Domain-Specific Tools for Collaboration in Architectural Design

Wassim M. Jabi
University of Michigan
Doctoral Program in Architecture
Ann Arbor, Michigan
U.S.A.

ABSTRACT

By using a semantically significant and parsimonious representation of collaborative work in architecture an approach is demonstrated that allows the construction of a computer environment that can support collaborative design among geographically dispersed participants. A principal characteristic of this approach is a shift away from a focus on multi-user access to shared databases towards a shared protocol of interaction that is independent of implementation and storage schemes. To arrive at the components of this protocol an analysis of the nature of collaborative design was conducted in order to derive its syntactic and semantic structures. This paper will detail the argument put forth and demonstrate a possible solution through a discussion of the elements of a protocol of interaction and a brief description of a prototype Synchronous Collaborative Design Environment (SYCODE) that was implemented on two heterogeneous computer systems at distant sites.

1 INTRODUCTION

While research in the fields of Computer-Aided Design (CAD), Computer-Supported Cooperative Work (CSCW) and Distributed Artificial Intelligence (DAI) has produced impressive systems that solve the specific problems they address, no single system to date satisfactorily meets the goal of providing a cross-platform, synchronous environment that allows a group of architects to dynamically collaborate in designing an architectural project. Currently, there is a dearth of tools tailored specifically to the early-design stage.

The basic premise put forward in this paper is that by combining knowledge from design theory with techniques from CAD, CSCW, and DAI we can achieve an integrated solution that will overcome previous limitations and allow the development of computer environments that support collaborative design in architecture. By using a semantically significant and parsimonious representation of collaborative work in architecture, an approach is demonstrated that allows the construction of domain-specific tools that support the synchronous exchange of information with acceptable time delays and allow geographically dispersed participants to collaboratively solve design problems.

A principal characteristic of this approach is a shift away from a focus on multi-user access to shared databases towards shared protocols of interaction that are independent of implementation and storage schemes. Furthermore, a key factor in developing such a protocol is analyzing the nature of the domain it supports (collaborative design in architecture) to derive its syntactic and semantic components. Additionally, I claim that this protocol of interaction must accommodate a heterogeneity in hardware, operating systems, and graphical user interfaces.

This paper will detail the argument put forth above and demonstrate a possible solution through a discussion of a conceptual model of collaborative design and a prototype of a Synchronous Collaborative Design Environment (SYCODE) that was implemented on two heterogeneous computer systems at distant sites – Ann Arbor, Michigan and Hong Kong.

2 A CONCEPTUAL MODEL OF COLLABORATION IN DESIGN

Based on previous observations of architectural teamwork in a professional setting (Jabi 1996), a model has been constructed that describes the contributions of the design participants and indicates the various activities and transactions that took place during a typical design process cycle (figure 1). The model also indicates the chronology and intensity of some transactions. Additionally, during the observation of this case study, various sketches and drawings were collected, categorized by creator, and associated with the particular activities that produced them (figure 2). To cross-reference the artifacts and activities, a roman letter is included in the upper right corner of some activity boxes in figure 1. This roman letter corresponds to a sub-section in figure 2. For example, the activity "Transfer Drawings (Engineering Firm)" includes the letter "A" which corresponds to the artifact depicted in figure 2-A. Due to space limitations, many of the collected artifacts are not included in this paper. However, a more complete and interactive presentation of this model has been developed using the World-Wide Web at the following Universal Resource Locater: http://libra.caup.umich.edu/www/.

What follows is a description of the conceptual model elements and an explanation of the chronology of activities carried out within it. Based on that analysis, a protocol of interaction is proposed that maps the essential elements of the analyzed model. It is hoped that this protocol will form the basis of a domain-specific, computer-supported environment for enabling collaborative architectural design among geographically dispersed participants. Portions of the protocol have implemented within an early computer prototype that supports synchronous collaborative design.

