staff members and experts can contribute their value to many more organizations than they have in the past. Time, monetary resources, and lack of involvement do not have to be compromised.

As aforementioned, electronic information can be more easily encrypted to prevent third parties from acquiring important mission objectives. Furthermore, the realm of data representation is greatly augmented through the use of computers. Computers have the capability of displaying information in a variety of modes and contexts. All of these benefits lead to more comprehensive collaboration experiences for command staffs.

# Chapter 2

## 2.0 Background

As stated in Chapter 1, the focus of this research is to develop a working prototype of a collaborative military planning system. This system integrates stand-alone military applications that have been developed at the Draper Laboratory with an existing collaboration system, CAIRO, developed at the Massachusetts Institute of Technology. However, to even begin considering an integration of two separate systems, one must first understand the underlying concepts behind each of the tools. A deep understanding of the issues that had to be considered in building the final product is also required. In this chapter, overviews of the CAIRO collaboration system built at MIT and the military applications built at Draper are provided.

### 2.1 The CAIRO System

The CAIRO system was developed in the Civil and Environmental Engineering department at the Massachusetts Institute of Technology as a tool to help engineers collaborate remotely to plan large engineering projects [Benjamin 1998, Hussein 1997]. It has functionality similar to a regular chat daemon; however, it has extra functionality that sets it apart from the typical chat rooms one may find on the Internet.

## 2.1.1 The Architecture

The distributed architecture of the CAIRO system is built upon the concept of client/server technology. Figure 2-1 illustrates an instance of the CAIRO system architecture. In this representation, there are four participants (a, b, c, and d) in the discourse. The design consists of three main elements: the Collaboration Manager, the Forum Server, and the Name Server. The following sections explain the roles of these constituents.

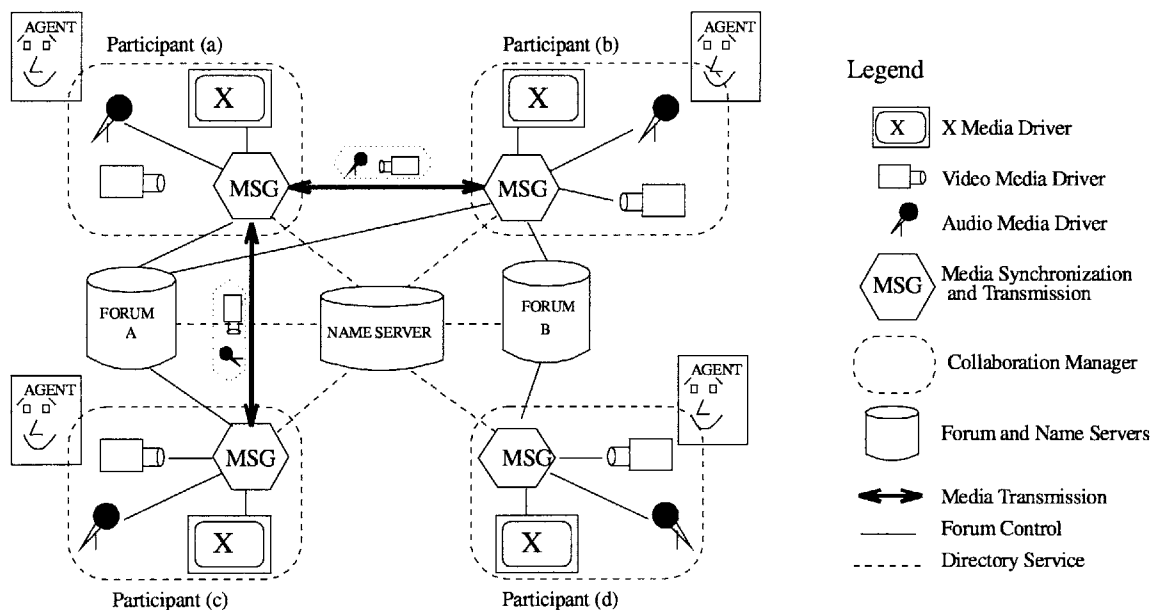


Figure 2-1. Current CAIRO Architecture [Hussein 1995]

### 2.1.1.1 Collaboration Manager

The collaboration manager is comprised of a number of media drivers and a message server. It also possesses a Graphical User Interface (GUI) that allows the user to easily navigate the system (see Figure 2-2). Encapsulated within the interface are numerous other tools, menus, and visual analogies that facilitate collaboration among participants. The interface and its features are discussed in Section 2.1.2.



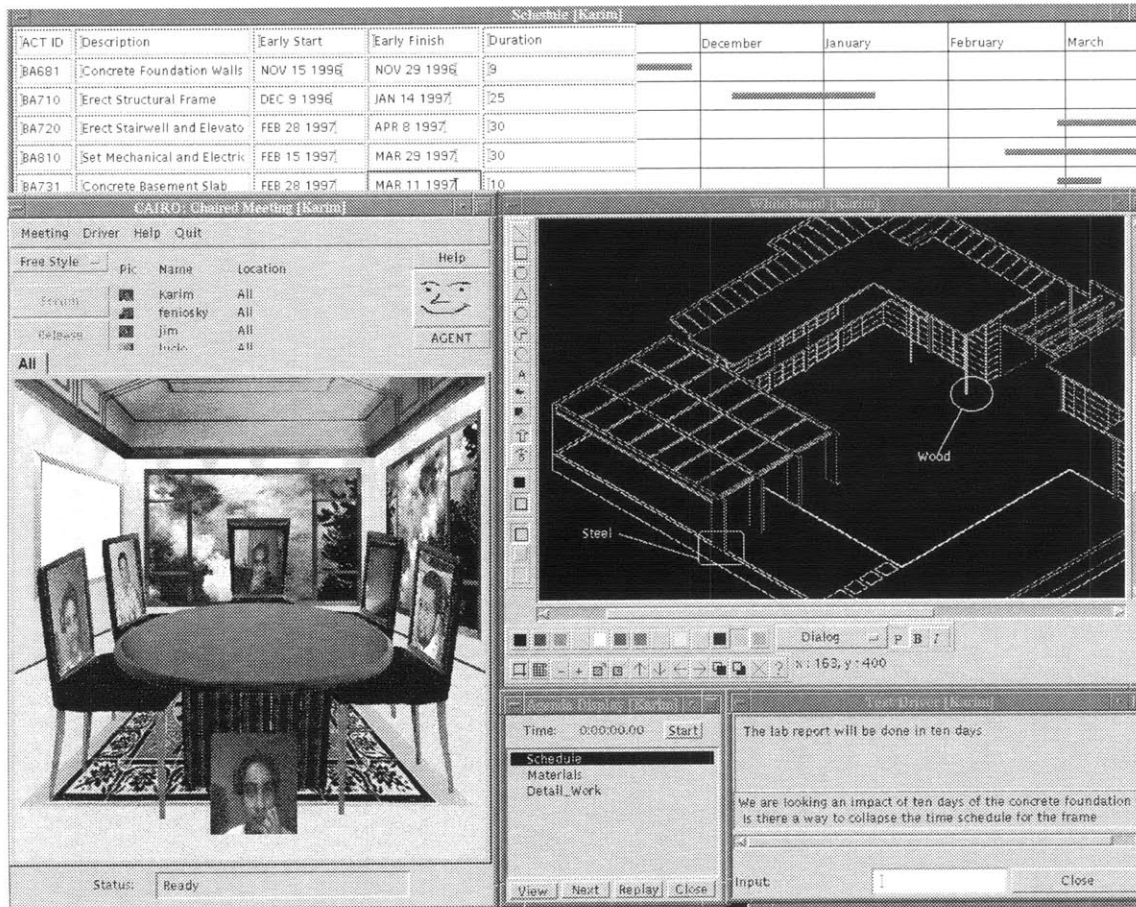


Figure 2-2. CAIRO Collaboration Manager Interface

### 2.1.1.1.1 Media Drivers

The term “media driver” describes a set of applications that users employ to transfer various forms of data to other users. The original version of CAIRO consists of a set of drivers that are specifically geared towards engineering project development. These drivers, which can also be seen in Figure 2-2, include a text driver, a whiteboard, a scheduler, and a design rationale and implementation tool (not pictured). The text driver permits the exchange of text messages among the clients so that they can communicate through the use of written language. The whiteboard provides a shared workspace that simulates an actual physical whiteboard that may be found in an office setting. This driver allows members to communicate information in the form of sketches or drawings. It is also capable of loading images and transmitting them to other participants. The

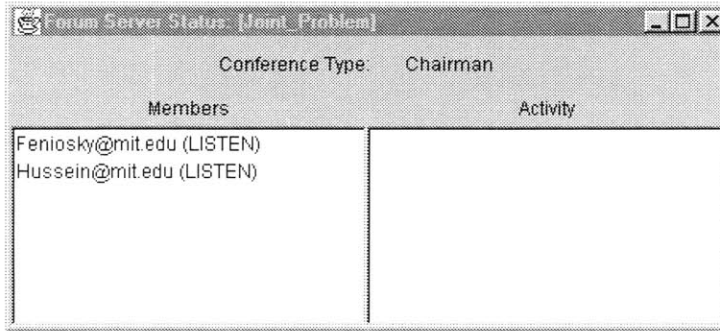
scheduling tool includes an interface that describes a plan in the context of time. It lets users create and modify a schedule in a group setting. Lastly, the design rationale and implementation tool enables participants to collaborate about design issues of an engineering project. The availability of these tools greatly augments the ability to convey specific engineering ideas from one person to another.

#### **2.1.1.1.2 Message Server**

This server handles all transmission of information between users and keeps track of membership in forums. The term “forum” is defined in Section 2.1.1.2. Messages that are exchanged between various participants in a forum are all in the form of TCP/IP datagrams. Each participant has his/her own handler that interprets those incoming messages and routes them to the appropriate drivers.

#### **2.1.1.2 Forum Server**

A forum is defined as “a structured group of participants involved in a collaborative effort” [Hussein 1995]. In this document, the terms “forum”, “meeting”, “engagement”, “conference”, “session”, and “collaboration instance” are used interchangeably, and all refer to this same definition. The forum server forms an analogy to the actual, physical meeting space. It keeps track of all of the individuals involved in a specific meeting and their current state (see Figure 2-3). A member of a forum may be in one of three states: active (logged in and listening), speaking (has control of the floor), or non-active (not logged in). In addition, the Forum Server maintains a log of events that occur within the forum for playback purposes. Thus, a user can go to specific entries on the agenda and replay the events that took place during that agenda item. The playback of the recorded log is only seen by the individual who requested the replay and is not saved in the event log.



**Figure 2-3. Forum Server Interface**

The forum server also has the responsibility of regulating floor control, one of the most important features in CAIRO. Three different meeting styles are presently defined: chairman, freestyle, and lecture. Each of these styles exercises a set of controls to regulate when and to whom users can speak.

#### **2.1.1.2.1 Chairman Meeting**

The chairman meeting style within CAIRO is modeled after the real world chairman meeting. In this meeting strategy, one participant is designated by some prior decision to be the chairman of the engagement. This person is then given authority to determine who gets control of the floor. The chairman himself can take control of the floor as he/she wishes. However, other members of the meeting must request permission from the chair to speak. The chair then decides whether he/she will grant permission to the requestor. These actions are performed using the menus described in Section 2.1.2.3.

#### **2.1.1.2.2 Freestyle Meeting**

The freestyle meeting style has an informal composition. Unlike the chairman style, no one is in charge of the meeting. Instead, members are allowed to take the floor and speak of their own accord. This style of meeting is common for brainstorming sessions in order to keep the conference proceeding smoothly. People can speak freely as ideas flow into their minds, and thus, can contribute ideas without the constraint of obtaining permission from another individual.

### 2.1.1.2.3 Lecture Meeting

The lecture meeting style is, as can be inferred by its name, analogous to a lecture or classroom setting. In this case, the lecturer assumes a role similar to the chairman in the chairman strategy. However, primary communication is directed to the lecturer and no communication takes place between participants in the lecture.

### 2.1.1.3 Name Server

The Name Server centrally manages a global directory for the CAIRO system. This server catalogs a variety of information that can be retrieved by any user. This information includes the name, location, and status of each participant. Likewise, it retains knowledge about the forum's name, location, and status. When a user logs into the CAIRO system, the Name Server can provide information to the user such as the other individuals who are logged into the system and what forums are available to the user. Unbeknownst to the user, information about the Internet address of the other participants and the forums are sent to the client as well. Figure 2-4 exhibits the name server interface where the list of forums and users online are displayed.

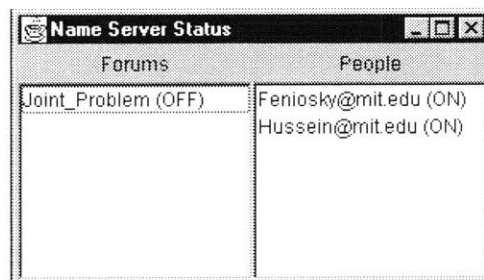


Figure 2-4. Name Server Interface

## 2.1.2 The Features

CAIRO possesses many resourceful features that produce an enriching collaboration experience. These features are what distinguish the CAIRO meeting environment from other collaboration systems, such as video teleconferencing, on the market today. The next few sections describe the features in more detail.

### 2.1.2.1 The Interface

One of the intended advantages of the CAIRO meeting environment compared to other collaboration systems available in the marketplace is that CAIRO provides a number of visual analogies to actual physical meeting spaces. As an example, the image of the table visible in Figure 2-2 can take on two different shapes. The round version cues the viewer that the meeting he/she is participating in is a freestyle meeting. In contrast, a rectangular shaped table denotes a meeting with a chairman. These visual analogies have dual concepts in the physical world that may be familiar to the typical individuals. As another example, the status panel located in the upper half of the main collaboration window in Figure 2-2 contributes information about whether a participant is speaking, being addressed, or being requested. This feature is described more fully in Section 2.1.2.2.

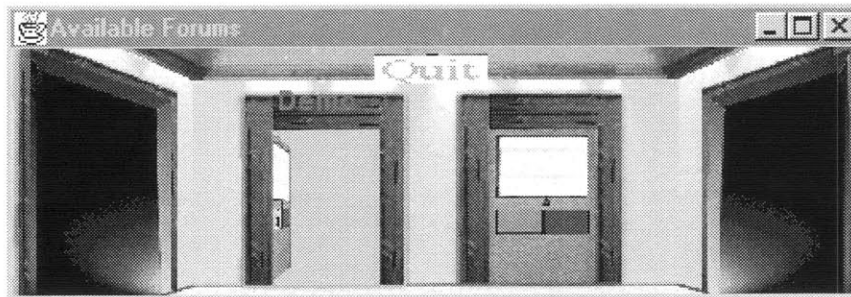


Figure 2-5. Hallway of Meetings

Figure 2-5 demonstrates a third instance of the intended richness of the visual medium. In this window, the analogy of a hallway is used to represent the choice of meetings that are available at a particular instance in time. Each door represents a separate meeting and has a title describing the meeting over it. At the left and right sides of the image lie extensions to the hallway through which users can reach more rooms. Once a user finds the room that he/she would like to join, he/she simply clicks on the corresponding door. It then opens as shown in Figure 2-5. An open door signifies that a room has been entered. Therefore, this doorway represents an entry to our virtual meeting.

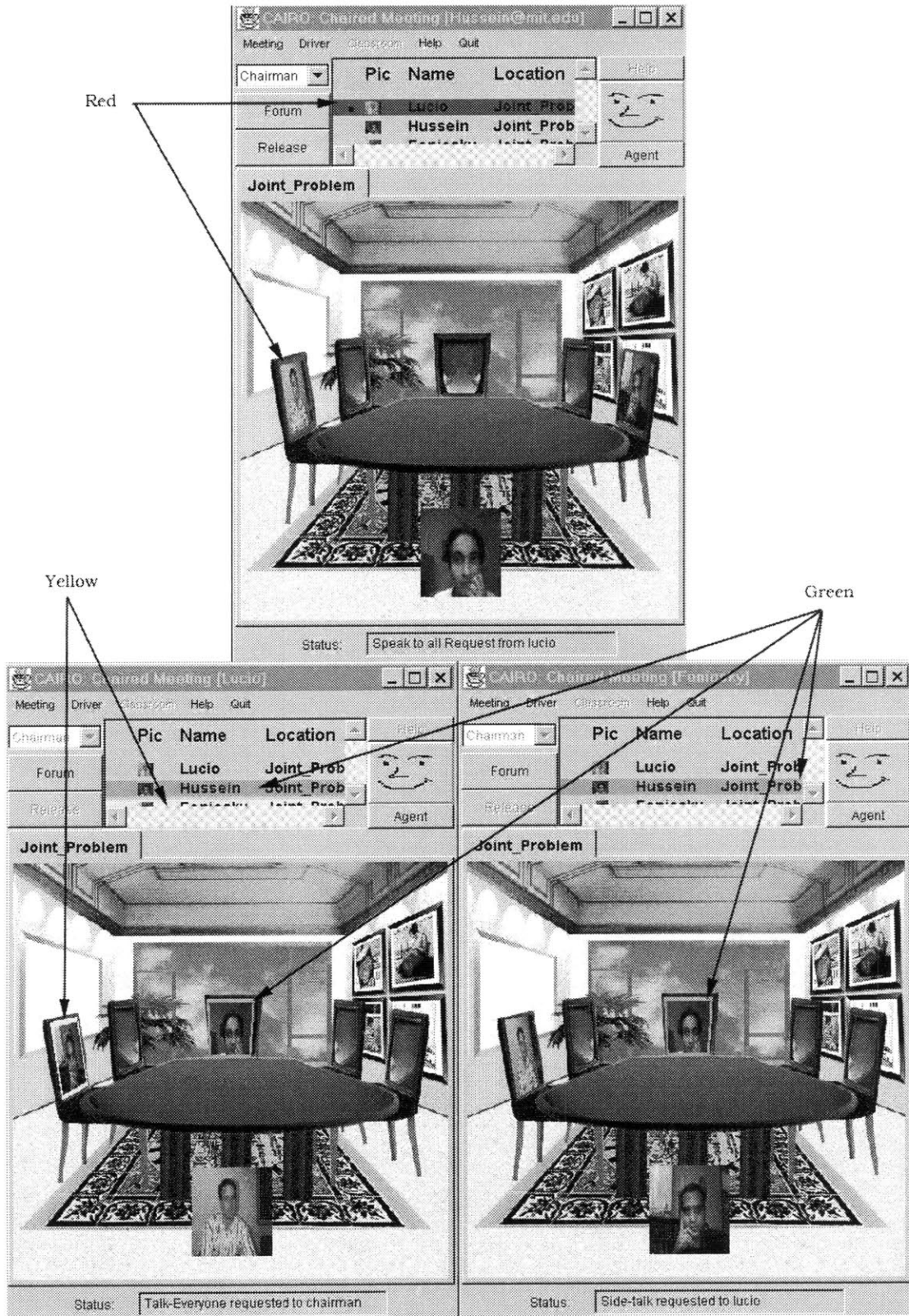


Figure 2-6. Status Panel

For each of the next three figures (Figures 2-6 through 2-8), the users are identified by name as Lucio, Feniosky, and Hussein. These names are used to refer to these users in the rest of this chapter. Hussein's interface resides in the top window while Lucio and Feniosky are in the bottom left and right windows respectively. Their names also appear in the title bar of each window. The rectangular table in this snapshot intends to convey that a chairman style meeting is in progress. Both Feniosky and Lucio's windows show Hussein as the chairman. Hussein's photo also appears at the far end of the table to signify his status within the meeting. Hussein, in contrast, sees no one at the other end of the table.

### **2.1.2.2 The Status Panel**

Figure 2-6 exhibits one example of the abundance of information that can be conveyed to the user through the status panel. At first glance, one may think that the status panel only shows the name and location of a user. However, much more information is concealed in this panel. In this screenshot, the Feniosky client, at the bottom left, has gained permission to speak to the entire group. The state of the meeting can be determined by a quick glance at the status panel. The green highlight on Hussein's name and picture in the windows of Feniosky and Lucio signal that Hussein presently has control of the floor. Similarly, on the Hussein's screen, the dots next to the names of Feniosky (cannot be seen due to size of status panel) and Lucio tell him that he is speaking to the other two participants. The red highlights seen in Hussein's interface around both Lucio's name and image signal to Hussein that Lucio has requested permission to speak. In this scenario, Hussein is the chairman, so he receives all requests to speak. Finally, the yellow highlights that appear around Feniosky's name and image in Lucio's window signal to Lucio that Feniosky is requesting a side-talk with him.

### **2.1.2.3 The Menus**

CAIRO provides a series of pop-up menus for users to moderate the collaboration process. For example, if a user wishes to get permission to speak to the whole group,

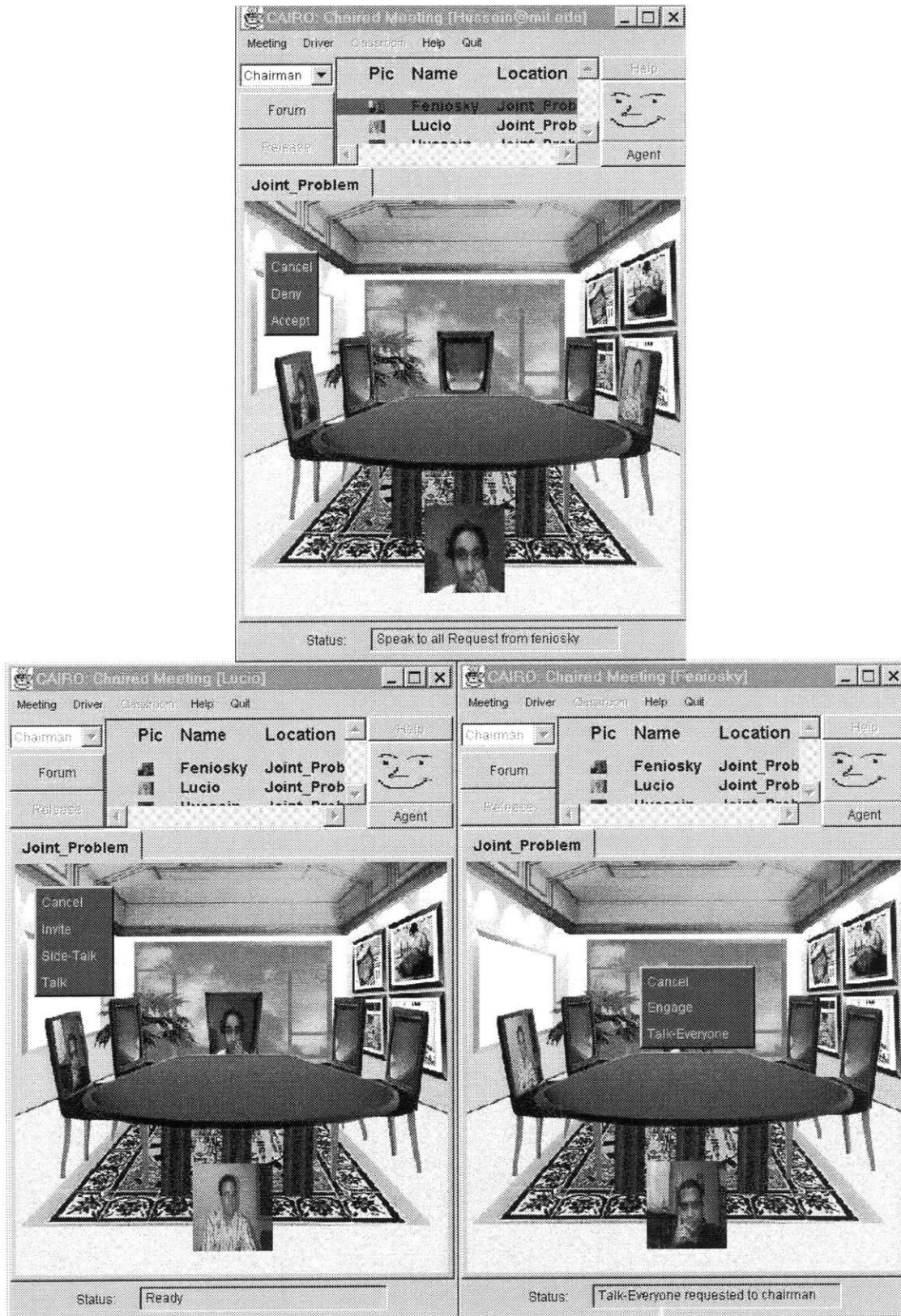


Figure 2-7. Menus



he/she can click on the table, which in turn generates a menu with three options as illustrated in Figure 2-7. In this example, Feniosky has activated the table menu on his window. The table menu gives Feniosky the opportunity to send a request to the chairman of the meeting, Hussein, by clicking on the “Talk-Everyone” option.

Lucio, on the lower left, has exercised the side-talk menu by clicking on another participant’s image. The “Talk” and “Side-Talk” options both send requests to speak to the selected user. However, the “Talk” option, if accepted, only establishes a direct link between the two users within the context of the meeting. On the other hand, the “Side-Talk” menu item causes a new collaboration instance to be initiated. The Side-Talk feature is explained in more detail in Section 2.1.2.4. The “Invite” feature is only enabled when a side-talk instance is active. It allows participants within a Side-Talk to bring other collaborators into the Side-Talk. Finally, a user can respond to a request from another participant by clicking on that participant’s image as Hussein is doing at the top. This click of the mouse generates yet another menu that allows him to either accept or deny the request. These menus attempt to simplify communication by reducing the protocol between participants to mouse clicks.

#### **2.1.2.4 Side-Talk**

In Figure 2-8, a Side-Talk is in progress between Feniosky and Lucio. In the main image panel, notice that their interfaces contain an extra tab labeled “feniosky 1”. This tab, consisting of the side-talk initiator’s name and an ordinal number, represents a Side-Talk. The number is incremented for each side-talk that is initiated, thus allowing for one person to initiate and participate in multiple Side-Talk instances simultaneously. In this case, information disseminated in this folder tab is only transferred between Feniosky and Lucio. Hussein sees none of their conversation and is completely unaware that this side conversation is taking place. This feature allows a subgroup of a conference to speak privately during a meeting. One can conceive a scenario in which a subcommittee is formed to discuss a side issue. In this case, the side-talk function could be used for the purpose of the subcommittee. Lastly, participants of the Side-Talk can switch between

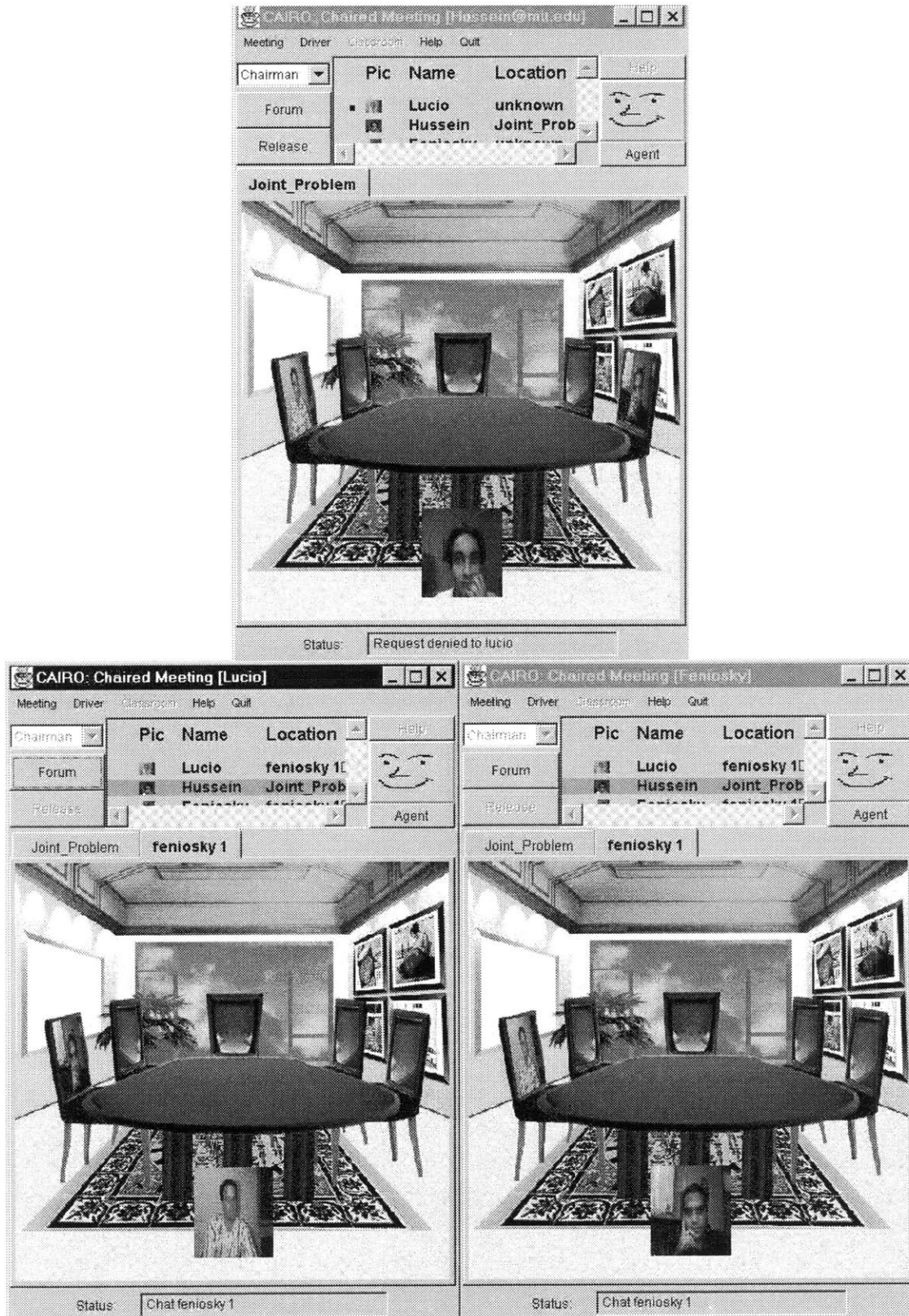


Figure 2-8. Side-Talk

tabs, and by doing so, can alternate their participation between the main meeting and the sidebar conversation.

### 2.1.2.5 The Agenda Tool

The agenda tool identifies the order of a meeting. As in a physical meeting, the agenda contains an outline of topics to be discussed. The agenda in CAIRO works in conjunction with the forum to control the flow of a meeting. The agenda recognizes the duration of each agenda item. CAIRO also provides an agenda wizard tool that can be run to interactively create an agenda for a meeting. This wizard has a graphical user interface that guides users through the steps in generating a schedule for a meeting. The agenda tool, presented in Figure 2-9, is simply comprised of a window that has a timer to keep track of the time spent on an agenda item and a view of the items in the agenda. The panels, from left to right, identify the name of the agenda item, the person in charge, and the style of meeting.

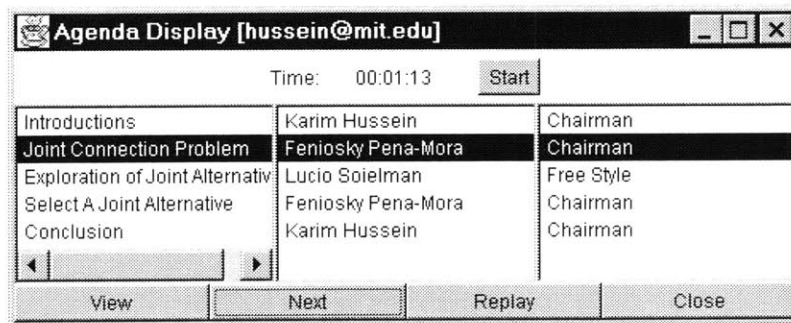


Figure 2-9. The Agenda Tool

### 2.1.2.6 The Agent

The CAIRO agent is one of the integral components that differentiates CAIRO from other collaboration systems. The agent is a program that monitors the actions of the users, residing with the moderator of the meeting. The agent procures information from the interactions among users and tries to make intelligent suggestions to the moderator that may increase efficiency in the meeting proceedings. Currently, the CAIRO agent

possesses only limited functionality, focusing on the agenda, certain textual cues, and request activity.

During a meeting, if the time allotted for an agenda item has expired, the agent will prompt the moderator to continue to the next agenda item. In this case, the agent is a passive timer for the user. The client can choose to accept the suggestion or reject it depending on how he/she feels the meeting is proceeding. However, the agent still serves as a reminder that keeps the participants on topic and in-line with their meeting agenda.

The agent also reacts to certain textual phrases that are sent and received through the text driver. It parses the text that is being delivered for specific key words. For example, if the agent sees a word such as “explain”, it may suggest to the moderator that he/she change the meeting style to a lecture control strategy. If the word “discuss” were used, then the agent would suggest that the freestyle meeting strategy be employed. Again, the user can then decide whether to accept the suggestion or not.

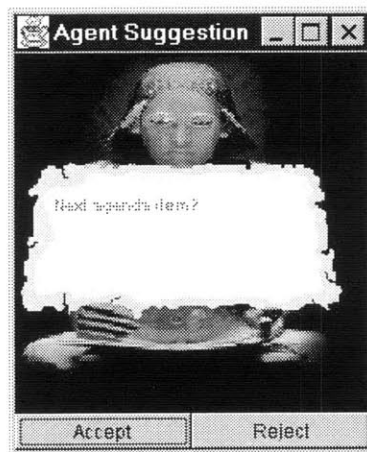


Figure 2-10. The CAIRO Agent

Finally, the agent also responds to actions that are being taken by the users. If the agent notices that a lot of requests to speak are being made by the participants, then it may recommend that a freestyle meeting strategy be applied. This way, the meeting may run more smoothly if users do not have to constantly request permission to speak. Figure 2-10 shows a sample view of the CAIRO agent.

## 2.2 Military Applications

Two main applications that are being developed within the context of tactical mission planning are named the Geospatial View [Strauss 1999] and the Temporal View [Hutchins 1999]. These applications are being developed separately as simple standalone programs that allow situation assessment, planning, and execution monitoring to take place independently on a single workstation. These tools were added as new application drivers within CAIRO during the implementation phase.

### 2.2.1 Geospatial View

The Geospatial View presents the tactical node mission problems from a spatial perspective. Figure 2-11 illustrates this view with annotations on imagery. The tool possesses the following functionality:

- Allow users to load in maps and imagery from a database and bring them into the view port. These images can consist of actual satellite imagery as shown in Figure 2-11. Of course, there exists the ability to zoom in and out of an image and to pan through various points on the map.
- Locations may be pinpointed and marked using designated code names. The shape of these location markers also provides information about the status of the area of interest.
- These locations can be linked by paths determined by the user that depict routes to be taken by units. The application also provides a route editor that permits detailed adjustments to routes through a zoomed in view.
- Times can also be specified for locations and checkpoints along each path. An animation tool can be used to observe how various units move with respect to time. In this way, a complete mission plan depicting the spatial arrangement of

units and their movements can be created. Snapshots of the spatial relationship of all points of interest can be generated for specific moments in time.

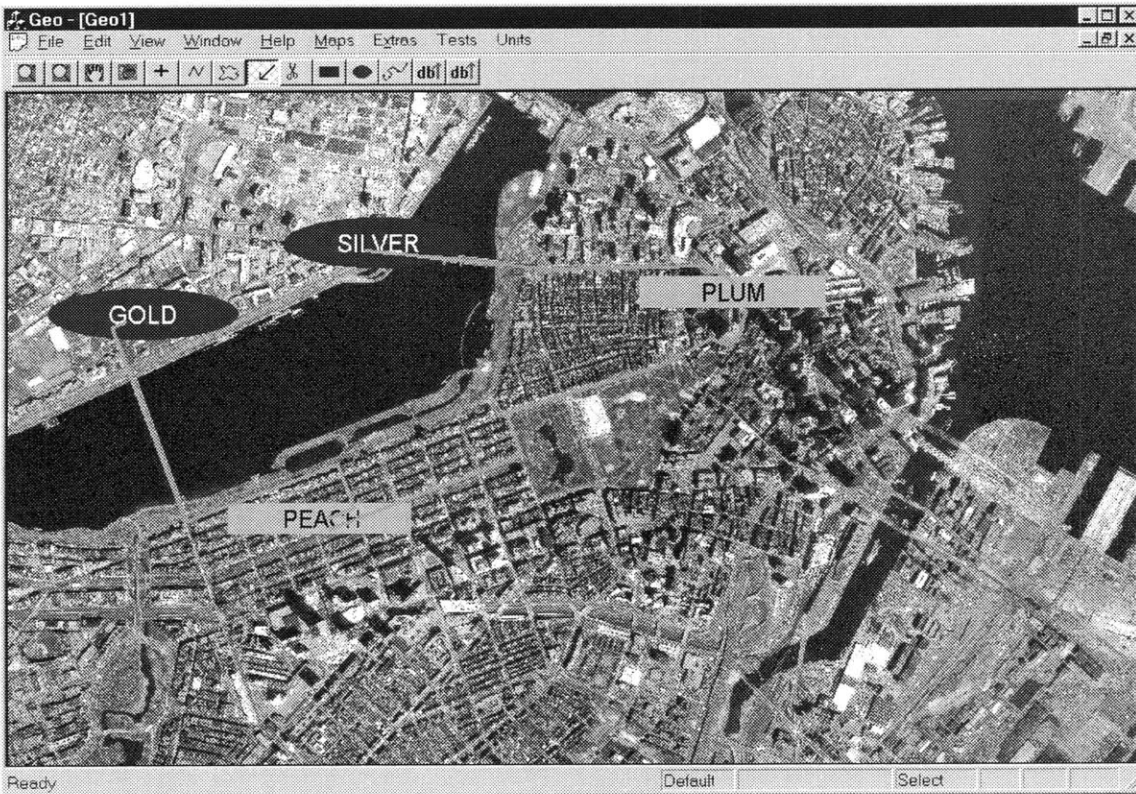


Figure 2-11. Geospatial View Interface

Although this overview does not cover the full functionality of the Geospatial View application or plans for additional features, it does cover its primary functionality in its current state. This summary intends to provide the reader with an impression of the range of utility that accompanies this application.

## 2.2.2 Temporal View

On the other hand, the Temporal View, displayed in Figure 2-12, approaches the planning issue from a fourth dimension, time. This view allows one to create a visual schedule that uses the locations and movements along routes described in the Geospatial View. The functionality attributed to the Temporal View application include:

- A timeline that runs along the top of the application view currently specifies an eleven-hour duration of time. As of now, this timeline remains static but in future revisions, it can be edited and customized for each particular mission.