Figure 1: **Conceptual Model of Teamwork in Architectural Practice**

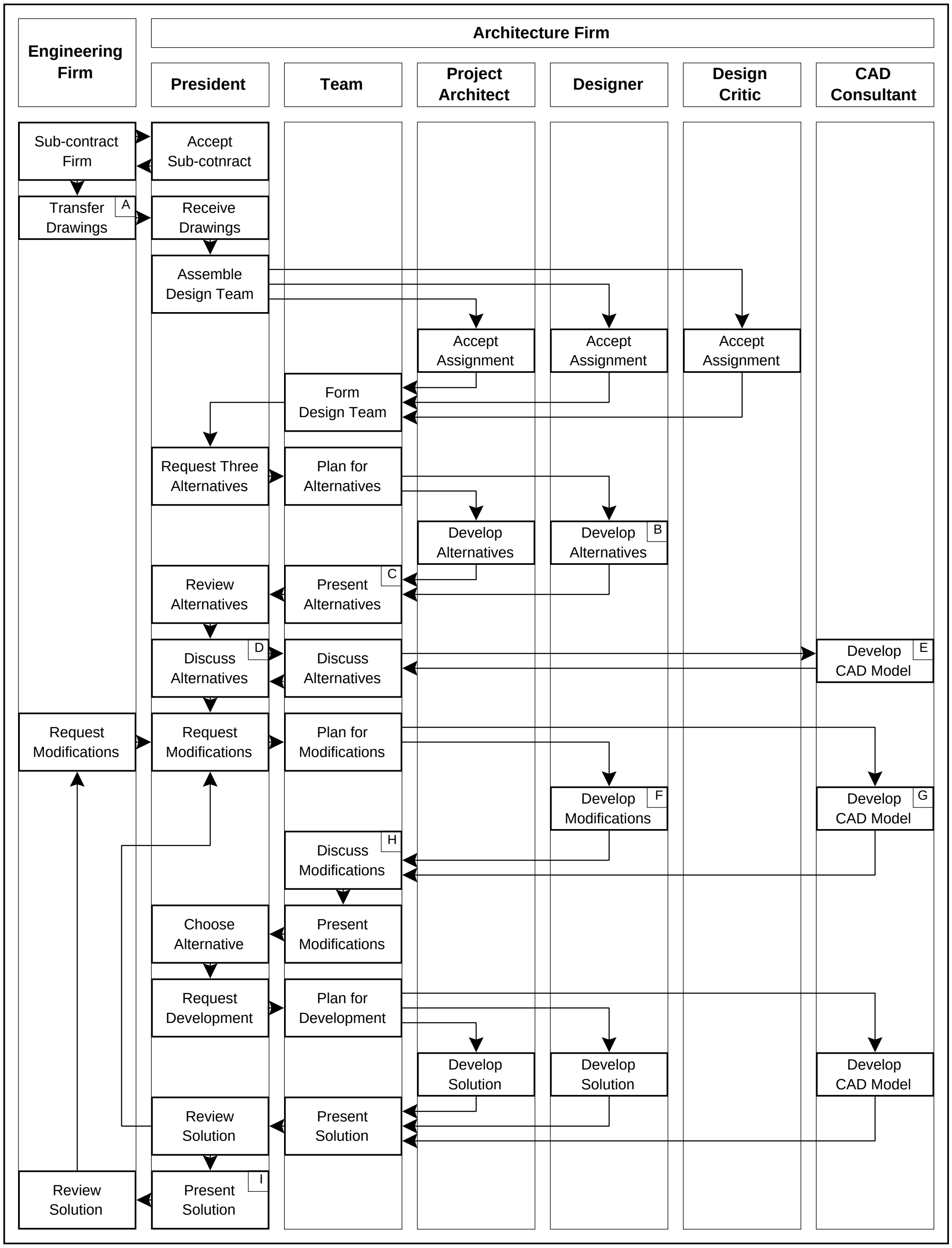
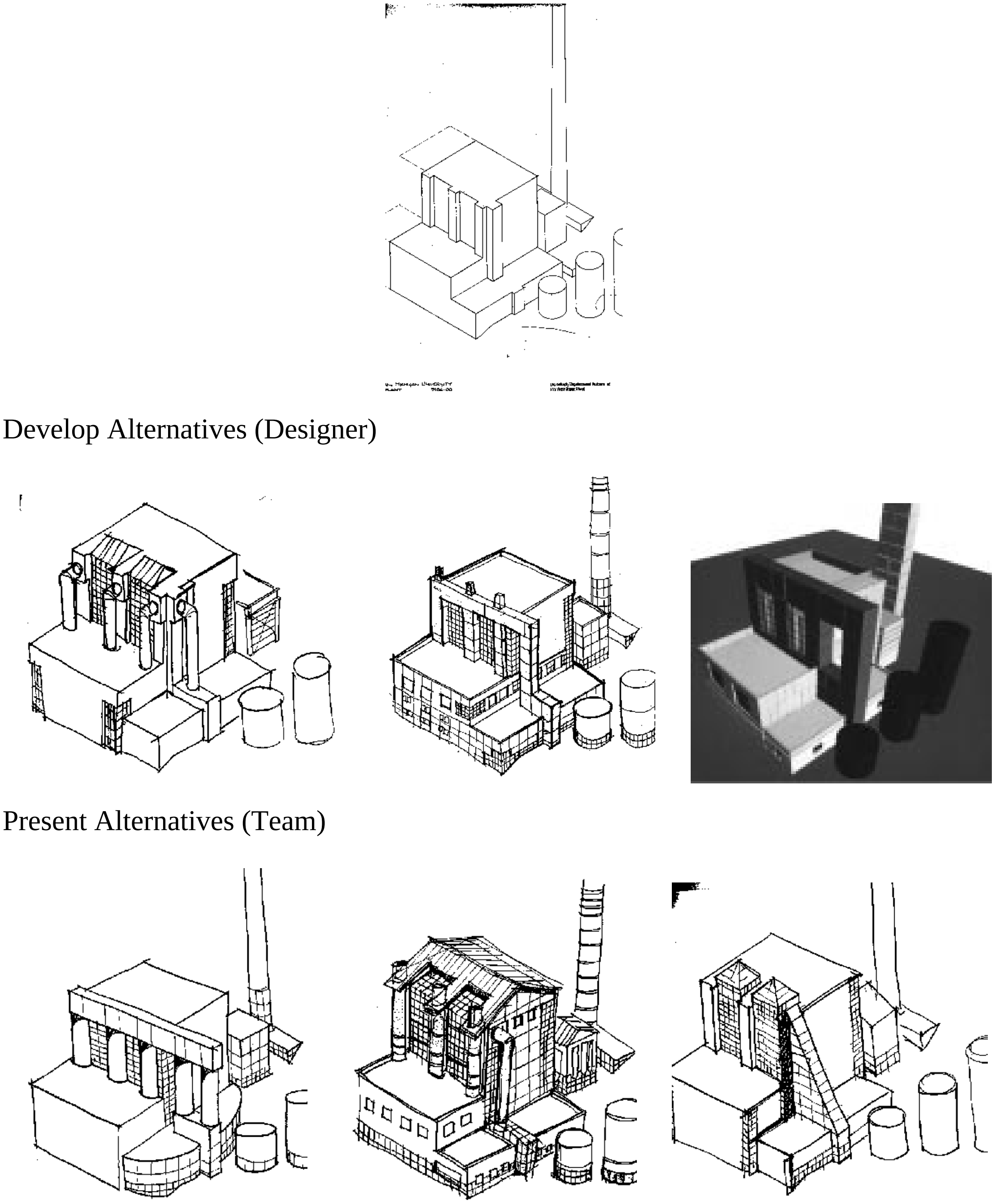