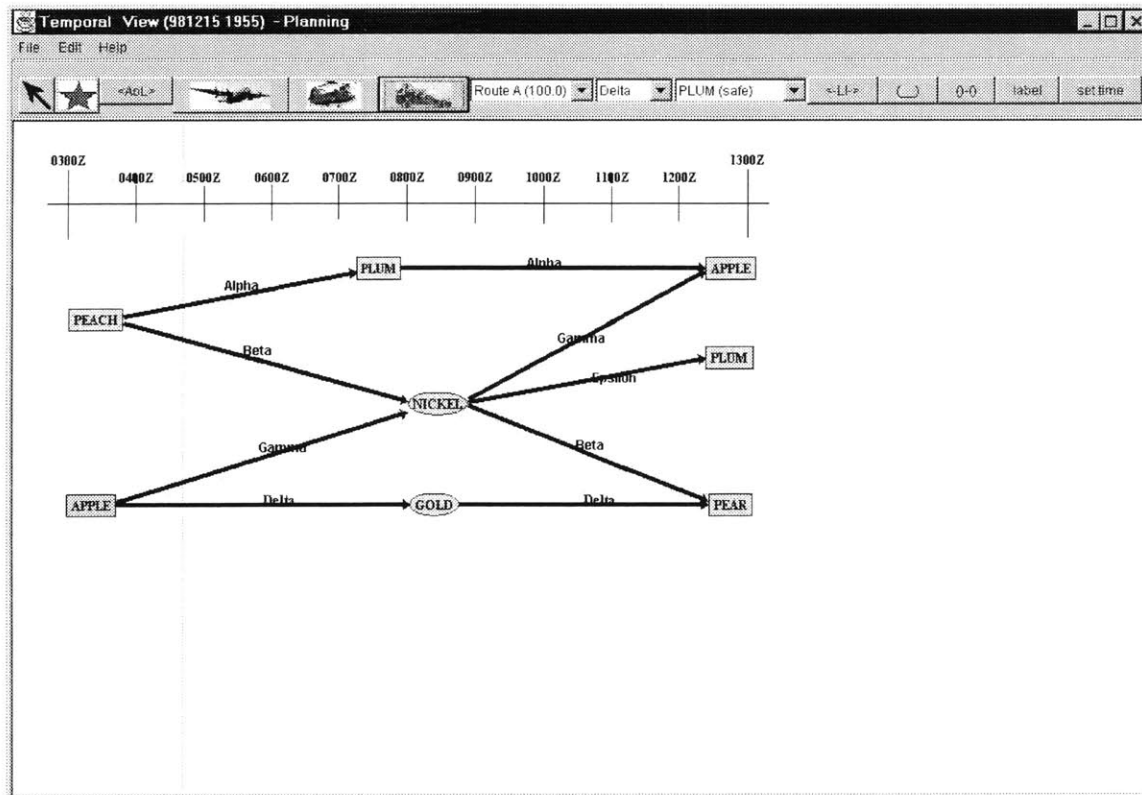


Figure 2-12. Temporal View Interface

- Rectangles and ellipses indicate various locations.
- Directed lines between rectangles and/or ellipses specify movements from one location to another. Currently, the choice of transportation modes is static, but in the future, other forms of transportation should be loadable from the database.
- Annotations along directed lines identify units by code name as well.

The Temporal View application provides a tactical node planning view from the perspective of the time dimension. It allows a glut of information to be passed to users in a visually descriptive schedule format.

## **2.3 Summary**

This chapter presents a brief overview of the three systems that are to be integrated. The important features of each of the systems are explored and sample screenshots are offered to provide the reader with some familiarity with the interfaces. Knowledge of the systems and their capabilities was necessary for any attempt at integration.



# Chapter 3

## 3.0 Methodology

For any substantial research effort, an approach outlining the steps leading to the eventual goal is vital to its success. In this chapter, the main considerations of this study are elucidated. The research process is comprised of a literature review, an integration of various systems, and the subsequent results along with their relationship to the final objective are described.

### 3.1 Literature Review

Three practical considerations dominate the design of a new collaboration system. These concepts are represented in Figure 3-1 as the axes of a three dimensional space. They are:

1. Meeting Protocols
2. Cross-Platform Compatibility
3. Database Integration

To gain a better understanding of how current technologies approach these concepts, a review of other systems developed within the context of these three dimensions was conducted. Each of the systems researched was compared to what

currently exists in the CAIRO initiative. If applicable, revelations of how each complementary system can contribute to the development of a new system were elucidated as well. Those research efforts with applicable architectures are disclosed in Chapter 4 and related to the study in collaborative environments that is being conducted here. The following subsections provide further explication of the three dimensions and how they pertain to the investigation in this research.

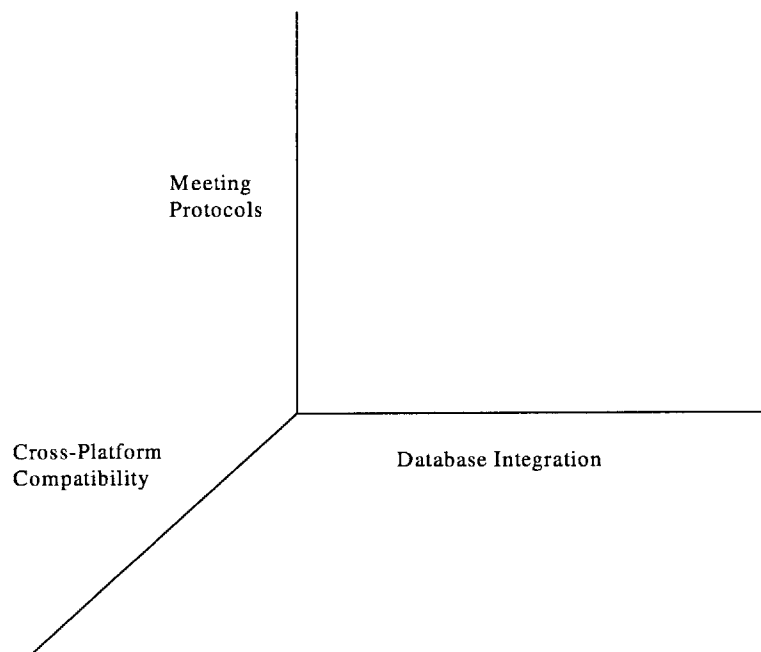


Figure 3-1. Three Dimensions of Design Consideration

### 3.1.1 Meeting Protocols

Meeting protocols are affected and influenced by the hierarchical command and control structures in military organizations. The overhead for conducting meetings might be drastically reduced by having computers handle a lot of the administrative tasks during meetings. For planning meetings, much of the control should probably be delegated to the participants. They shall determine who should get what information and when. However, during the execution monitoring phase where time is of critical essence, rules might be

developed to manage the routes of communication flow. Methods might have to be developed to enable rapid dissemination of information through a chain of command.

### **3.1.2 Cross-Platform Compatibility**

With every attempt to reduce the number of different kinds of computers and operating systems available in the market, there still exists extensive heterogeneity in hardware and software. As some organizations attempt to standardize on Windows NT, others continue to advocate Unix platforms. At the same time, handheld, palmtop, and wearable computers with their own unique operating systems are beginning to permeate society. Due to the nature of military collaboration, field agents may occasionally have to be able to conduct collaboration sessions with base sites. In this case, they may be using a small hand held device that has limited resources such as memory and display ability. To deal with these less equipped systems, a textual interface may be employed to replace the regular graphical interface. A robust solution would be one where a collaboration system is generic enough to run on all existing and future hardware platforms. Thus, these design considerations should be taken into account as well.

### **3.1.3 Database Integration**

The method of distributing data between applications will be changed from the current event-passing model to one that uses an underlying database to facilitate the communication. One of the main incentives for this transition is the unreliability of the event-passing paradigm. During a collaboration instance, for example, if a message from one client to another is somehow lost, the two views will no longer be synchronized. Over time, this error may be propagated to produce contaminated results.

The system depends on the overall reliability of its network. With an event-passing paradigm, lost information cannot be retrieved at a central repository because it simply does not exist. To permit resynchronization after reconnection, a database is introduced. Each participant's view is synchronized with the information in the database. When an individual loses his/her connection to the database, the changes that were made

during the lapse in communication are reflected as soon as the client is reconnected to the system. In addition, the database is a site where data can be stored. The existence of an underlying database enables data to persist from one session to another. Multiple databases may retain duplicate copies of information such that if one database is destroyed, the information lost in one database can be recovered from another. However, protocol for database interaction and synchronization needs to be defined.

## **3.2 Implementation**

The integration of the Draper applications into the framework of CAIRO consisted of two phases. The goal of the first phase was to gain familiarity with the CAIRO architecture by integrating minor features from another collaboration system into the CAIRO system. The second step was to take the knowledge gained from the previous phase and apply it to the actual integration of the military tools with the CAIRO meeting environment. Both of these procedures are detailed in Chapter 5 while a brief summary is presented in the next two sections.

### **3.2.1 Familiarization with the CAIRO system**

Before development could be initiated on either of the Draper applications, knowledge of the architectures and underlying code must be gained. To do so, some initial modifications were performed on the CAIRO system. These modifications allowed for a better understanding of the program and its implementation so that integration of the military applications could be conducted with detailed comprehension of the underlying event-passing model. To achieve this level of understanding, several new drivers were added.

### **3.2.2 Development of a Prototype**

After completing the modifications described in the previous section, a prototype system that allows distributed collaboration between clients for mission planning

purposes was implemented. The Temporal and Geospatial View applications were integrated within the context of the CAIRO system, taking advantage of the predefined event-passing paradigm provided by CAIRO. This prototype served as a first iteration upon the distributed collaborative system toward the target system. It provides a solid framework upon which new modifications and versions of the program may be built. Furthermore, this prototype can then be used to study the types of interactions that may occur and provide insight on how to deal with them.

### **3.3 Results**

After the hands-on implementation phase, the research assumed a more introspective nature. Based on the conclusions from the literature review and the prototype, a set of requirements that are to be met by a new collaboration system was determined. Then, the architecture of a new database collaborative system was designed conforming to the aforementioned requirements. These results are explored in Chapter 6.

#### **3.3.1 Definition of Requirements**

Upon completion of the prototype, an evaluation of its functionality was conducted. This evaluation led to a set of requirements upon which an architecture was designed. In addition, a study of the advantages and disadvantages of the event-passing model compared to a database notification model is found in Section 6.1. This assessment elucidates the need to change toward a collaboration design using a database framework. In particular, the focus of the requirements revolves around the three dimensions outlined earlier in this chapter.

#### **3.3.2 Formalization of an Architecture**

Based upon the requirements outlined from the results of Section 3.5, a preliminary architecture for the new military collaboration system was designed. This architecture is built upon a distributed database structure. Several different collaboration

scenarios within a military context are discussed in relation to the new architecture. These scenarios provide a comprehensive review of how the system responds to certain circumstances and aims to confirm the robustness of the design.

### **3.4 Summary**

The eventual architectural design, employing a database notification paradigm, was generated by following the process described in this chapter. The methodology practiced here is comprised of three main stages. First, a review of previous studies in the collaboration arena was conducted. Then, a rudimentary prototype was built using the collaboration framework of the CAIRO system with the Draper military applications. Finally, a database architecture was designed that supports distributed collaboration efforts for situation assessment, mission planning, and execution monitoring.

# Chapter 4

## 4.0 Literature Review

In this section, several other complementary collaboration systems are examined which tackle one or more of the following issues: meeting protocols, platform compatibility, and database integration. Following each synopsis is a brief analysis of the system compared with CAIRO and the objectives of this research project.

### 4.1 Meeting Protocols

In the following subsections, research from the University of Illinois at Chicago, IBM T.J. Watson Research Center, MIT, and MITRE are presented. These systems describe protocol concepts that span meeting control structures, software agents, and command hierarchies.

#### 4.1.1 University of Illinois at Chicago Research

Research at the University of Illinois at Chicago's Department of Electrical Engineering and Computer Science looks at the concept of collaboration using the World Wide Web [Chang, et. al. 1997]. The results of their study centered on four central ideas. The next few paragraphs expound upon their ideas and how they pertain to the research conducted in this document.

Their first finding is that a collaboration system should be Web-based. Although CAIRO does not operate specifically as a Web-based application in the conventional sense (as an applet within a web browser), the underlying communication technology in fact takes full advantage of Internet capabilities through the use of the TCP/IP network protocol to deliver and receive messages within the system. Security measures built into web browsers such as Netscape Navigator place restrictions upon message transmission, so the effort to create an applet in CAIRO had to be abandoned in lieu of an independent application.

Secondly, the Chicago research group contends that such a system must be supported in a multimedia environment. Currently, CAIRO allows for text and visual communication between individuals. Full audio and video streaming capabilities will eventually be added to CAIRO to augment the collaboration process. This functionality can create a more natural environment for communication to occur in a vastly electronic setting.

In this study, it was also determined that a virtual conferencing environment would have to possess a set of "floor control rules" similar to Robert's Rules of Order [Robert 1996]. Robert's Rules of Order are a set of guidelines that describe the general rules of parliamentary procedure. They describe a methodology of how members of a deliberative assembly can speak and vote. In CAIRO, there have already been these different rules of order defined with the capability of adding more as needed. These meeting styles, as discussed earlier, control who is allowed to speak when and to whom. In addition, the paper [Chang, et. al. 1997] points out that the rules should be modified to account for asynchronous collaboration efforts. This area has not been fully explored in the CAIRO initiative and could be a great source of improvement in the technology.

Lastly, the authors [Chang, et. al. 1997] state that a formal language for defining meetings needs to be studied. They present the concept of a Meeting Declaration Language (MDL). This original language allows users to describe the rules and operations that are to be followed in a meeting. Much like a programming language, a compiler is then used to process a user's input and create the protocols and structure for



the meeting. CAIRO approaches this problem from a user's perspective. Instead of developing a new language for defining meetings, CAIRO simply employs an easy to follow interface that steps the user through various setup options. This simple program then applies the user's choices to initiate a forum and an agenda for that meeting.

#### **4.1.2 DiCE**

At IBM T.J. Watson Research Center, a study is being conducted on methods of collaboration management [Vin, et. al. 1993]. Their research serves as a foundation for the Distributed Collaborative Environment (DiCE) being developed at their facility. Their research approach comes from a theoretical perspective. Here, the main results of their research with some description as to how CAIRO resolves each of the issues are presented. In addition, the architecture of the DiCE system is depicted with an analysis of its advantages and disadvantages.

##### **4.1.2.1 Establishing a Conference**

The essence of the paper lies in its explication of what they call "Conference Management" [Vin, et. al. 1993]. They believe that the process of establishing a conference warrants some attention. In fact, they believe it consists of three important phases:

1. Invitation and arbitration
2. Negotiation
3. Setup

Invitation and arbitration deals with the tasks that occur prior to meetings such as sending invitations for a meeting, electing one person to initiate a meeting, collecting responses of participation, and the establishment of connectivity. Presently, the CAIRO system has not effectively tackled many of these problems relating to pre-meeting

situations. A skeleton framework does exist for sending out e-mails that invite participants to a meeting. CAIRO also operates under the assumption that an arbitrator, who establishes the meeting, will be selected through prior communication using other means of notification. This feature does not exist in CAIRO. The arbitrator would then collect responses through e-mails as well. The CAIRO environment consists of an interface for creating and entering meetings, but establishment of connectivity between clients is done automatically by CAIRO's collaboration framework.

The negotiation phase refers to the stage when the operational parameters of the conference are defined. They bring forth two main problems to consider:

- Management of heterogeneous hardware and network environments - For CAIRO, network heterogeneity is not a major concern since the networking aspects of the system are built around the TCP/IP convention. As long as a system adheres to the networking protocol of TCP/IP as most computers these days do, networking will not be a problem for users of the CAIRO system. The problem with hardware compatibility is definitely an issue, though. Presently, the only focus on hardware issues has been portability across different operating systems. More research will have to be done into the actual hardware capabilities of a system. This concept is further examined in Section 4.2.
- Flexibility of participation in conferences - This phrase describes how a collaboration system should allow participants to decide upon their own mode of participation. Once again, CAIRO does not venture into this realm of operation. For users of the CAIRO system, the arbitrator who possesses the power to define the meeting and its requirements does most of the negotiation. However, some research in this area may be valuable.

The final stage, which the authors [Vin, et. al. 1993] present, is that of setup. The setup phase creates a system of long-term connectivity for the clients. CAIRO encapsulates much of the setup and connection of the numerous clients. Once a meeting

is entered in the CAIRO environment, the proper connections and links are created for the duration of the collaboration instance.

#### **4.1.2.2 Controlling the Progress of a Conference**

The authors touch upon the concept that conference interactions over a network may have to be coordinated in some manner. Their view is one of necessity rather than one of efficiency. They point out that applications such as shared whiteboards or concurrent editors may require some sort of access controls in case two or more users attempt to change the same data simultaneously. In their paper, they propose a simple baton-passing schema. This strategy centers upon a predefined number of batons, which represent access privileges to a particular application. A participant gains control of a baton by procuring it from the previous owner. Of course, this action requires that the owner permit the requesting participant to take control of the baton. CAIRO, on the other hand, focuses its attention on the concept of coordination within a conference. CAIRO has three defined control strategies that are expounded upon in Section 2.1.2, which are based upon the concept of tokens. Although the current version of CAIRO does not support the baton-passing method of collaboration, this strategy can easily be incorporated into the system using the token approach.

#### **4.1.2.3 The Architecture**

The software architecture for the DiCE system has been reproduced in Figure 4-1. The acronyms found in the figure are revealed in the following sections. After some preliminary analysis, one can see that the architecture parallels the CAIRO collaboration manager design in many ways. The system is composed of three main components: a collaboration management unit (CMU), media transmission control unit (MTU), and interfaces.

### 4.1.2.3.1 Collaboration Management

The collaboration management functionality can be further broken down into three more sections:

- Multimedia server (MMS): The MMS provides a naming and addressing schema for agents so that invitations can be conducted by name. It is also responsible for creating a Collaboration Management Unit (CMU)

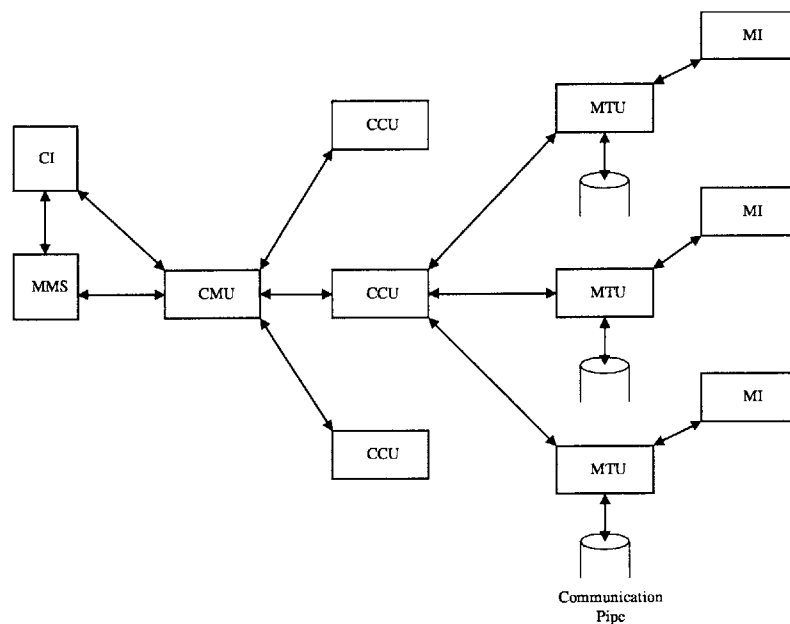


Figure 4-1. DiCE Architecture [Vin, et. al. 1993]

- Collaboration management unit (CMU): This module represents the user, performing the functions of a “schedule keeper”. The CMU is in charge of generating a Conference Control Unit (CCU) for each conference that is started. The CMU and MMS together incorporate the functionality of the Name Server in the CAIRO architecture.