Figure 2: **Samples of Design Artifacts collected at Various Stages of the Process**

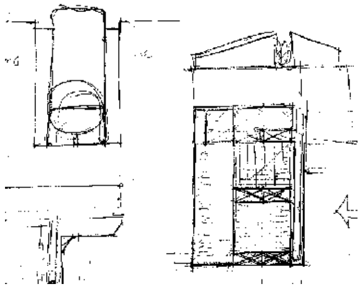


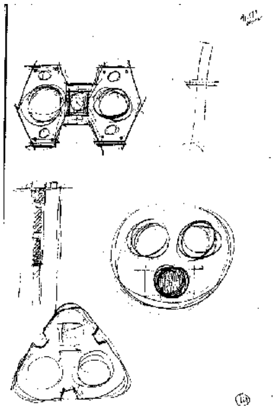
A. Transfer Drawings (Engineering Firm)

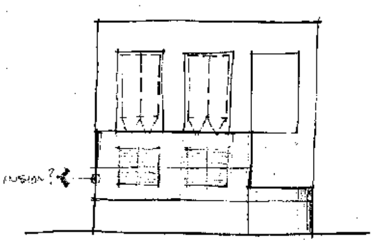
B. Develop Alternatives (Designer)

C. Present Alternatives (Team)

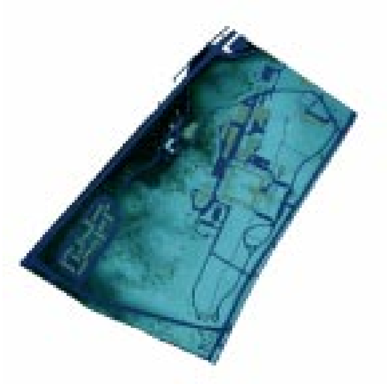
D. Discuss Alternatives (President & Team)

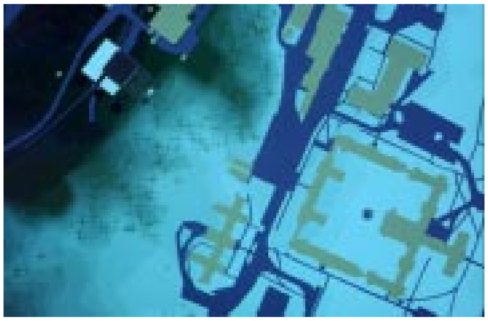




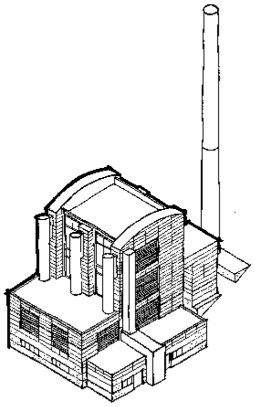


E. Develop Digital Terrain and CAD Model (CAD Consultant)



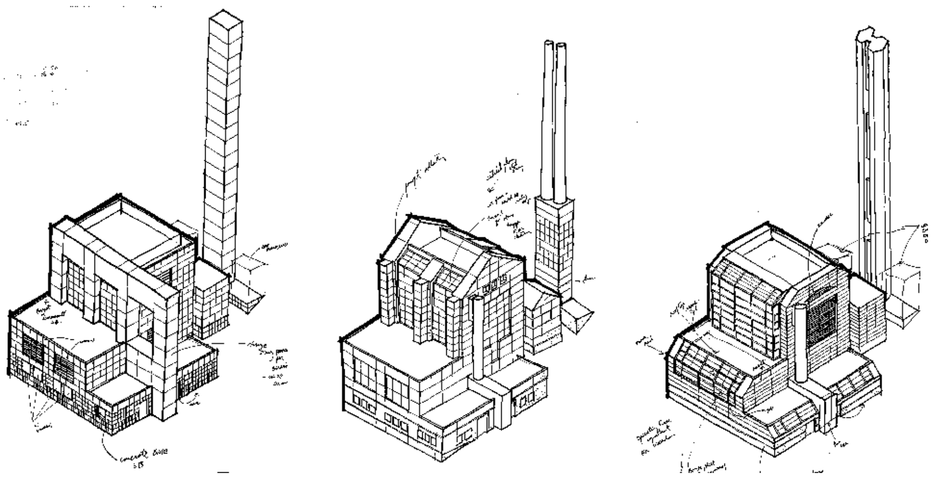


F. Develop Modifications (Designer)

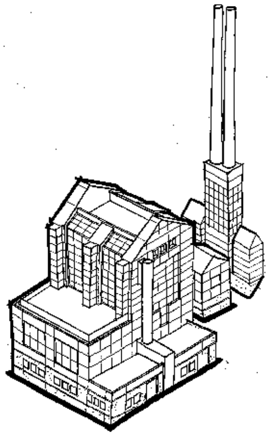


G. Develop CAD Model (CAD Consultant)

H. Discuss Modifications (Team)



I. Present Solution



2.1 Participants

In this case study, the participants consisted of an engineering firm and an architecture firm. In a reversal of typical design contract scenarios, the engineering firm subcontracted the project to the architecture firm in order to design the exterior shell. The participants within the architecture firm were: The firm’s president, the project architect, a designer, a design critic, and a CAD consultant. For the purpose of representing activities carried out by the architecture firm team as a whole, a column is added with the title "Team."

2.2 Chronology of Activities

The chronology of the first phase of the project was as follows:

- The engineering firm negotiates the contract with the architecture firm and transfers the project documents.
- The architecture firm president assembles a design team.
- The team meets and plans for creating alternatives.
- The individual members create alternatives.
- The design team presents, discusses, and modifies design solutions.
- A CAD consultant is hired to develop CAD models and integrate the project with a digital terrain model.
- The 3D CAD model is presented and modifications are requested.
- A preferred solution is pursued and submitted for further development.
- The developed solution is presented internally and to the client.
- Smaller modifications are requested and a second development cycle is executed

2.3 Categories of Activities

Several general categories of activities have been derived from the afore-mentioned conceptual model: (1) Request, (2) Work, (3) Plan, (4) Discuss, (5) Present, (7) Choose. These categories involve individual as well as collaborative work and can take place synchronously as well as asynchronously. For example, developing a solution can be an individual task while discussing the different alternatives is by nature a collaborative activity. Requesting information and its transfer can be carried out immediately (synchronously) or over a longer period of time (asynchronously). The scope of this paper does not permit the presentation of the full analysis carried out. However, based on this analysis, a protocol of interaction for supporting collaborative design was derived and is included below.