- Conference control unit (CCU): The CCU exists as an analogy to an actual conference and encapsulates the meeting structure and controls. It takes care of defining the meeting within the context of the conference and setting access privileges for the participants. Multiple instances of this module may be present since users may participate in multiple conferences simultaneously. This element mimics the role of the Forum Server within CAIRO.

#### **4.1.2.3.2 Media Transmission Control**

The Media Transmission Units (MTU) regulates the exchange of information between clients. Again, notice a similarity between this component and the Message Server in CAIRO. Although their implementations most likely differ, both elements serve the same function: directing the transmission of information from one client to another. Moreover, in the DiCE architecture, a MTU exists for each application driver whereas in CAIRO, only one Message Server exists for each client that is in charge of all communications for that client.

#### **4.1.2.3.3 User Interface**

The user interface is composed of four different interfaces at the moment. They include:

- Collaboration Interface (CI): The Collaboration Interface acts as the go-between for the user and the underlying collaboration system. In reality, the Collaboration Interface is similar in functionality to the Collaboration Manager in CAIRO. It allows the user to place, join, and terminate conferences as well as adding and deleting applications within the conference.
- Video Interface (VI): The Video Interface is comprised of two parts: 1. Video Window, and 2. Video Control Panel. The role of the Video Window is simply to display the motion video at 30 frames/sec. It provides information about the participants in a meeting and provides a series of menu options to alter the source

of the video. The Video Control Panel allows the client to change display-specific features.

- **Audio Interface (AI):** The Audio Interface provides a window, which allows the participant to control audio-specific parameters such as volume or mixing of audio streams.
- **Shared Workspace Interface (SWI):** This module consists of a Shared Workspace Control Panel and an Application Window. The Control Panel constitutes an interface that can call up other applications. The Application Window, therefore, exists as the interface between the user and the application itself. Each of these interfaces behaves in much the same way as the CAIRO interface and its drivers.

These final three interfaces are represented in Figure 4-1 by the more general acronym for a Media Interface (MI) implying that each of the components listed could be any one of these three applications.

### **4.1.3 Angelo System**

The ANGELO system [Ali-Ahmad 1997], another application, which evolved from the Da Vinci initiative [Pena-Mora, et. al. 1996], is not in itself a collaboration tool. However, some of its features may have relevant application to achieve the goals of this research. ANGELO observes and records user actions to generate a model of the user's interests. Based on this model, the system then locates other related objects that the user may be interested in. Thus, some sort of artificial intelligence is gained by what the computer "learns" about the user and reacts accordingly. The learning capacity is implemented using a clustering algorithm [Ali-Ahmad 1997].

Presently, CAIRO does possess limited artificial intelligence capabilities. Its agents keep track of meeting time, and provide suggestions about meeting style changes based on user interaction norms. In addition, the collaboration system may be further

improved by allowing it to learn about its user and determine what sort of relationship he/she has with other users. Using this information, the CAIRO meeting environment can then decide what sort of meeting protocol to follow and what information policy to apply.

#### **4.1.4 MITRE Research**

The MITRE Corporation has itself performed preliminary research into collaborative planning in a military context [Maybury 1997]. Their study however is focused on the realm of air force planning operations. Maybury elucidates upon the importance of electronic collaboration research, saying that “while we are increasingly able to work in concert with our allies at a political level to control the proliferation of weapons of mass destruction and to promote stability and democracy, in battle we remain limited in our ability to share information and collaborate using an electronic information infrastructure” [Maybury 1997 p. 2]. The work being conducted at MITRE consists of a number of different planning applications being developed in parallel. Some of these systems have been connected to databases through self-developed IDL interfaces as a front-end to CORBA. They found that “wrapping existing legacy systems by defining and implementing CORBA interfaces provides a powerful method for systems migration” [Maybury 1997 p.6]. From this study in air campaign planning, a distinct hierarchical collaboration process has been identified. This result reinforces what is believed to take place in the military as a whole. The hierarchy of the military ranking system lends itself to a decision process that must matriculate through a command hierarchy.

Figure 4-2 depicts a snapshot of the activities that might take place in a typical planning process. One can see that the planning procedures move through different levels of the command hierarchy from the command headquarters down to individual combatants going through various problem redefinitions at each stage of the collaborative process. In addition, other research in air campaign planning has produced Air Campaign Planning Tool known as the Theater Analysis, Replanning, and Graphical Execution toolbox (TARGET) and the Force Module Analysis and Management Tool (ForMAT). These tools are geared towards deployment of different units during a mission. Another

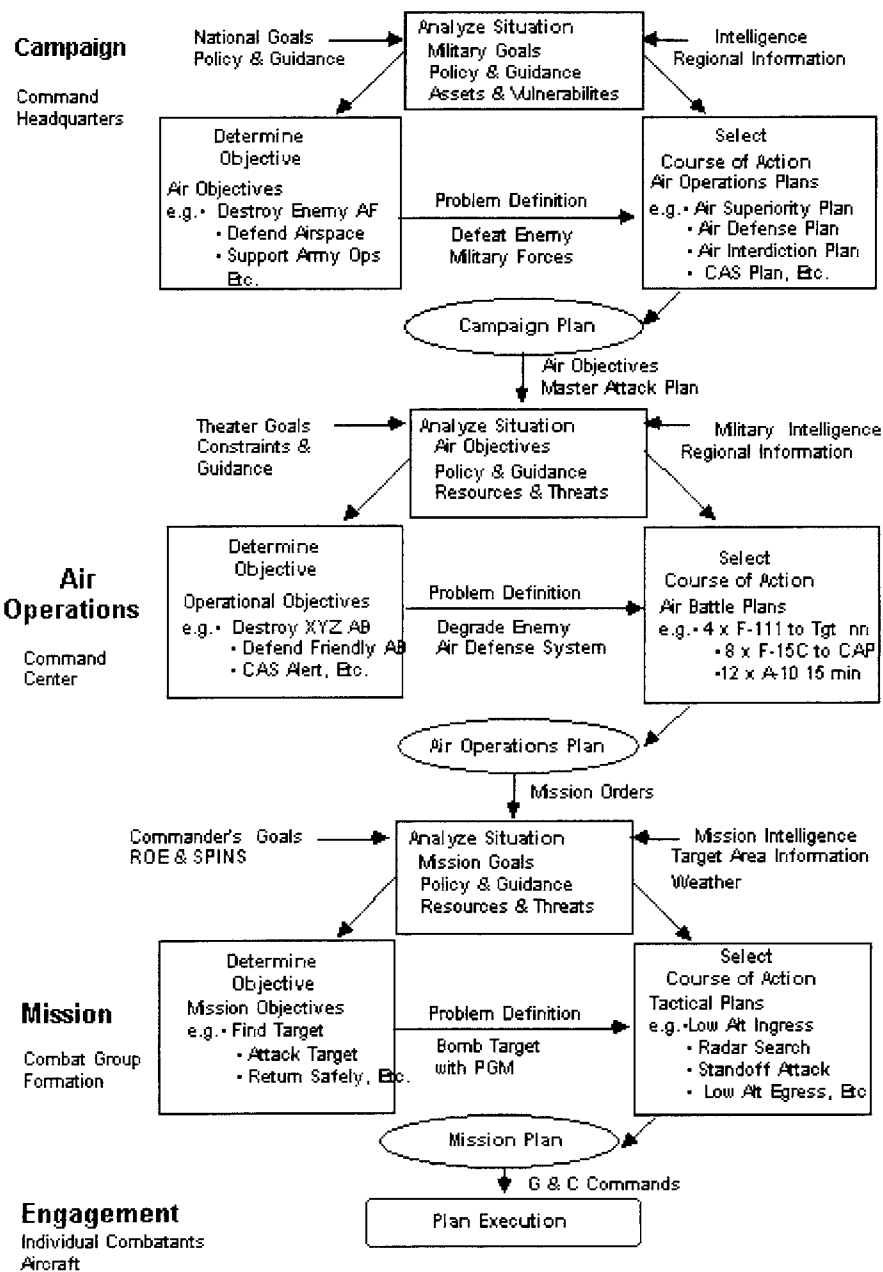


Figure 4-2. Example Air Campaign Planning Scenario [Maybury 1997]

application that has been developed is referred to as the Air Campaign Planning tool (ACPT) that assists in situation assessment, specifying objectives, developing courses of action, and allocating resources. Yet another application of MITRE's endeavor is a tool named Fusion. Fusion provides a database of intelligence information and possesses functionality very similar to the ANGELO system described in Section 4.1.



MITRE's study demonstrates the wide range of applications that may be used for situation assessment, mission planning, and execution monitoring. However, in this case, the applications are geared towards air campaigns. The most important characteristic of this paper is found in Figure 4-2. It provides an example of the hierarchical structure of a military command. Currently, CAIRO does not take into account issues of hierarchy in its collaboration architecture. Therefore, any new developments in the military arena should incorporate these issues.

## **4.2 Cross-Platform Compatibility**

Cross-platform compatibility refers to the ability of a program to be used with a number of different hardware configurations. The Java programming language is used by one group to develop a collaborative language called Java Collaborative Environment. The SHASTRA research effort provides an architectural design that could assist in developing flexible applications. And finally, a study into problems with PDAs is presented.

### **4.2.1 Java Collaborative Environment**

In a joint effort by Old Dominion University and the National Institute of Standards and Technology, the Java Collaborative Environment or JCE was designed and implemented [Abdel-Wahab, et. al. 1997]. This system was developed under the hypothesis that "each participant in a collaborative conference should be able to use whatever platform he or she prefers" [Abdel-Wahab, et. al. 1997 p. 112]. Thus, the authors also believe that Java is part of the solution to the platform independent concept since Java programs can be compiled and run on any system that is running a Java virtual machine. In this implementation, they built the JCE system architecture over the java.awt package through the use of subclasses. Using this approach, applications written in Java can immediately be extended to take advantage of the collaborative functionality.

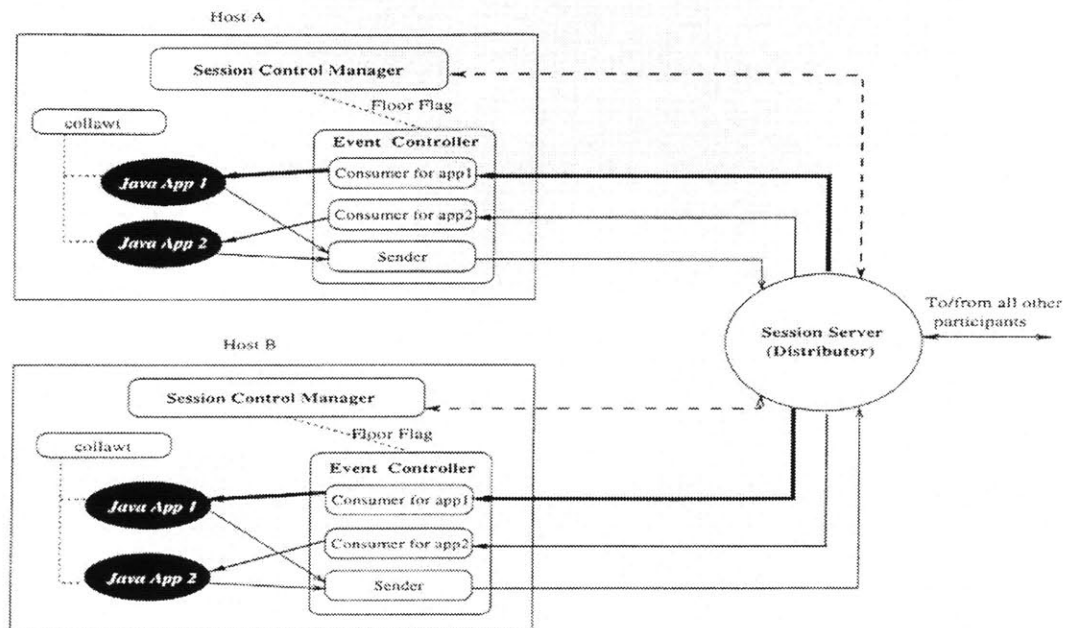


Figure 4-3. JCE System Architecture [Abdel-Wahab, et. al. 1997]

The JCE architecture is a fairly simple one as shown in Figure 4-3. It is comprised of three main elements:

- Session Control Manager (SCM): A graphical interface that provides options pertaining to the session itself. These functions include calling, joining, or leaving a session, starting applications, and requesting and releasing the floor. This component imitates the role of the Collaboration Manager to some extent.
- Event Controller: This element takes care of processing events from the Java application. It also can be broken down into two more sections: the Sender and Consumer. The Sender is responsible for intercepting events and deciding whether they should be sent to the Session Server. The Consumer, as expected, is responsible for capturing events from the Session Server and updating the application view appropriately. This model resembles the CAIRO system very closely. Although the specific methods are not so clearly defined with respect to one another, there are also `sendMessage` and `readMessage` methods within CAIRO that take care of the event handling capabilities for collaboration.

- **Session Server:** The server possesses three main functions: it distributes messages to participants, manages the group during a session, and takes care of floor control issues. Again, this server parallels the Forum Server in CAIRO. However, this system does not seem to provide the capability for multiple sessions to occur within the context of an overall collaboration. Furthermore, the use of a single server as an intermediary for all communication leads to the bottleneck issue that is so common in client/server architectures. In other words, during periods of high message traffic, the server may become overloaded and unable to operate at peak efficiency.

Again, the analogies between the architecture here and the CAIRO architecture are present. The focus on the Java programming language reinforces the choice of this language in a collaboration system implementation. However, the CAIRO system achieves many of the features of this JCE, but goes even farther to exceed JCE's capabilities.

#### **4.2.2 SHASTRA**

SHASTRA, another collaborative environment developed for engineering design purposes, was founded at Purdue University [Anupam & Bajaj 1993]. The significance of the SHASTRA system lies in its application architecture as exhibited in Figure 4-4. As can be seen, the design is made up of three main parts: the Application Engine, the Interface Mapper, and the Interfaces. The Application Engine encapsulates the basic functionality of the particular tool. The Interface Mapper is a mediator that sits between the user interface and the engine. It takes requests from the Interfaces and projects them to the proper functions in the Application Engine. Finally, the benefit of this architecture is that it provides a foundation upon which various interfaces may be easily implemented. With this type of flexibility, new interfaces that can be supported on less capable hardware platforms can be developed to resolve the cross-platform compatibility problem. This architecture can be adopted to support different types of displays.

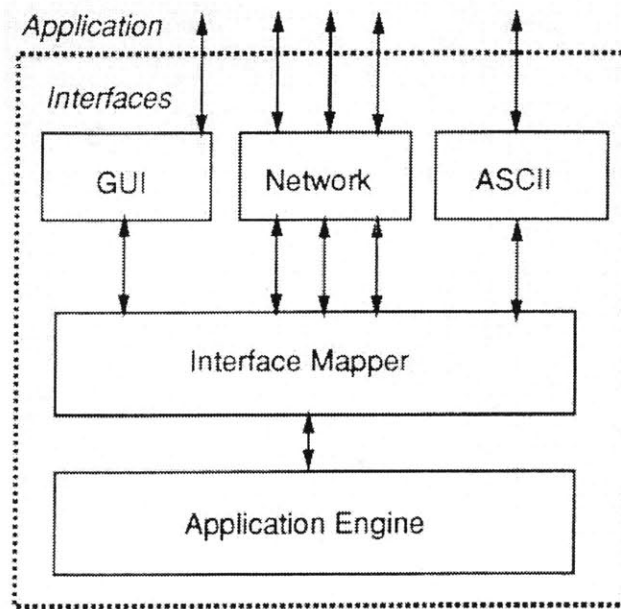


Figure 4-4. SHASTRA Application Architecture [Anupam & Bajaj 1993]

### 4.2.3 Problems with Personal Digital Assistants

In their paper, “Multimedia Client Implementation on Personal Digital Assistants”, Markus Lauff and Hans-Werner Gellersen [1997] discuss the limitations of using handheld computing devices as network clients. They attribute the difficulty in developing network applications to three main reasons:

1. Resource limitations
2. Network-level integration
3. Expensive PDA Software Development.

#### 4.2.3.1 Resource Limitations

Under the heading of resource limitations, the authors delve into more detail, citing problems such as small display, limited color depth, small memory, and weak

computing power as issues that must be dealt with when trying to develop network applications on PDAs. The following subsections address these issues more elaborately.

#### **4.2.3.1.1 Display Limitations**

In their dialog, two common display problems are chosen, each with a number of possible solutions. The first is the display problem encountered with tables that are too wide for the screen. In their text, they introduce a number of possible solutions:

- The first possible resolution is to use two scrollbars. Although double scrollbars seems like an intuitive answer at first, the disadvantage is that it can be very inconvenient to have to scroll across columns for each row. Thus, since one can not see all of the columns at a time, it may be difficult to track one's position within the table. To account for this problem, the first column can be locked in view, but again, this approach will in some sense waste that space on the screen.
- Another approach is to reformat the table so that the columns are divided among the width of the screen. Unfortunately, with a small screen, this may limit each column to only one word or perhaps even less. This result, of course, leads to the length of the table growing exponentially. A forced word break may alleviate the problem slightly although it will make the document harder to read.
- A third solution just gets rid of the table altogether and decomposes it into multiple sections as in Figure 4-5. This solution may be acceptable for tables in which the format is employed merely for aesthetic purposes. However, the downside for actual tables is that one loses the advantage of easy comparison of items that is made possible within a column structure.
- Finally, floating columns may be used to display the table within a small area. Instead of having the columns always beginning at the same horizontal position, the columns can start whenever the text ends. The authors suggest using different colors for the columns if possible to further distinguish them from one another

and to prevent confusion. However, as this method is not very conventional, it can be very confusing to the user and should only be used if completely necessary.

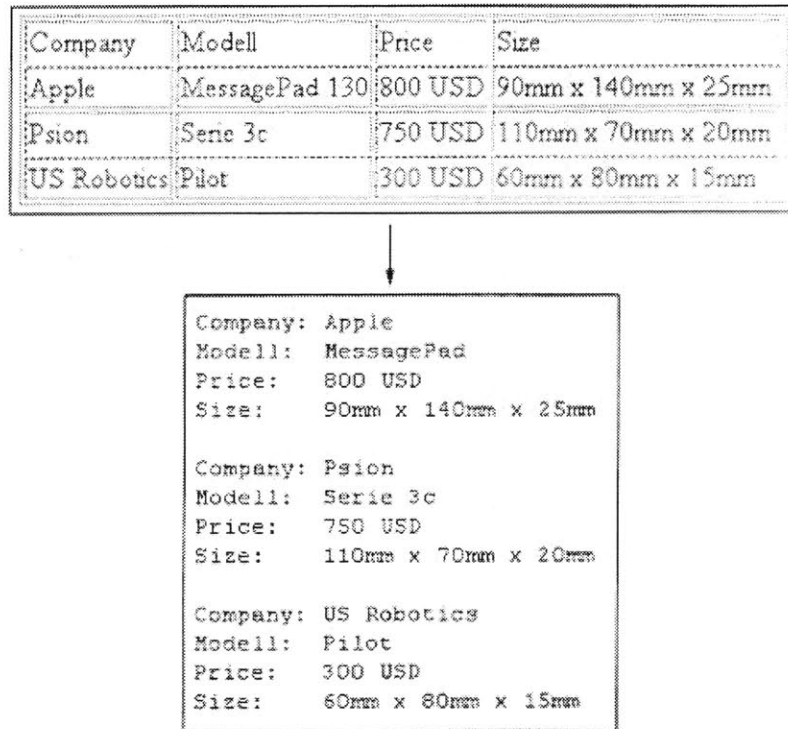


Figure 4-5. Replacing Table with Formatted Text [Lauff & Gellersen 1997]

Lauff and Gellersen identified a similar problem for images as well. Again, they present a few possible solutions:

- Once again, two scrollbars may be used to resolve the problem. This solution requires that the PDA have enough computing power to store the whole image and scroll through the picture. Another drawback is that it may be difficult for the user to picture the entire image.
- Secondly, the image may be scaled, but of course, depending on the original specifications and application of the image, the result may be useless. Thus, the

user may be given the choice of either a scaled image or the previous scrollbar option.

- Lastly, the image can again be ignored if it is not important to the document. This approach requires some sort of heuristic to be developed to determine the relative worth of an image.

Each of these examples elucidates the problems that may be encountered with the small displays available to Personal Digital Assistants. As more research is done in this arena, more problems are sure to arise. These issues should be analyzed further to allow the military planning collaboration system to account for these problems.

#### **4.2.3.1.2 Memory and Computational Power Limitations**

The small memory and low computing power of PDAs also presents a problem. To alleviate the situation, the authors suggest the use of a proxy server to preprocess a lot of the data. The PDA could send the server information about its dimensions and capabilities. The proxy then redefines the data as a bitmap to fit the capacity of the unit. This approach removes the restriction on the information format as well as the computing power of the PDA. On the other hand, the PDA must be capable of accessing a proxy server. In addition, the bitmaps themselves may consume a lot of memory. All in all, the use of a proxy server is probably a necessary evil that one must confront if PDA portability is desired.

#### **4.2.3.2 Network Connectivity**

Another issue with Personal Digital Assistants stems from their ability to connect to networks. As of the time of the writing of the Lauff and Gellersen paper, none of the PDAs had any true network capabilities. The Apple Newton MessagePad, the US Robotics Pilot, and the Psion Serie 3c all require a serial line to connect to either a PPP-server or another machine to establish a network connection. However, more recent versions of the Pilot and other new portable devices do have the ability to establish a

direct TCP/IP connection to the Internet with the addition of a peripheral wireless IP link such as the one provided by Novatel Wireless [Novatel Wireless, Inc. 1999].

### 4.3 Database Integration

In attempting to design a database architecture, familiarity with past efforts in this domain must be investigated. A number of systems use databases, but only a few were chosen to be presented here due to their applicability. These systems include SHASTRA, the Synchronous Collaboration Environment, and M-RAM systems. The findings are explored in the following sections.

#### 4.3.1 SHASTRA

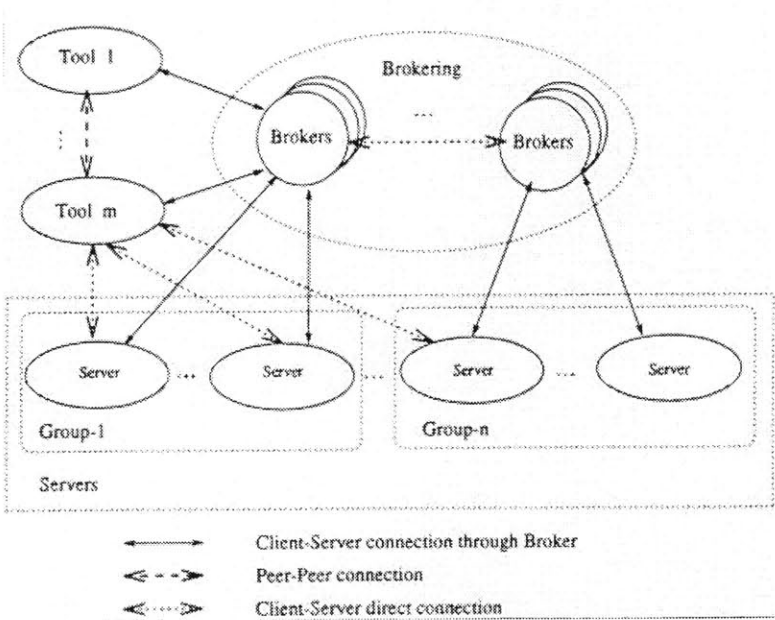


Figure 4-6. Brokered Connection and Communication [Bajaj, et. al. 1995]

Later research in the SHASTRA initiative [Bajaj, et. al. 1995] describes the concept of brokering to the collaboration scenario. As the authors point out, a “broker can provide a full transparent connection and communication between clients and servers as in a common object request broker system” [Bajaj, et. al. 1995 p. 210]. The broker in this



case (see Figure 4-6) exists to distribute the load of session management between the various session management servers. Its role is to reduce bottlenecks by sending requests to servers that are not overloaded. For the purposes of this research, this concept could be converted within the database context. Instead of creating direct connections to the actual databases, a “broker” serves as a mediator between the applications and databases, thus concealing the existence of multiple databases of information. This approach could lead to a cleaner abstraction and a separation of the user from the database implementation.

### 4.3.2 Synchronous Collaboration Environment

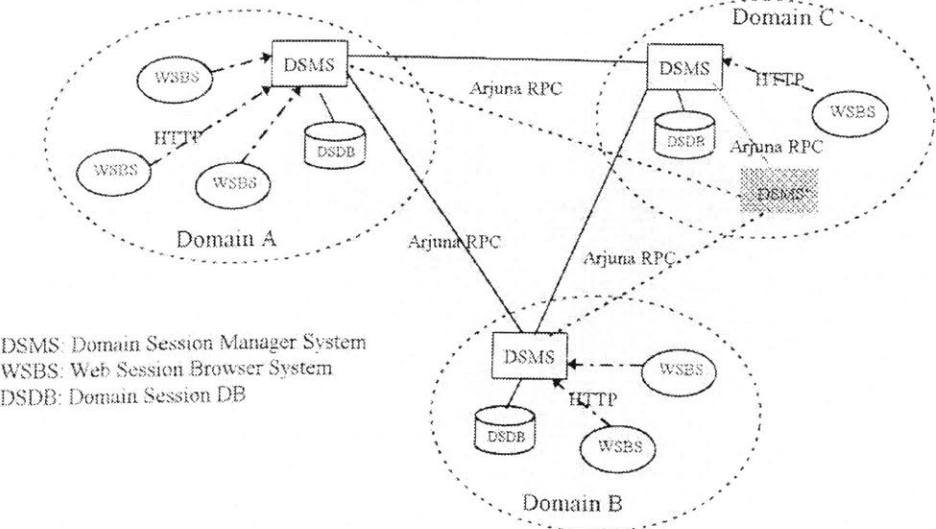


Figure 4-7. SCE System Architecture [Park, et. al. 1996]

The Synchronous Collaboration Environment (SCE), created at Konkuk University in Korea makes use of the Common Object Broker Architecture (CORBA) framework to implement a database compatible collaboration system [Kweon, et. al. 1997]. CORBA refers to a set of database standards instituted by the Object Management Group (OMG) [OMG 1999]. This system is built upon the concept of domains, which describes a certain range of users by their location much like the domain server concept on the Internet. The architecture for the SCE is portrayed in Figure 4-7. It consists of three major components:

- The Web Session Browser System (WSBS): The WSBS serves as the client within this architecture. Its purpose includes forming and registering to session. Each participant generates his/her own WSBS which provides an interface into the system
- The Domain Session Manager System (DSMS): This component acts as the server within the WWW environment. As the name suggests, its role is to manage a session within a particular domain. Each new WSBS is connected to the DSMS in its domain through the trading service provided by the CORBA middleware. The WSBS and DSMS regularly communicate through HyperText Transfer Protocol (HTTP), but the HTTP also interacts with CORBA through the Java gateway. Of course, there exist multiple DSMS's for various regions. They transfer information to one another using the Internet Inter-ORB Protocol (IIOP).
- The Domain Session Database (DSDB): Each DSMS possesses its own dedicated database. The database here simply stores information important to sessions. Not much more information is provided in the papers to explain their functionality within the system.

Although the design of the Synchronous Collaboration Environment is fairly simplistic in nature, it does provide some reasonable insight into the development of a collaboration system with an underlying database. Similar to CAIRO, the SCE technology was implemented using Internet technology such as HTTP and TCP/IP. Furthermore, the system makes use of the CORBA paradigm to establish an abstraction between the client and the server. This concept mirrors that of the broker described in Section 4.3.1. So once again, the notion of abstracting the underlying implementation of the system is exercised here as well.

### **4.3.3 M-RAM System**

M-RAM, short for Multi-Reasoning Artificial Mind, is another system developed at MIT [Soibelman 1998]. This system does not perform collaboration in the sense of

communication between two individuals. Instead, the M-RAM model conducts collaboration with an underlying database in order to retrieve information from it. To do so, it takes advantage of CORBA, which serves as an interface to the database server. Besides basic messaging services between the client and the server, the broker separates the application from the hardware configuration. This ability allows for portability of both the client and server, a feature that should be extracted and placed into the new system. Furthermore, M-RAM uses artificial intelligence algorithms to aid in engineering design. The algorithms used include:

- C4.5 – a machine-learning algorithm that generates classification models by analyzing patterns in sets of training data. A decision tree is created that can be used to classify new data.
- Case-Based Reasoning Agent – a reasoner applies similar situations that have occurred in the past and applies them to the present problem. It learns from previous successes and failures. Usually, a database is used to store cases that are then retrieved by the case-based reasoning agent.
- Genetic Algorithm – these algorithms simulate the biological process of genetics. In M-RAM, the genetic algorithm was used to implement the adaptation engine.

These algorithms could be applied in a similar manner such that the application could search the space of previous missions to help in the planning of the current mission by finding related protocols. This functionality may greatly reduce the time of mission planning by having the computer conduct the search.

## 4.4 Summary

By looking at other work that has been conducted within this research space, one can see that CAIRO is definitely a step in the right direction. In terms of meeting control structures, the implementation may be different, but the capabilities exceed anything that

has been done in other studies. CAIRO provides a choice of different meeting protocols to the user based upon the kind of collaboration that is desired. Furthermore, the addition of new protocols such as a baton-passing scheme can be implemented fairly easily with CAIRO's token concept. The idea of user focus presented by the SHASTRA project is something that is currently lacking in CAIRO, but would be a very strong addition to the interface. In addition, more research should be done on adapting CAIRO to asynchronous meeting efforts.

The study of platform portability considerations seems to show that the use of Java was a good choice for the base programming language. The flexibility of Java provides a solid foundation for eventual attainment of complete portability. The research by Lauff and Gellersen demonstrate the need for further investigation in the area of PDAs and their compatibility with legacy systems.

Finally, the research into the use of a database in collaboration systems revealed one thing—that databases have not found their way into many of these systems. The concept of a database provides a nice framework for a collaboration system. The systems described above provide a peek at some possible design issues to be considered. The Draper applications are being built around a distributed database architecture. However, the information policy adopted in CAIRO has to be altered in order for it to communicate with databases.

# Chapter 5

## 5.0 Implementation

The implementation phase of this project is divided into two main phases. The first phase consists of integrating some features taken from other systems into CAIRO. This phase allows some tangible experience with the design and implementation of CAIRO. In addition, these modifications enhance the usability of CAIRO, and at the same time, greatly expand its functionality for the future. Once a sense of comfort was reached with the CAIRO code, the integration with the Draper enhanced tactical node applications began.

### 5.1 Initial CAIRO Modifications

Before initiating the integration with the military planning tools, some simpler modifications were done on CAIRO. These modifications integrated some ideas and concepts from another system called Cliq! to gain some working knowledge of the CAIRO meeting environment. The Cliq! system was designed and developed at the Massachusetts Institute of Technology as part of the Distributed Software Engineering Laboratory (DISEL) class [Su 1998]. The goal of the laboratory class was to develop a medium scale software project dealing with collaborative learning. It was believed that some of these concepts for collaborative learning could be ported to CAIRO in order to provide a greater collaborative experience for planning and design as well. The three

main features that were targeted are Expressions, the Main Hallway, and Casual Contact. These features are explained in the following sections.

### 5.1.1 Expressions

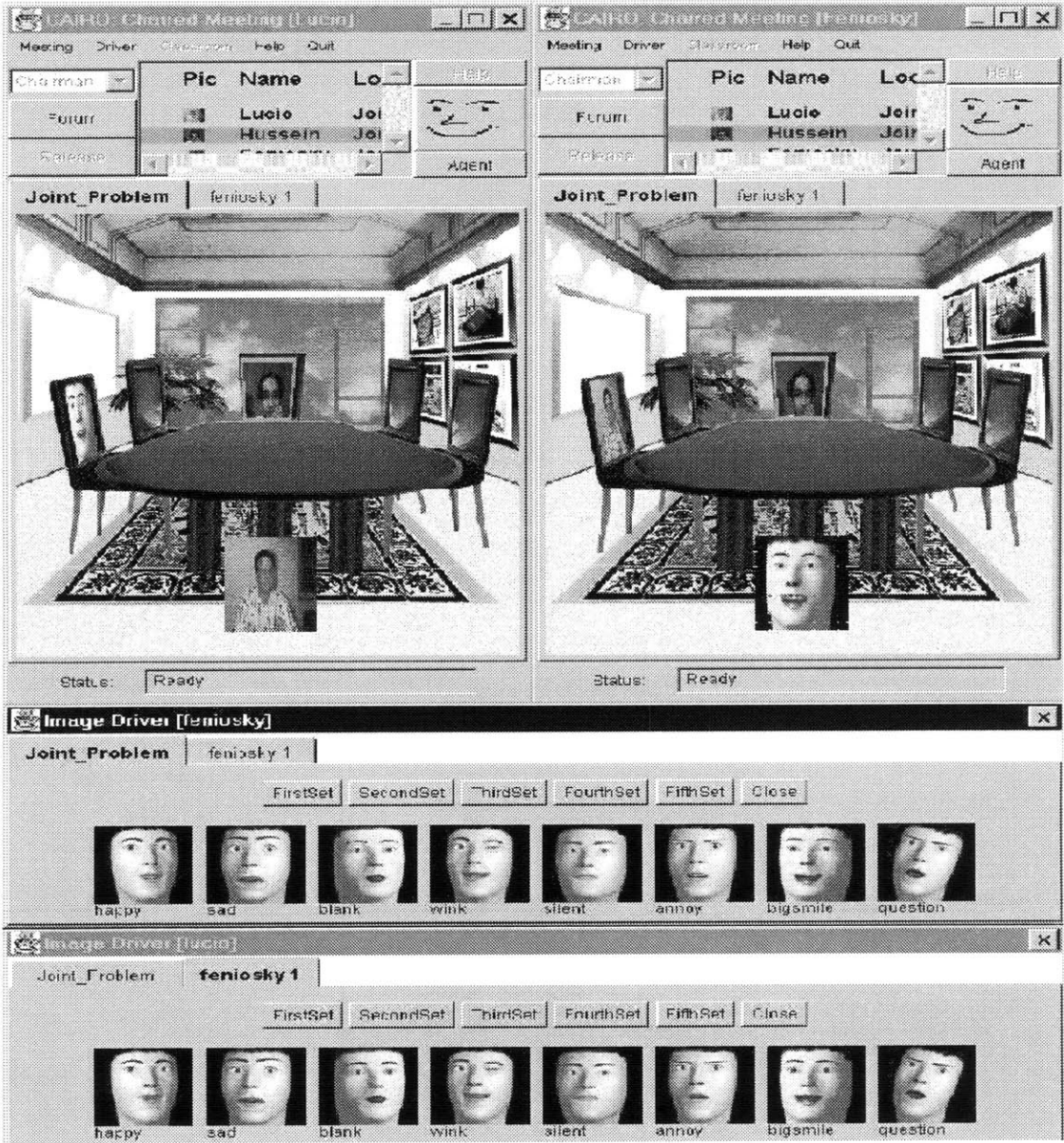


Figure 5-1. Snapshot of Expressions Tool

During a conversation, it is an added benefit to understand the emotions of the other individuals. Knowledge of another person’s mood can be an incredible asset during

an engagement and can in fact lead to a more effective discourse between those involved. Thus, the idea of an Expressions toolbar was conceived. This toolbar allows an individual using the CAIRO client to change his/her facial features within the collaboration interface to convey to other clients his/her feelings at a particular instant in time. This type of information can be conveyed in a visual fashion so that users do not have to explicitly state how they are feeling. Currently, the toolbar consists of a fixed set of images that are either cartoons or simple drawings.

Figure 5-1 presents a snapshot of two participants who have opened their Expressions toolbar. One of them has chosen a face with a big smile to exhibit his/her satisfaction. However, this functionality can be easily expanded to allow clients to take various pictures of themselves, which can then be used in place of these caricatures. In addition, the system has been programmed to recognize a series of emoticons in the text driver. Therefore, a participant can simply use one of the emoticons in his/her text driver and CAIRO automatically recognizes it and updates the user's image. Table 5-1 provides a list of the emoticons that are currently recognized by the CAIRO agent.

Table 5-1. List of Acknowledged Emoticons

<b>Expressions</b>	<b>Emoticons</b>
Happy	:-) :) =)
Sad	:-( :( =(
Blank	:-  :  =  :I =I
Wink	;-) ;)
Silent	:-x :x =x :-X :X =X
Annoy	:-/ :/ =/
Big Smile	:-!) :!) =!) :-D :D =D
Question	?:-  ?:  ?=  ?:-I ?:I ?=I

**5.1.2 Main Room**

Prior to the addition of the MAIN\_ROOM concept to CAIRO, users were unable to set up a meeting completely within the CAIRO system. In other words, participants would have to decide on a time through other communication means. For example, the chairman might send e-mail to all the meeting persons to let them know when they should log into the server and where they should go. The MAIN\_ROOM displayed in

Figure 5-2 provides a centralized meeting place where interactions are more random. Users who log into the CAIRO system are automatically brought into this main room. Thus, individuals may come to this initial screen and perform more casual interactions. In fact, users can randomly bump into people with whom they would like to speak. If this is the case, they can speak to them regularly through the text driver. In addition, a user may then decide to create a new meeting room where issues may be discussed in a group setting. If a participant does decide to do that may then he/she can create a meeting normally and then enter it. This feature merely provides a centralized location where users can meet and hold informal conversations or even see if a particular user is online. In a military context, one can make an analogy to the briefing room where officials convene to discuss general issues of importance to everyone. Afterwards, the group may separate into various forums to continue discussion of specific issues.