3 A PROTOCOL OF INTERACTION IN COLLABORATIVE DESIGN

A protocol of interaction maps and abstracts part of the activities performed by designers in the early stages of an architectural project. In that sense, a protocol of interaction can be thought of as a *model*. The usefulness of this model depends on its capacity to represent reality in a consistent manner thus enabling us to better understand it. Broadly, a model formalizes a set of declarative and procedural units that are usually mapped from their counterparts in reality. In the domain of interest addressed in this paper, a protocol of interaction is a model that formalizes a set of declarative and procedural units that govern the way in which a group of participants build design objects.

The units in the proposed model consist of two main entities: A Participant (PAR) and a Collaborative Design Object (CDO). PARs represent the human designers using the system to collaboratively design objects represented by CDOs. These objects are the focus of the collaborative activity and can have several sub-types: (1) Task, (2) Proposal, and (3) Artifact. Examples of attributes shared by Tasks, Proposals, and Artifacts are: (1) Identification, (2) Creator, (3) Date of Creation, (4) Date Last Modified, (5) Access Control List, (6) Location, (7) Status, (8) Version Number. Due to the object-oriented design of the protocol (Booch 1994), these attributes are defined at the CDO level and inherited by the Task, Proposal, and Artifact objects.

3.1 Explanation of the overall model

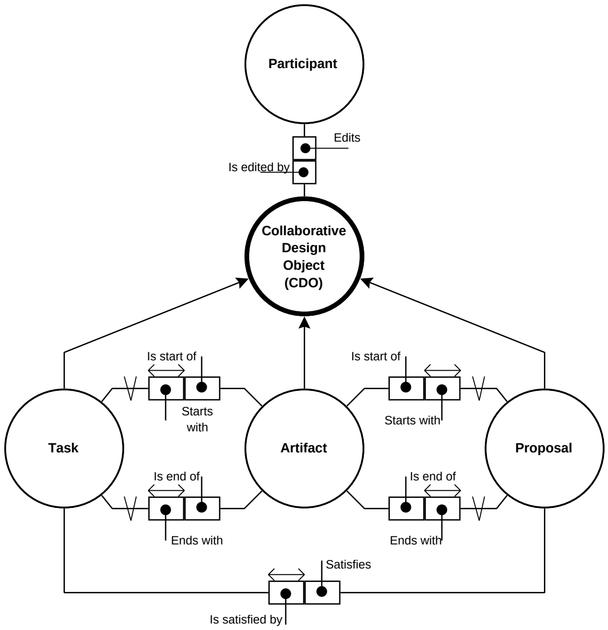
Figure 3 shows these representational units using the NIAM notation (Turner, 1989). The following sections include a description of the units and the relationships among them which are best understood when compared to their graphical representation in figure 3:

- One or many members of PAR, edit (i.e. create, delete, modify) one or many members of a CDO.
- A CDO can have other embedded CDOs in it, but cannot embed itself (Strict hierarchy).
- A CDO has three sub-types: Task, Artifact, Proposal. These subtypes inherit the attributes of a CDO such as the ability to embed other CDOs in themselves. Since PAR can edit a CDO, PAR can edit a CDO subtype.
- A Task must start with (i.e. take as input) one and only one Artifact.
- A Task must end with (i.e. produce) one and only one Artifact.
- A Task is satisfied with (i.e. can be solved by) zero, one or many Proposal.
- A Proposal must start with (i.e. take as input) one and only one Artifact.
- A Proposal must end with (i.e. produce) one and only one Artifact.

3.2 The Basic Mechanisms of the Model

The participants join a common session and start by viewing a shared window containing the status of the project at hand. Using this overall view of the project, participants can navigate the various nodes to explore, in more detail, the status of the various sub-nodes. Figure 4 illustrates a conceptual diagram of a possible flow of tasks (T), proposals (P) and artifacts (A) in an imaginary project. Participants can collaboratively create new task nodes and arrange them in a hierarchy. They can subdivide a task into several sub-tasks, attach competing proposals to a given task, and specify the related artifacts.