Figure 5-2. The Main Room

### 5.1.3 Casual Contact

Finally, another method for passive communication that has been termed “casual contact” was added to CAIRO. The concept of casual contact originates from the notion of public and private spaces. When a person works in his/her office or home, this location is considered to be a private area where he/she would not want to be bothered. However,



once an individual leaves the confines of a private space to, for example a hallway, he/she has entered a public space. In the public space, there exists the possibility of a chance encounter with other persons. These encounters may easily lead to an engaging conversation. This idea may be extrapolated to military personnel as well. For CAIRO, the hallway in this analogy is the Internet while the office is one's own personal computer. When users are utilizing the network, other persons logged into the CAIRO system are notified of their presence. An observer can decide to hail that person if they so choose.

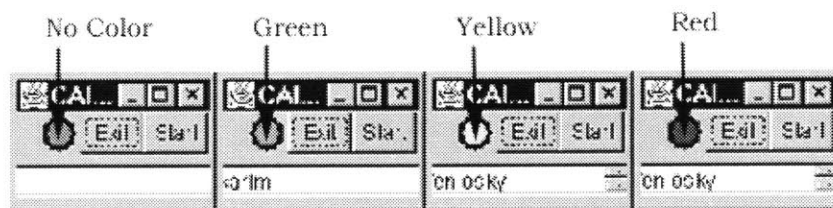


Figure 5-3. Casual Contact

Figure 5-3 displays the four possible states of the casual contact interface. The colored circle reveals the state while the two buttons allow the user to either enter the system or exit it. At the bottom is a list that identifies the users that are on-line. No color represents that nothing is going on. The green circle denotes that the user himself has entered the public domain. A yellow circle signifies that another user has entered the public space. And lastly, a red circle indicates that another user has hailed the user. If this happens, he/she can then enter the CAIRO system to the MAIN\_ROOM specified in the previous section to see who has requested to speak with him/her.

## 5.2 Development of the Prototype

In creating a prototype system which combines the planning capabilities of the Geospatial and Temporal View applications with the collaboration features of the CAIRO system, the standalone planning applications must be modified to take advantage of the message passing paradigm that is employed through the servers of CAIRO. As explained previously in Section 2.1, the drivers in the CAIRO system pass information about

specific keyboard and mouse events and pass them directly to other clients who are interested in that information. In order to allow each of these applications to collaborate within the CAIRO model, messages have to be devised to respond to these events and take advantage of the CAIRO structure to pass them along to the necessary clients.

### **5.2.1 Temporal View Integration**

The Temporal View application was written using the Java Development Kit by Sun Microsystems [Sun Microsystems, Inc 1999]. Since CAIRO was also developed using Java, the integration of the two systems was not too difficult. By taking advantage of many of the methods for sending messages between drivers, code was added to capture the appropriate mouse and keyboard events and send them to the other clients. In addition, the system had to be edited so that it could process these messages and simulate the proper events on the receiver's interface. Specifically, messages were added to deal with the adding and deleting of locations, and directed movement lines. Moreover, the dragging of these icons also was passed along from one client to another.

One problem that was encountered in the Temporal View integration was that the tabs that are used for side-talks could not be created to encapsulate an entire window frame. Therefore, the side-talk capability for this new driver had to be disabled. The reason that this problem arose lay in the fact that as new features such as the Side-Talk were added to CAIRO, certain abstractions in the software design became indistinct.

### **5.2.2 Geospatial View Integration**

The integration of the Geospatial View required much more effort. Since the system was developed with Microsoft's Visual C++, some technique of cohesion between the Java code in CAIRO and the Geospatial code had to be applied. JDK provides an interface to other languages in its Java Native Interpreter (JNI) package [Stearns 1999]. The employment of this method required the creation of a Dynamic Linked Library (DLL) containing a library of the C++ functions needed. Java would then access the functions within the library through a dedicated interface.

In attempting to achieve this integration using JNI, many problems were faced in converting the original executable code created using the Microsoft Foundation Classes into a DLL. The generation of a usable DLL was not a trivial task and soon became very time consuming. When a working DLL was finally produced, it was later determined that a dynamic linked library could not access window operations that were crucial to the Microsoft Foundation Class implementation. Instead, the user interface for the application would have to be separated from the underlying application engine to facilitate an integration. In this case, the Java code would generate the windows and other user interface components while the C++ code would simply contain the code for the application engine. Therefore, it was discovered that the incorporation of tools not developed in Java could not be easily achieved with the current architecture unless the applications were developed in a manner that would allow easy separation of the interface from the software implementation.

### **5.3 Summary**

The first stage of the implementation phase resulted in enhanced capabilities of the CAIRO system. The ability to use expressions to convey one's emotions definitely adds to the collaboration experience for the users. In addition, the concept of a public interaction space was explored and fulfilled through the introduction of the Main Room concept and casual contact.

Integration of the Java-based Temporal View was successful. Its functionality was verified with up to five users on separate Windows NT Workstations. In addition, the system ran on Solaris platforms simultaneously as well. Although the complete prototype was not ever fully realized, the experience did provide some useful insight into the limitations of the current CAIRO architecture. More precisely, it was discovered that the incorporation of new applications into drivers within the system was not a trivial task that consisted merely of adding event-messaging capabilities. The functionality of CAIRO had expanded to the point where certain abstractions were violated in the design.

Furthermore, the eternal dilemma of consolidating disparate programming language became an issue as well.

# Chapter 6

## 6.0 Results

From the beginning, it was thought that many of the ideas and concepts brought forth from the CAIRO meeting environment could be taken and applied to a new military collaboration system. This system would be built upon the foundation of a database notification scheme. From the work that has been done here, one can see that many of the collaboration concepts currently found in CAIRO are indeed applicable in a military setting. Furthermore, the current CAIRO architecture can be modified slightly to include the new database design. However, the entire system will probably need to be rebuilt to allow full utilization of the database structure and to enable an uncontaminated implementation. In this section, the system requirements and the new design are revealed.

### 6.1 Database Notification vs. Event-Passing

Before attacking the problem of how to move from an event-passing collaboration model to a database notification model, some discussion about why the collaboration model itself should be transformed is in order. Overhauling an entire software package is a taxing procedure that most programmers would like to avoid. Therefore, a case must be presented to demonstrate why a change to a new communication protocol, which will result in a complete reimplementaion of the collaboration system, is necessary. Some of these issues were disclosed in the last chapter. However, the crucial topics deal directly with data.

## 6.1.1 Definitions

Following is a list of affected issues pertaining to data along with a brief explanation of what each term means:

- **Data Persistence** – The ability of data to sustain its state from one time period to another.
- **Data Synchronicity** – Assures that data among all distributed clients is identical during specific time periods.
- **Data Transfer** – Moving data from one collaboration instance to another.

**Table 6-1. Comparison of Event-Passing and Database Notification Models on Selected Issues**

Issue	Event-Passing	Database Notification
Data Persistence	Data has to be explicitly saved in some pre-specified data format to be persistent.	All data are updated to the database, which allows data to be stored from one session to another.
Data Synchronicity	As long as all events are transmitted and received correctly, data should be synchronized. However, data can lose synchronicity if event transfer does not occur correctly.	Since all data are updated through the database, synchronicity is maintained as long as various views update periodically with the database. There may be short time spans of asynchronicity.
Data Transfer	Transfer of data from one collaboration instance to another is a difficult task that is yet to be implemented. Either a list of events has to be passed from one instance to another or the data has to be saved to disk.	Moving data from one instance to another can be accomplished by simply specifying the location of the desired information within the proper database.

## **6.1.2 Comparison of Issues**

The use of a database in place of an event-passing scheme has many advantages. These advantages are outlined in Table 6-1, comparing various issues between the two models. From the table, one can see that the database would either ease or eliminate the problems of data persistence, synchronicity, and transfer altogether. All in all, the introduction of a database will provide a more robust collaboration framework to insure the data integrity for the clients.

## **6.2 Requirements**

Requirements for the system are based on the three issues discussed in Chapter 4. Once again, they are meeting protocol, platform compatibility, and database integration.

### **6.2.1 Meeting Protocol**

As mentioned previously, the meeting protocol concept describes the controls that create a structure for the on-line collaboration effort. Once again, the notion of meeting styles is revisited from a military standpoint. In addition, the focus on military situations warrants at least some investigation into the well-defined command hierarchy found in military circles. These issues are explored in the ensuing sections.

#### **6.2.1.1 Meeting Styles**

The meeting structures presently defined in CAIRO seem to be sufficient for collaborative situation assessment, mission planning, and execution monitoring. Below is a list of the proposed meeting styles and their applicability in each of these circumstances:

- **Freestyle Meeting Style** – This format can be used mainly in the situation assessment stage for identifying important information about the mission. In

addition, early stages of mission planning may require brainstorming about possible response scenarios, which may warrant this style of engagement.

- **Chairman Meeting Style** – Should be used for the meeting planning phase in which the actual formation and definition of a meeting plan is made. This way, suggestions for a plan can be made in an orderly fashion.
- **Lecture Meeting Style** – A formal briefing session where staffs explicate their proposed plans would follow this control strategy. During execution, leaders may employ this style with field agents to direct their movements.

The introduction of the database into the design of the system establishes a setting in which persistent data may be stored. Thus, the forum server discussed in Section 2.1.1 can be easily converted into a database maintaining much of the current functionality. This database can hold much of the meeting style control information as defined rules within the database itself. Furthermore, user information such as rank can be loaded into the database from a Name Server to provide a greater range of information that can be used to help facilitate the meeting in a more efficient manner.

### **6.2.1.2 Command Hierarchy**

In defining a set of meeting protocols for a collaborative planning system, some attention is diverted to the role of a command hierarchy. The hierarchy can be simplified for research purposes as a simple tree structure of units. The nodes represent meetings within units. Since the tree itself is duplicated at each level, one can further simplify the model to just three nodes of the tree as illustrated in Figure 6-1. This cluster of nodes represents a hierarchical relationship. Sets of these basic units can be joined together to form any arbitrary tree structure imaginable with the exception of the one or two node cases, which are degenerate cases.



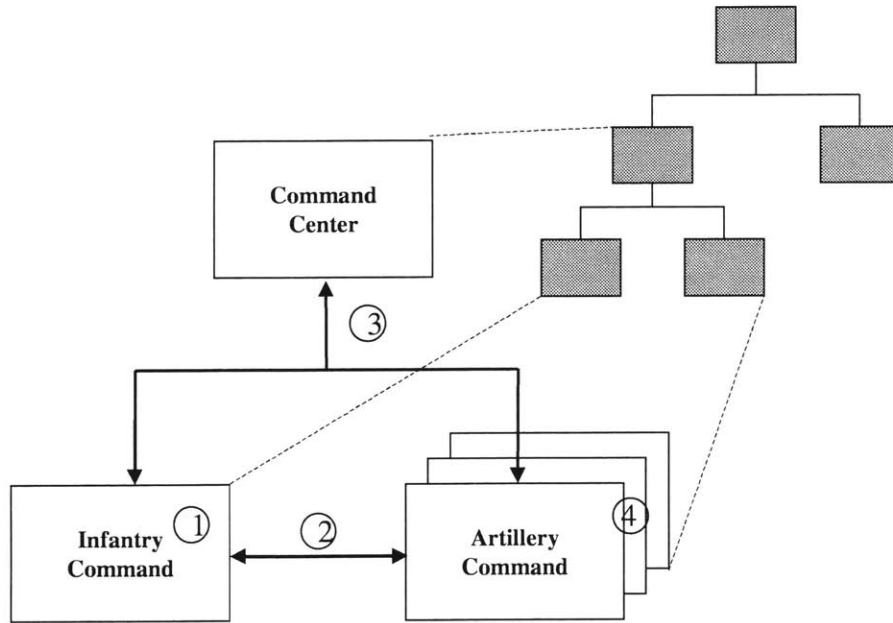


Figure 6-1. Basic Structure of Command Hierarchy

The text within the boxes in Figure 6-1 is given only to represent examples of unit functions. The uppermost box represents a higher-ranking unit than the two boxes below it. The overlay of boxes in the lower right node marked “Artillery Command” symbolizes the concept of versioning. In terms of operational planning, it is quite feasible that a certain situation may be passed along to a number of teams to determine separately distinct options. These options from subordinate units can then be passed to the next higher level for consideration.

Figure 6-1 identifies four main paths of information policies. These paths are identified by the circled numbers on the diagram and are discussed in that order. (1) First and foremost is the flow of information within a single node. The node represents a single collaborative meeting. Information flow within a single meeting is a problem that CAIRO has solved successfully. (2) The second information flow is between nodes of the same level. For example, there may be a need to transfer data from the Infantry Command to the Artillery Command. A change to the placement of an infantry unit may be important for the artillery commanders to avoid fratricide. Therefore, this information must be delivered to the Artillery Command so they can process the information accordingly.

③ In the third case of information flow, data moves from one level to another. As stated earlier, plans defined by a lower level must be propagated to the upper levels for consideration and integration with the overall mission plan. Alternatively, orders and mission parameters need to be passed to lower levels in real-time so these levels can develop plans with up-to-date knowledge of the situation. ④ The fourth case deals with contingency meetings where versions of the same scenario may be created for various groups to work on simultaneously. In this case, there should instead be no transfer of information between the nodes that overlap one another since their plans are distinct. Each case of information flow is analyzed and resolved in the final architecture. The architecture along with its responses to these claims is explored in Section 6.3.

## **6.2.2 Cross-Platform Compatibility**

Members of command staffs across a breadth of forces and military organizations are more likely to be able to engage in electronic collaboration if the collaboration system and drivers can operate on a variety of hardware platforms and operating systems. The system should be able to work in computers with minimal memory and display abilities. The range of machines that should be available for operation should encompass systems from supercomputers to the limited Personal Digital Assistants (PDAs) available in the market today. In addition, it should be capable of running on PC, Macintosh, and UNIX environments.

By creating a distributed architecture where applications are independent and disconnected from one another, the concept of compatibility can be achieved. As long as requirements that all applications must adhere to are defined within the database notification model, applications should be able to collaborate with one another using the underlying database structure. In addition, independent applications may be developed separately to operate on a specific platform. If further abstraction is desired, the modular design architecture for the different software tools can emulate the SHASTRA model presented in Section 4.2.2. The functionality of applications can also be customized for each hardware platform by providing flexible installation procedures. These installation

procedures would only distribute the necessary components to the target system, thus conserving disk space and memory.

### **6.2.3 Database Integration**

The database stores information during the collaboration process and facilitates the transfer of data from one client to another. Its role determines what data objects, contained within the database, users can access. In this way, the database is an addition to the forum server described in Section 2.1.2. The database basically works in conjunction with the forum server to conduct activities for the meeting. In effect, the database adds the following new functionality, which can be extremely beneficial if properly implemented:

- **Application registration:** Applications must be able to register interest in specific data. When an application changes this data, the database is then in charge of notifying all registered applications of the modification.
- **Maintain a set of access rights to data:** Each data item within the database maintains a list of access rights. These access rights identify the users that have the right to read, modify, or even have knowledge of the data.
- **Locking and unlocking mechanism:** To maintain synchronicity, the database must have the ability to lock and unlock the specific data objects. This feature prevents two or more users from trying to modify the same data at the same time. The data in the database should be defined as outlined by some predefined paradigm such that the size of the data objects is known. This way, a number of these data objects in the same database can be accessed simultaneously, and the database will not become a bottleneck within the system.
- **Copying data:** The database should also be able to create multiple copies of the data objects so that various users can work on the same problem

simultaneously. Copies can be an aggregation of subsections of various different databases.

The database version of the forum server retains many of the responsibilities of the original CAIRO forum server. The database, however, replaces the old event-passing model and also addresses many of the issues brought forth in Section 6.2.1.2, which deal with hierarchical information flow. Section 6.3 describes how these issues are addressed.

## 6.3 Proposed Architecture

First, the architecture being proposed in this research paper is described. Then, various collaboration scenarios are analyzed with respect to the proposed design. This analysis is intended to demonstrate its robustness. Finally, a synopsis of the results is provided.

### 6.3.1 The Architecture

Figure 6-2 depicts an instance of the architecture designed through this research effort. Here, some similarities can be seen between this diagram and the current CAIRO architecture (see Figure 2-1), but the main differences lie in the introduction of a database infrastructure. In this collaboration scenario, there exist one Name Server, two Session Servers, seven participants (designated by the dotted boxes), five different applications (APP), and a number of domain databases. Figure 6-2 also intends to convey that participants can use any number and combination of applications. Although only five applications are pictured here, additional applications can be added as long as they adhere to the communication protocol. Furthermore, users can even bring in their own databases for collaboration purposes such as Participants A and B do in Figure 6-2. The arrows represent communication links through which requests and notifications are sent. The fact that the arrows have two heads only means that the link is reciprocal. Dotted lines signify registrations to the Name Server.

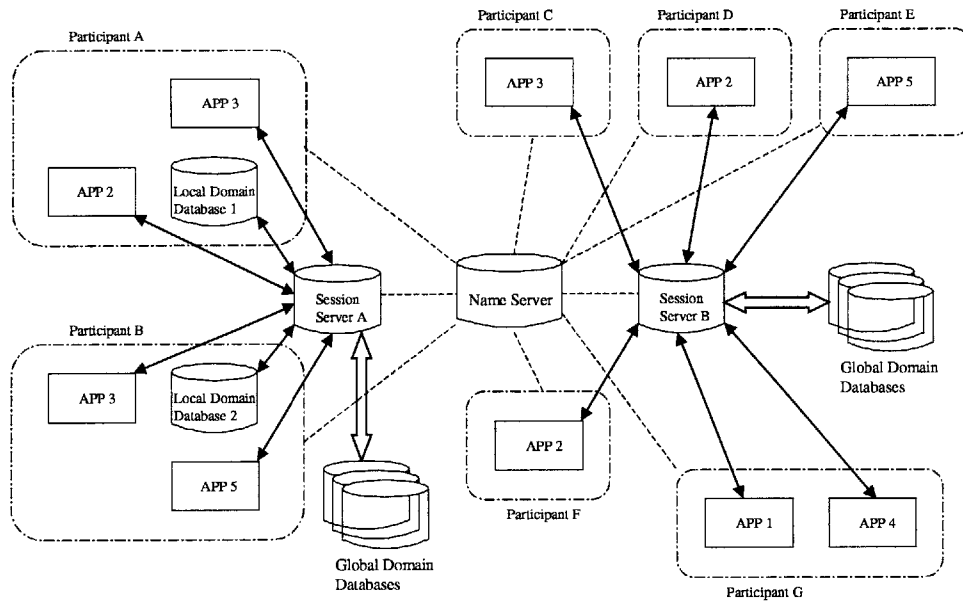


Figure 6-2. Proposed Architecture

### 6.3.1.1 Participant

A participant refers to a user of the system that registers with the Name Server. A participant may have a specific role in a collaboration instance. The simplest example is that of the chairman. He/she is in charge of a meeting and presides over the proceedings. On a deeper level, there may be participants that are in charge of certain functional units. He/she may have specific applications or databases that are dedicated towards his/her role and can bring any of these to a session.

### 6.3.1.2 Application

Any application (APP) that supports the notification paradigm of this system may be used within the context of this system. As opposed to the original CAIRO, the applications here are no longer bound together by a common Collaboration Manager. Instead, the applications are separate software packages that conform to the communication guidelines on how to define data objects within the database. These applications create a direct connection to the Session Server and communicate through it.

#### **6.3.1.4 Session Server**

The Session Server (SS) is quite similar to the Forum Server discussed in Section 2.1.2. However, it has been renamed as a Session Server to differentiate it from the CAIRO Forum Server, which has no database. The Session Server, in this design, basically consists of a Forum Server with a dedicated database. Rules for meeting procedures and agenda information are still encapsulated within this module along with user-related information. In contrast to the original CAIRO system, though, all communication is conducted through this component of the system rather than through the Message Servers described in Section 2.1.1.1.2. Applications that modify any data item first notify the Session Server of its changes. Then, the server acts as a broker to inform the registered domain databases and clients about the modification. The database in the server can use the same notification paradigm that is implemented for the domain databases. The session database must also keep track of the registered applications that are logged into it and information of interest to them. Moreover, it must also possess knowledge about all of the databases that are being accessed and keep them synchronized.

#### **6.3.1.4 Name Server**

The Name Server (NS) performs the same functions as the Name Server within the CAIRO system described in detail in Section 2.1.1. However, whereas the original CAIRO system saved directory information within volatile memory of the workstation that was running the server, the new system incorporates a database into the Name Server to enable this data to be persistent even if the server is down. In the former implementation of the CAIRO meeting environment, names and addresses of users and forums had to be logged in each time the Name Server was restarted, but now, they will be stored through downtimes. With a repository for persistent data objects, the implementation of a password verification paradigm becomes easier. Username and password pairs can be stored in some predefined manner within the Name Server database, and then compared when a user attempts to log in. A similar scheme can be used for creating sessions.

### **6.3.1.5 Domain Databases**

The Domain Databases (DDB) represent the storage locations of domain information. The domain information is the focus of collaboration sessions. There can be two types of domain databases: global and local. In Figure 6-2, the multiple databases designated as global domain databases represent databases that are accessed among a number of participants. The databases designated as local domain databases within Participant A and Participant B are local databases. These two participants can bring their databases to meetings. These two instances of domain databases are treated exactly the same within this architecture and are differentiated only by their location. The global databases reside on a public network while the local databases might exist on a local network.

The local database typically contains data that highly correlates with a participant's role in a collaboration session. The owner of a local domain database may only want to temporarily share its data with other participants during a single agenda item of a meeting. The local database can be viewed as another global database, connecting to the Session Server in the same manner. However, the main difference lies in the fact that the access rights described in Section 6.2.3 for the data in the database could be set to regulate whether other users can access the database. Thus, the owner would have permission to manually set these access rights, or they could be automatically set to change for specific agenda items. Thus, the main difference is that the access rights are dynamic in the local domain database implementation, but they are static in the global database.

### **6.3.2 Possible Resource Allocations**

One way to develop a model for a new collaboration system is to delineate various scenarios. For the military, numerous global domain databases may be available that contain information about various items of interest such as maps, functional units, and command hierarchies. This stored information may or may not be categorized into a

number of overlapping classifications. The Session Database serves as a mediator between the participants and these informational databases as outlined in Section 6.3.1.2.

Table 6-2. Database Dimensions

Geographical Locations Functional Data Units Data	Apple Database (L1)	Peach Database (L2)	Pear Database (L3)	Plum Database (L4)
Infantry Database (U1)				
Artillery Database (U2)				
Armored Database (U3)				
Cavalry Database (U4)				

This analysis focuses on two sample dimensions of database information although it can be easily extended to cover more dimensions. Two conceivable dimensions are geographical location data and functional unit data. These dimensions were chosen arbitrarily and merely serve as an illustration of a possible situation. Let Table 6-2 represent the viable allocation of data resources in this sample data space. There exists an intersection of four geographical location databases designated by code names and four functional unit databases. Now, it is feasible, as represented in this table, that each of the location databases contains information about all four of the functional unit databases (imagine that certain units are placed in each location). Likewise, each of the functional unit databases possesses data that pertains to all of the geographical locations. Thus, there is some overlap of information among these databases and each cell within the table represents the intersection of the databases of that row and column. In other words, if a session occupies a particular cell, then it must be using data objects that are replicated in the location and unit database designated by the row and column heading. The data objects found in the databases are simply resources being used by a collaboration instance. Therefore, if two sessions use the same resource, it becomes a shared resource. The following examples refer back to this table, and from now on, these instances are designated by the descriptor found in parentheses.



### 6.3.2.1 Sessions Using Independent Resources

A simple situation for collaboration occurs when two simultaneous meetings take place using separate databases. In other words, the two meetings are completely disjoint, and thus, require no interaction between the underlying Session Servers and Domain Databases. Table 6-3 depicts the same table layout as Table 6-2 but illustrates a possible database collaboration scenario. In this scenario, there are two meetings taking place, Sessions 1 and 2. The first meeting is discussing issues relating to the infantry located in location Peach while the second meeting covers the artillery unit in the Plum area. Thus, each session requires two databases, but there are no shared resources in this case.

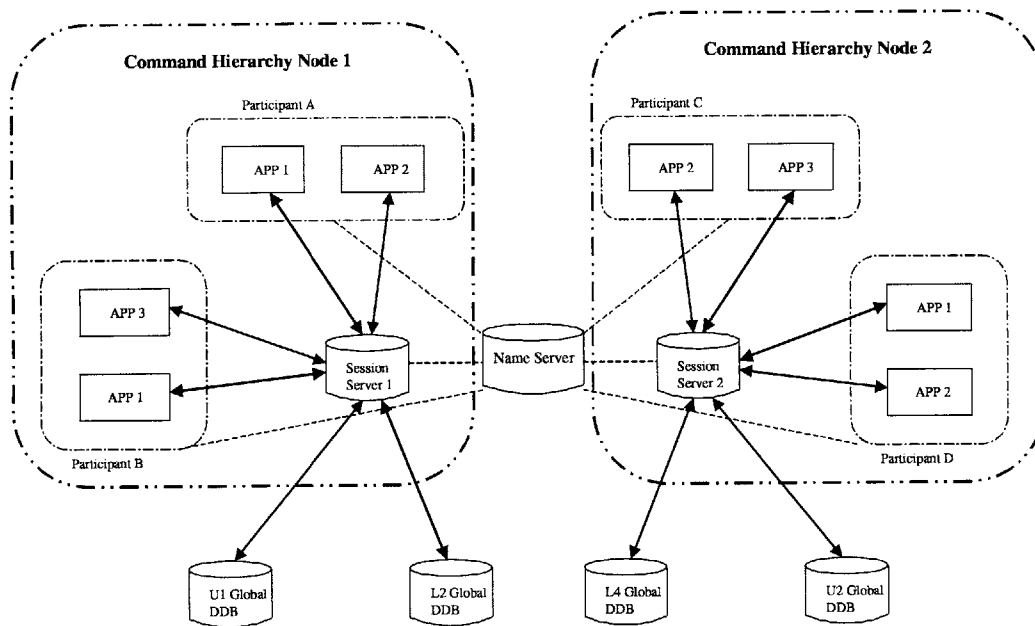
Table 6-3. Affected Databases for Sessions Using Independent Resources

Geographical Locations Functional Data Units Data	Apple Database (L1)	Peach Database (L2)	Pear Database (L3)	Plum Database (L4)
Infantry Database (U1)		▨		
Artillery Database (U2)				▩
Armored Database (U3)				
Cavalry Database (U4)				

- Session 1 Data
  - Session 2 Data

Figure 6-3 presents the initial state of the two separate meetings described above, which are being managed by Session Servers 1 and 2. Participants A and B collaborate in meeting 1 while Participants C and D collaborate in meeting 2. These collaboration instances can be viewed as two different nodes within the command hierarchy representation (see Figure 6-1). The bold dotted boxes distinguish the two meetings that are taking place. Each of the participants in this scenario uses two applications during his/her meeting sessions. This example can be extended to include more participants and applications but using only two participants suffices to illustrate the desired concept.

These meetings evolve into existence in stages. First, each participant logs into the Name Server. From here, they view the Session Servers, representing meetings, that can be entered. Once a participant selects the appropriate meeting, the applications they launch automatically register with that Session Server. Then, these participants begin to collaborate with one another through the global Domain Database. The Domain Databases themselves are linked to the corresponding Session Servers when the virtual meeting is first created. The arbitrator that initiates the meeting must identify the databases that are being used, and then registers them with the Session Server.



**Figure 6-3. Initial State of Sessions Using Independent Resources**

In Session 1, Participant A may modify information pertaining to the L2 Domain Database using App 1. Participant A's request is first sent to Session Server 1. Next, Session Server 1 insures that Participant A is permitted to make this modification. If so, the SS then determines which database or databases own this information, and sends an update request to those databases. An information item may contain data items that are either duplicated or spread across a number of domain databases. In this case, only one database, L2, is affected. Now, the L2 database must satisfy the change request. The L2 database first verifies that the data is not currently in use. This L2 DDB then locks the

data, makes the change in its copy of the data, and unlocks the data afterwards. Finally, the L2 database notifies Session Server 1 about whether the request has been accepted or not. Since the request was accepted, the Session Server notifies all registered applications of the modification. In this case, suppose the change affects information vital to both App 1 and App 3. Event notifications are sent to all versions of App 1 and App 3 that are registered (see step 4).

The meeting taking place in Command Hierarchy Node 2 illustrates an example of how a change in one application can affect multiple databases. Participant C, in this instance, alters some data using App2. This request is sent to Session Server 2, which in turn sends the information to the L4 and U2 databases. These databases return an accept or reject message depending on whether the data is available for modification. If either one rejects the information, then the change is not committed. Assuming it is accepted, the Session Server then disseminates the change to the appropriate registered applications (APP 1 and APP 2 in this example).

Figure 6-4 illustrates the flow of the data in the two sessions mentioned above. The sequence of events can be traced by following the series of numbered arrows in Figure 6-4. This figure depicts the flow of information within two separate nodes as denoted by number ① in Figure 6-1.

Each session here is independent and distinct from every other session in this example. An independent, distinct collaboration instance is the model implemented in the original version of CAIRO. However, the original version only notified applications that were of the same kind as the application making the data change. Here, App 1 and App 3 have both registered interest in the information in L2, so a change made with one application actually has an effect on the other application. In the original CAIRO, different applications were treated as separate entities and could not share information.

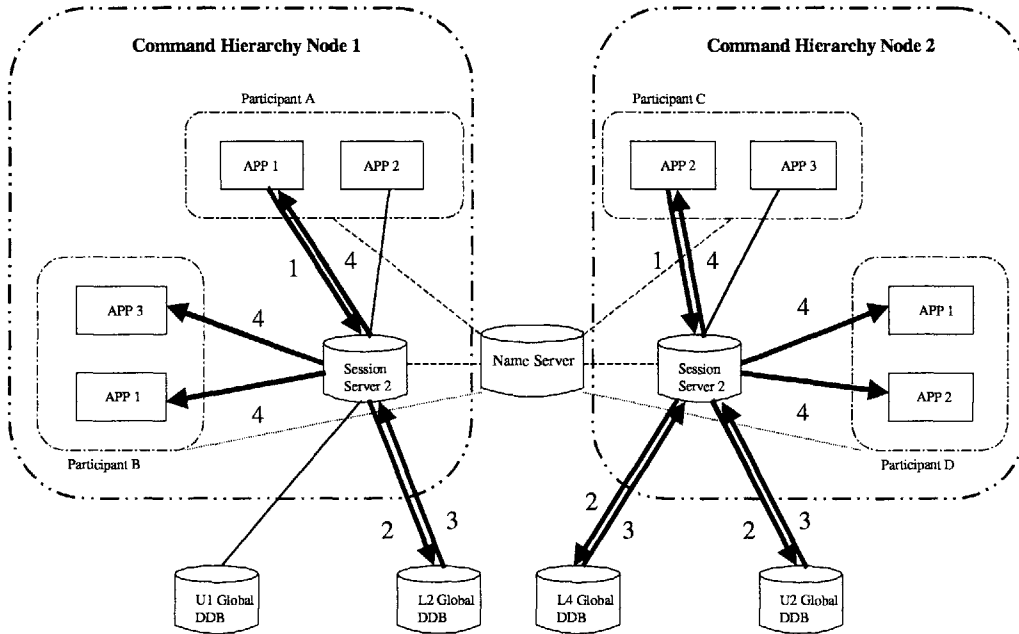


Figure 6-4. Information Flow for Sessions Using Independent Resources

### 6.3.2.2 Sessions Using Shared Resources

Now consider two meetings that are using the same resources. For example, one session may be in charge of distributing artillery units over all active military locations. Another session may only be interested in the placement of all functional units in location Peach. These meetings may require information from databases as depicted in Table 6-4. From this table, notice that Sessions 1 and 2 are both using information from U2 and L2. Therefore, they are in effect sharing these resources. If someone from one session changes the placement of an artillery unit in the Peach locale, the other session should be knowledgeable of this change.

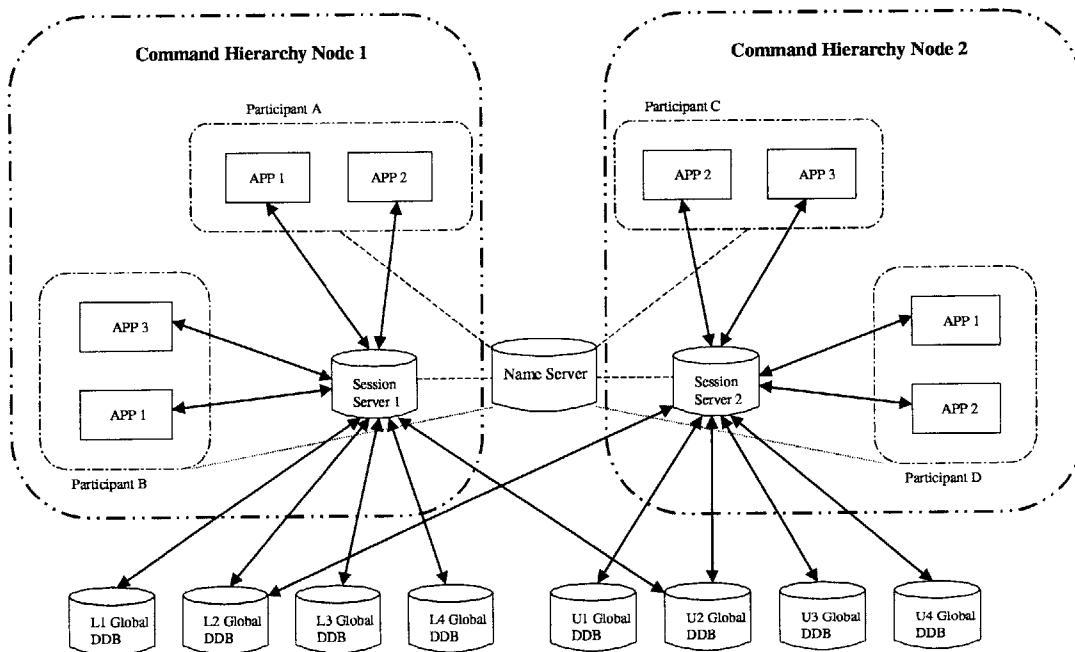
Consider, as in the previous section, only two participants in each meeting using two applications. Figure 6-5 depicts more complex connections that are necessary for this meeting structure. Notice that the L2 and the U2 Domain Databases are registered to both Session Servers 1 and 2 since both meetings require information from them.

**Table 6-4. Affected Databases for Sessions Using Shared Resources**

Geographical Locations Functional Data Units Data	Apple Database (L1)	Peach Database (L2)	Pear Database (L3)	Plum Database (L4)
Infantry Database (U1)				
Artillery Database (U2)				
Armored Database (U3)				
Cavalry Database (U4)				

 - Session 1 Data

 - Session 2 Data



**Figure 6-5. Initial State of Sessions Using Shared Resources**

What happens when Participant A modifies data in the L2 database as in the previous example? For Session 1, the same events take place as explained in Section 6.3.2.1. The difference exists when the L2 database recognizes that Session Server 2 is also registered to it. The L2 database passes the information along to SS 2, which then redistributes the data to registered applications. In this case, the data is sent to App 1 and 2 of Participant C and App1 of Participant D. Thus, if two sessions are working from the

same base information, a change in one will be reflected in the other. This information flow is portrayed in Figure 6-6 by the sequence of arrows numbered 1 through 4.

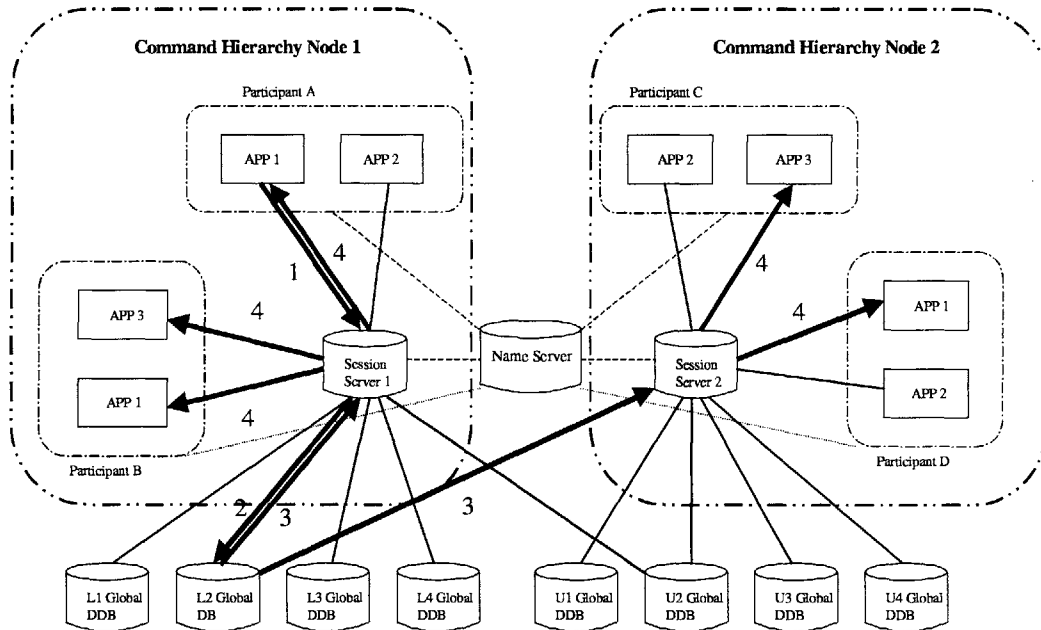


Figure 6-6. Information Flow for Sessions Using Shared Resources

This scenario provides an example of information flow between two nodes at the same level. Information transfer between two units at the same level is a feature of the basic command hierarchy denoted (2) in Figure 6-1. If one group is discussing the placement of artillery units within location Peach, the other group should be notified of the changes that have been made so they can plan accordingly. Basically, shared resources need to be distributed along to the collaborators who are sharing the resource. This architecture allows for easy dissemination of information among multiple sessions.