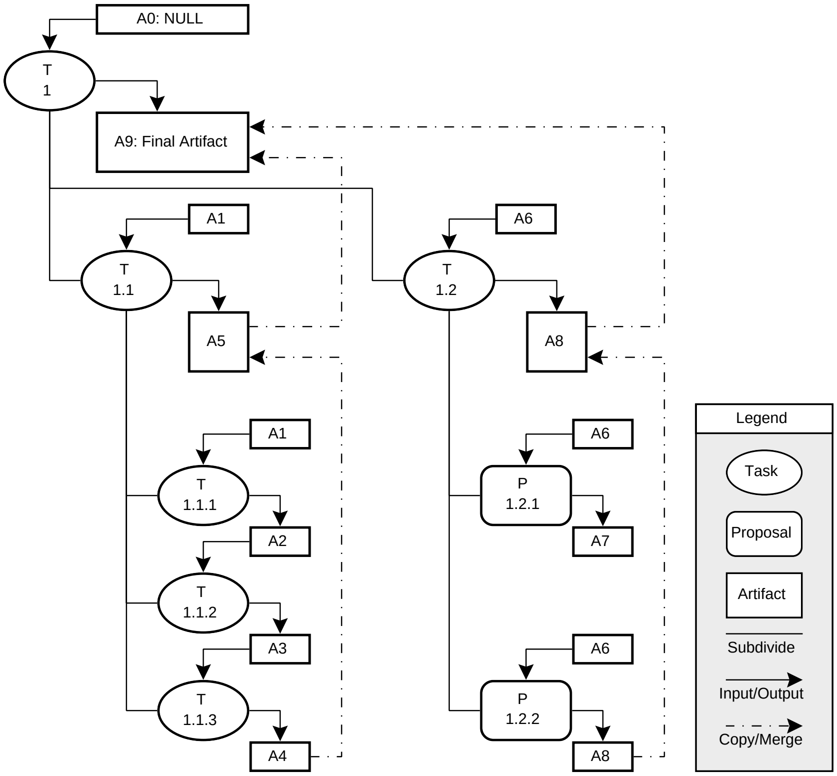
Figure 3: **Top-level NIAM Representation of Protocol Elements and Roles**



Each new task has an input artifact (e.g. A1, A5) and an output artifact (A5, A8). Tasks can be subdivided into sub-tasks (e.g. T1.1, T1.2) and the input artifact of the parent task can be passed along to the sub-task or proposal (e.g. A1, A6). Competing solutions are attached to tasks in the form of proposals (e.g. P1.2.1, P1.2.2). The output artifact of a proposal is considered acceptable when it becomes the output artifact of the parent task (e.g. A8 was chosen when evaluated against A7). Additionally, the output artifact of a task can result from a merger of artifacts from other sub-tasks or proposals. For example, the final artifact (A9) is the result of merging the output artifacts of sub-task 1.1 (A5) and sub-task 1.2 (A8).

Collaboration happens at all levels of the flow depicted in figure 4. At the level of task design, participants conduct a meeting to decide on what tasks need to be created and who should work on them. This is usually done synchronously. Asynchronous collaboration happens when an individual works on sub-tasks or creates proposals and attaches them to the parent task. Proposals can embed a series of artifacts that can be played in sequence (thus emulating design moves). On the other hand, a proposal can embed at minimum the output artifact that represents the outcome of the proposal. Synchronous collaboration as well as competition takes place when a task or a proposal is assigned to a group of participants. Additionally, synchronous collaboration and competition occurs when participants meet to evaluate and defend proposals, review the status of the project, and merge artifacts from sub-tasks and proposals into output artifacts of tasks. This merging process is left up to the participants in that they verbally discuss the proposals, manually copy and paste parts of proposals into the final artifact and edit that final artifact to make it coherent.. Participants can at all times observe the status and content of tasks, proposals, and artifacts.

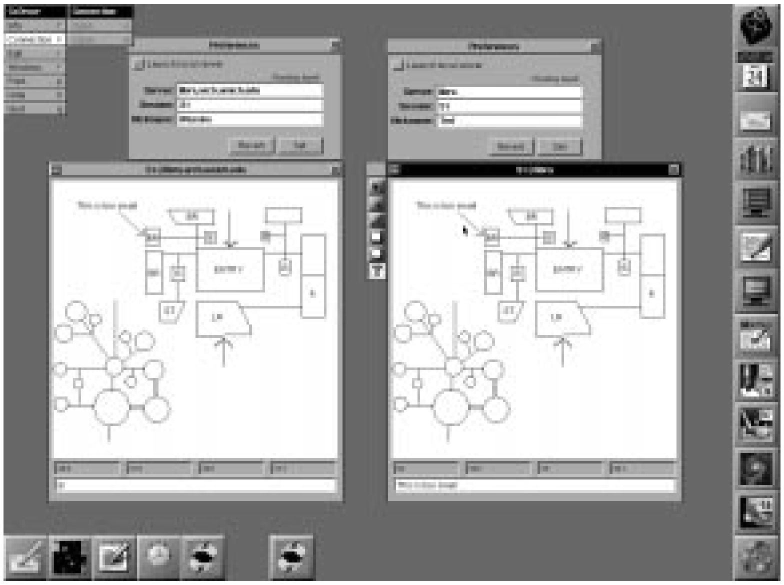
Figure 4: **An Example Flow of Tasks, Proposals, and Artifacts**



4 IMPLEMENTATION

The protocol as generally described above is currently under implementation. An early prototype of a Synchronous Collaborative Design Environment (SYCODE) (Jabi and Hall, 1996) has been implemented and tested between Ann Arbor, Michigan and Hong Kong (Figure 5). SYCODE enables geographically dispersed designers to share common representations even when using heterogeneous hardware. By using the Share-Kit collaborative toolkit (Edlich 1993), SYCODE is able to provide a *client-server* environment in which design participants use computer tools called *clients* to exchange information when they join the same session within a continuously running program called a *server*. Furthermore, the Share-Kit enables software developers to invent a protocol of interaction that includes a description of the data being shared and a specification of the methods that operate on that data as well as group interaction procedures. The first phase of this project included the installation of the Share-Kit servers at the two sites – Ann Arbor and Hong Kong – and the definition of an initial protocol for communication of shared artifacts. Two SYCODE client prototypes were then developed independently on different hardware platforms.

Figure 5: **Screen Shot of SYCODE Interface**



The SYCODE client starts by requesting the address of the machine where the server is running, the session name to connect to, and an optional alias for the participant to be identified with. Once a connection is made, the participant can create entities and add them to the shared workspace (Figure 5). These entities, however, are not merely graphical. They behave as complete objects that can be communicated with, asked to perform certain actions such as drawing themselves; and queried about ownership, status, assigned unique identifier and other task-specific attributes. The limitations of the network bandwidth as it exists now impelled us to devise a parsimonious representation scheme and to semantically define pending and confirmed actions.

5 CONCLUSIONS AND FUTURE DIRECTIONS

With the advent of globalization of architectural practice, there is a need to better support the world-wide activities of designers. In order to better serve these needs, domain-specific tools must be invented to augment and, in some cases, replace general-purpose tools. By analyzing observations of architects in practice, an identification was made of a need to invent a protocol that semantically and parsimoniously represent interactions in design. This protocol can enable the creation of domain-specific computer-supported collaborative design environments. To illustrate the feasibility of the above hypothesis, a computerized prototype was implemented that enables geographically dispersed designers to synchronously create and share design artifacts using efficient data structures and methods. The usefulness of the proposed protocol will only be verified with the construction and testing of more mature domain-specific tools that utilize it.

REFERENCES

Booch, G. (1994) *Object-Oriented Analysis and Design With Applications.* Benjamin/Cummings, Redwood City, California.

Edlich, S. (1993) Software Cooperation with the Share-Kit: Influences of Semantic Levels on the Working Efficiency, *Proceedings of the Vienna Conference on Human Computer Interaction*, Vienna, Austria, pp. 225-234.

Jabi, W. (1996) An Outline of the Requirements for a Computer-Supported Collaborative Design System, *Open House International* **21.3**.

Jabi, W. and T. Hall (1995) Beyond the Shared Whiteboard: Issues in Computer-Supported Collaborative Design, *Proceedings of the Sixth International Conference on Computer-Aided Architectural Design Futures*. Singapore, September 24-26 1995.

Turner, J. (1989) NIAM Modeling Guide, College of Architecture and Urban Planning, The University of Michigan, Ann Arbor.