### 6.3.2.3 Hierarchical Sessions

Now, within the framework of the command hierarchy marked as number (3) in Figure 6-1, there exists yet another pathway of information—from one level of a hierarchy to an inferior level. This flow of information is the most obscure and requires

the most complexity. To think about this situation, realize that while traversing down a command hierarchy, the level of information detail usually increases as the information of interest domain to a node decreases. In other words, higher levels of a command hierarchy encapsulate information from lower levels. This encapsulation may be realized in a more general format than lower level units use. For instance, the higher levels of command look at the entire mission as a whole. However, they distribute sub-goals of the mission to lower level units to analyze, plan, and execute.

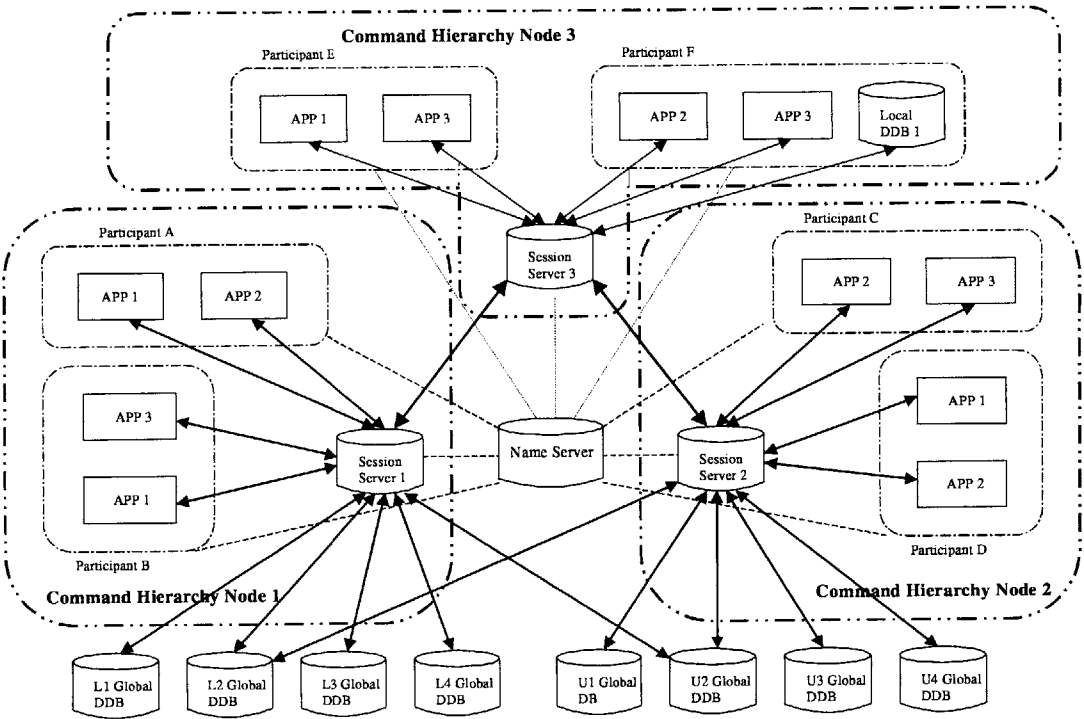


Figure 6-7. Architectural Representation of a Collaboration Hierarchy

This encapsulation concept is realized in this architecture by creating a communication link between the Session Servers as illustrated in Figure 6-7. In this representation, Session Server 3 represents the higher-level collaboration node. By registering with Session Servers 1 and 2, it automatically has access to all of its information without having to make a direct connection to each of the underlying external databases. This linkage itself creates a hierarchical structure among the three Session Servers to represent and support the actual command hierarchy. When the

participants in Session 3 change mission parameters, Session Server 3 notifies the two Session Databases below it. They, in turn, send that information to the registered applications for update, if necessary. Conversely, Session Server 3 has access to all the data in Session Servers 1 and 2, and can retrieve that information at any time to validate their planning process.

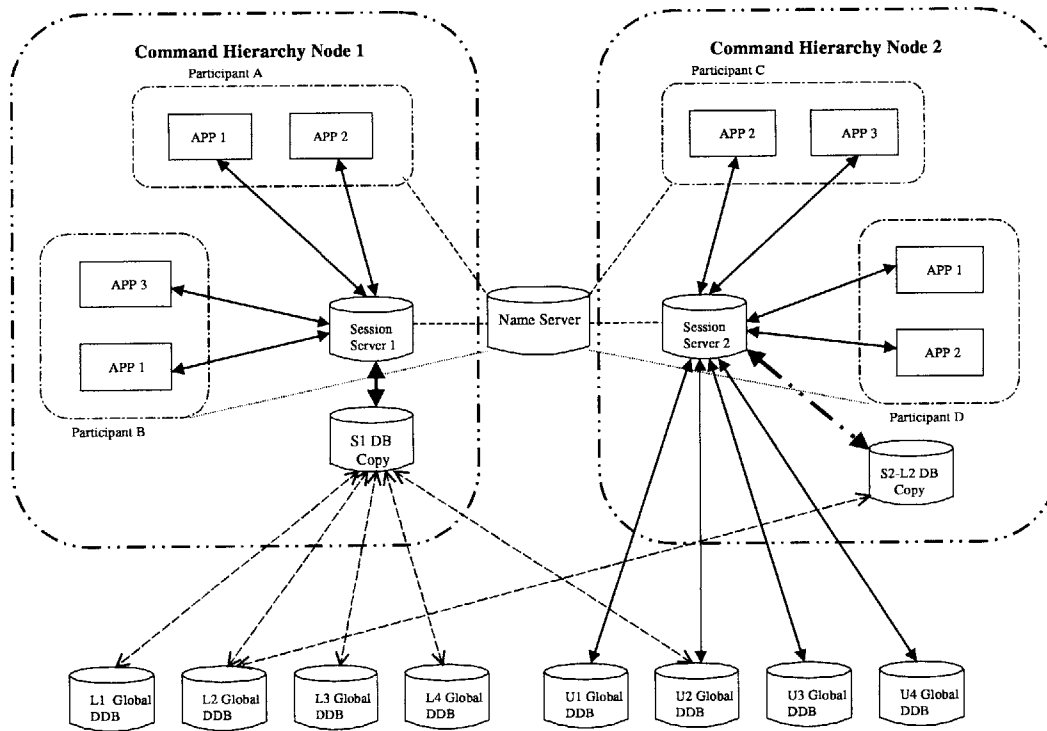
### 6.3.2.4 Contingency Meetings

Often during the course of military operations, it is conceivable that multiple units engage to devise alternative strategies for the same mission. Once the alternate plans are available, war-gaming and experience are used to evaluate and select an operational plan. This arrangement is referred to as versioning and is labeled number ④ in Figure 6.1. So far, the architecture does not explain how to support this versioning concept. There must exist a methodology through which multiple copies of the same data can be created.

The easiest method to achieve this goal is to use the database associated with the Session Server to create a copy of the pertinent information. This feat, in effect, creates a local copy for a contingency meeting that is taking place. When the collaborators are finished devising a strategy from their contingency meeting, then the changes can be committed to the original databases.

Figure 6-8 illustrates versioning within the context of the database architecture. Here, the two database copies are actually a part of the Session Servers to which they are connected. They are just depicted separately for comprehension purposes. The scenario on the left presents a situation where all of the affected data in the underlying databases is copied for versioning purposes. This situation is necessary if two groups are given the same mission objectives to satisfy. On the right, only the location database has been duplicated. In this case, imagine a situation in which the units are a resource shared with other groups, but the locations are not shared. Each session has free access to all of the geographical data, but they are confined access to certain unit data, namely the data in the L2 DDB.





**Figure 6-8. Architectural Representation of Versioning**

Figure 6-9 illustrates a sample flow of data through the versioning paradigm. For Session 1 on the left, Participant A generates a request to Session Server 1 for a certain data object. Session Server 1 queries the copy of the data it retains to determine if it possesses the requested information within its data items. For this demonstration, it is assumed that the information has not been extracted from the proper database. Therefore, Session Server 1 sends a request to *copy* instead of modify the data from the appropriate location, which, in this case, is database U2. This copy request is designated by a dashed arrow to emphasize the distinction. When U2 processes this request, it returns the corresponding data back to Session Server 1, which saves it in its primary database. Notice that the copy request sent to the U2 database is not propagated to the opposing session. Finally, Session Server 1 completes the transaction by modifying the data within the confines of its replica of the data and then notifies the appropriate client applications.

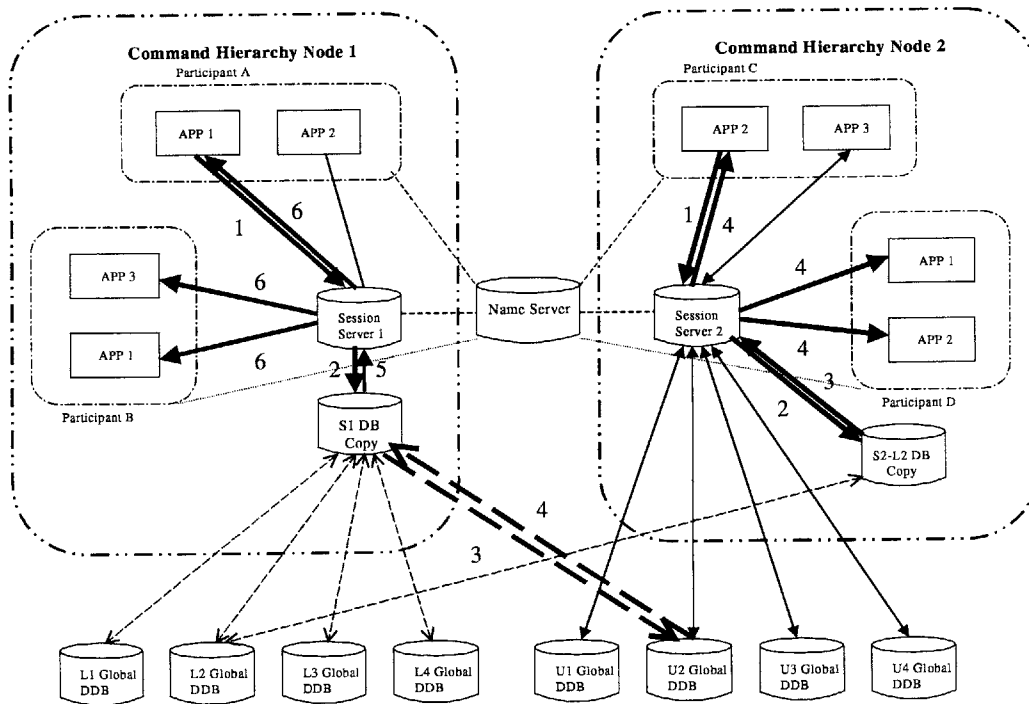


Figure 6-9. Information Flow for Versioning

The second case, in which the data being changed is already within S2-L2 DB copy, is outlined in Command Hierarchy Node 2 of Figure 6-9. This case is strikingly similar to the information flow presented for intersecting databases in Section 6.2.2. However, the main difference occurs in step 2, where the modification request is processed internally by Session Server 2 using its own copy of the data. Thus, the request is never seen by the real L2 database, but only by the S2-L2 DB copy. Therefore, the modifications are visible only to participants of Session 2 but are not available to other sessions that may be registered to Database L2.

These two examples encompass the possible operations that can take place within the versioning proposal. The introduction of the concept of duplicate data creates a difficult problem. Although the solution here may seem simple, its implementation requires meticulous attention to detail.

## 6.4 Summary

In this design, the Session Server takes the place of the Forum Server in the original CAIRO architecture. As expected, it retains many of the responsibilities of the Forum Server, but in addition, it takes on the function of a broker between the participants and the underlying databases. The Session Server mediates many of the transactions that occur within a collaboration instance. The architecture can support collaboration within a single meeting or many meetings. This chapter also elucidates a method by which versioning is accomplished as well as the incorporation of local databases to demonstrate the capacity of this architecture.

# Chapter 7

## 7.0 Conclusion

This research consisted of two main goals. The initial goal was to develop a prototype tactical collaboration system by incorporating military planning applications into the framework of the CAIRO meeting environment. The result of this integration would have provided a foundation upon which various scenarios could have been played out. As it stands, the integration was not wholly successful. However, the problems that were encountered did bring to the limelight some of the limitations of the current design. Not only is the communication paradigm outdated and unreliable, but the integration of new tools is not as easy as it should be. Especially in the today's electronic age, the key to success is to have a software package that is flexible and can be easily integrated with other systems. As was discovered, CAIRO at this stage does not provide this kind of service. The separation between the collaboration manager and the drivers has been blurred over the years. Furthermore, the connectivity to applications developed in other languages was poor.

The second goal of this thesis was to develop a more robust architecture for the CAIRO environment. The architecture was founded on a database notification schema that is able to support multiple applications in conjunction with multiple database repositories. The architecture was tested and refined by devising several conceivable tactical scenarios. The findings are detailed in Chapter 6 to provide the groundwork for further research in this area.

The next step is to take the architecture presented within this research effort and bring it to realization. The entire CAIRO system should be overhauled and rebuilt from the ground up to take full advantage of the new messaging protocol. The updated Name Server simply is an extension of the current Name Server implementation within CAIRO. The key to a successful implementation will lie in the development of the Session Server. Much of the code for the Session Server may be transferred from the original version of CAIRO, but some new source code must be developed to provide a methodology for registration. Applications and databases alike as well as other Session Servers will apply this methodology for registration. This implementation must also implement the request and notify processes within the databases. For the Collaboration Manager, some of the user interfaces such as the meeting backgrounds need to be updated to reflect a military setting. In addition, a view port to all of the meetings taking place within their hierarchical arrangement should replace the present hallway analogy. This way, users can easily recognize the relationships between the various meetings.

Finally, in the not too distant future, more research must be conducted into the information policy. Although the information policy is briefly touched upon in this document, the details need to be identified more concretely.

## Bibliography

- [Abdel-Wahab, et. al. 1997] Abdel-Wahab, H., Kvande, B., Kim, O. and Favreau, J. "An Internet Collaborative Environment for Sharing Java Applications." *Proceedings of the Sixth IEEE Computer Society Workshop on Future Trends of Distributed Computing Systems*, October 29-31, 1997, pp. 112-117. Tunis, Tunisia.
- [Ali-Ahmad 1997] Ali-Ahmad, W. "Adaptable Software Agent for Retrieval and Presentation of Information in Collaborative Engineering Environments." M.S. Thesis, September, 1997, Massachusetts Institute of Technology.
- [Anupam & Bajaj 1993] Anupam, V, Bajaj, C. "Collaborative Multimedia Scientific Design in SHASTRA." *Proceedings ACM Multimedia 93*, 1993, pp.447-56, 479-80. New York, NY, USA.
- [Bajaj, et. al. 1995] Bajaj, Chandrajit, Zhang, Peinan, Chaturvedi, Alok. "Brokered Collaborative Infrastructure for CSCW." *Proceedings of the Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 1995, pp. 207-213. Los Alamitos, CA, USA.
- [Benjamin 1998] Benjamin, Karim A. "Defining Negotiation Process Methodologies for Distributed Meeting Environments." MEng Thesis, May 1998, Massachusetts Institute of Technology.
- [Chang, et. al. 1997] Chang, C., Quek, F., Cai, L., Kim, S. "A Research on Collaboration Net." *Interactive Distributed Multimedia Systems and Telecommunication Services*, September 1997, pp. 228-233. Darmstadt, Germany.
- [Ethos 1996] "Internet II Offers Promise of More Techno-Fantasies." URL: <http://www.tagish.co.uk/ethosub/lit6/47d2.htm>. Ethos, May 21, 1999.
- [Hussein 1995] Hussein, Karim. "Communication Facilitators for a Distributed Collaborative Engineering Environment." B.S. Thesis, May 1995, Massachusetts Institute of Technology.
- [Hutchins 1999] Hutchins, Mike. Personal Communication. Jan-May 1999, Draper Laboratory.
- [Kweon, et. al. 1997] Kweon, M., Hong, S., Joung, S., Han, S., Kim, M. "Scalable and Reliable Synchronous Environment on CORBA Using WWW." *1997 High*

*Assurance Systems Engineering Workshop*, August 11-12, 1997, pp. 100-103. Washington, DC.

[Lauff & Gellersen 1997] Lauff, M., Gellersen, H. "Multimedia Client Implementation on Personal Digital Assistants." *Interactive Distributed Multimedia Systems and Telecommunication Services*, September 1997, pp. 283-295. Darmstadt, Germany.

[Lipnack & Stamps 1997] Lipnack, Jessica, Stamps, Jeffrey. *Virtual Teams: Reaching Across Space, Time, and Organizations with Technology*. John Wiley & Sons, Inc, New York, 1997.

[Maybury 1997] Maybury, Mark T. "Distributed, Collaborative, Knowledge Based Air Campaign Planning." URL:[http://www.mitre.org/support/papers/dc\\_kkb](http://www.mitre.org/support/papers/dc_kkb). MITRE, November 1997.

[Novatel Wireless, Inc. 1999] "Corporate Profile." URL: <http://www.novatelwireless.com/html/profile.htm>. Novatel Wireless, Inc., May 13, 1999.

[OMG 1999] "OMG." URL:<http://www.omg.org>. Object Management Group, May 13, 1999.

[Park, et. al. 1996] Park, S., Chung, I., Jo, G., Han, S., Song, K. "Fault-Tolerant Real-Time Synchronous Collaboration Environment Using WWW." *Proceedings of WORDS'96. The Second Workshop on Object-Oriented Real-Time Dependable Systems*, 1996, pp.226-31. Los Alamitos, CA, USA.

[Pena-Mora, et. al. 1996] Pena-Mora, F., Ali-Ahmad, W., Chen, H., Hussein, K., Kennedy, J., Soibelman, L., and Vadhavkar, S. "The Da Vinci Agent Society Initiative Progress Report as Fall 1996." *IESL Report No. 96-06*, Intelligent Engineering Systems Laboratory, Engineering System Group, Henry L. Pierce Laboratory, Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, November, 1996.

[Robert 1996] Robert, Henry M. "Robert's Rules of Order." URL: <http://www.constitution.org/trror/trror-00.htm>. Constitution Society, 1996.

[Soibelman 1998] Soibelman, L. "The Exploration of an Integrated Representation for the Conceptual Phase of Structural Design for Tall Buildings Through Distributed Multi-Reasoning Algorithms." Ph.D. Thesis, April, 1998, Massachusetts Institute of Technology.

[Stearns 1999] Stearns, Beth. "Java Native Interface." URL:<http://www.javasoft.com/docs/books/tutorial/native1.1/index.html>. Sun Microsystems, Inc., May 13, 1999.

[Strauss 1999] Strauss, Charlie. Personal Communication. Mar-May 1999, Draper Laboratory.

[Sun Microsystems, Inc. 1999] "Java Development Kit." URL: <http://www.javasoft.com/products/jdk/1.1/index.html>. Sun Microsystems, Inc., May 13, 1999.

[Vin, et. al. 1993] Vin HM, Chen M-S, Barzilai T. "Collaboration management in DiCE." Computer Journal, vol.36, no.1, 1993, pp.87-96. UK.

[ZDNet 1999] "Crypto Man: RSA Founder Discusses Future of Encryption." URL: <http://www.zdnet.com/zdn/content/pcwo/0124/pcwo0005.html>. ZDNet, May 21, 1999.