SOME EXPERIMENTS ON IMPLEMENTING COLLABORATIVE BUILDING DESIGN ENVIRONMENTS

Y.Z. Chen, D. Robinson, I. Frame
Building Performance Research Unit
Anglia Polytechnic University
Victoria Road South
Chelmsford CM1 1LL, UK

T.W. Maver
ABACUS
University of Strathclyde
131 Rottenrow
Glasgow G4 0NG, UK

ABSTRACT

A collaborative building design environment has been proposed to integrate together both the heterogeneous applications and the dispersed project participants. Based on the functional requirements identified, the conventional building product models have been extended to incorporate high-level concepts such as activity and organisation, which are essential for coordination, and a generic human-human interaction model has been developed, which could not only make the building domain models interaction-aware, but also serve as a base model for developing general interaction utilities. Collaborative design environment prototyping has been described, covering the common project workspace, general interaction utilities and multi-user interfaces. Three distribution schemes for implementing the common project workspace within a distributed environment have also been discussed.

1. INTRODUCTION

Computer supported collaborative building design refers to people working together on the same project with the help from computers and networks to overcome time and geographical constraints. The development of such a collaborative design environment serves a dual purpose: from a pragmatic point of view, with the popularity of computer networks the organisational integration of bringing more closely together the geographically dispersed project participants is a natural extension to the current integrated building design systems; from a methodological perspective, building design has always been regarded as multi-disciplinary in nature and thus should be carried out as collaboratively as possible. With the experience gained from previous related research and development, the authors have chosen to develop a collaborative building design environment by focusing mainly on extending traditional integrated building design systems. The most related fields include concurrent engineering[1] and computer-supported cooperative work (CSCW)[2,3]. In the building design domain, some of the more closely related research efforts include synchronous cooperative architectural design[4] and the agent-based distributed building model[5].

The essence of such collaborative design environment lies in supporting interaction and coordination among relevant personnel through the mediation of computer artifacts. As in many other CSCW systems, the sharing of common information objects has been exploited as the fundamental means for facilitating collaborative activities. The presently described research efforts have been particularly focused on several key issues, which include representing the information for coordination support, modelling human-human interaction and designing distributed prototypes. Object-oriented technology has been adopted from analysis, through design to implementation.

2. FUNCTIONAL REQUIREMENTS FOR COLLABORATIVE BUILDING DESIGN ENVIRONMENT

Collaborative work centres around the coordination of the activities of multiple actors. Coordination exists because the activities are interdependent. These interdependencies are dependent on the domain and the specific situation. Apart from other factors good communication among the actors is essential for managing these dependencies. Information technology could be utilised to support coordination by helping represent and detect the dependencies and facilitate better communication and interaction. Specifically, the authors suggest that computer-mediated collaborative design may be facilitated by integrating some generic interaction mechanisms with the coordination supporting information regarding the building design domain and the personnel[6]. Instead of prescribing specific coordination procedures for the collaborators to comply with, the adopted strategy emphasises supporting the actors to become more aware of the interaction and collaboration by exposing the collaboration-relevant information and providing general interaction facilities within an integrated environment. Such an environment is a facilitating medium which provides the basis for a high degree of synergism, such that the entire design process become more productive, effective and efficient. The

following key functional requirements have been identified for a collaborative building design environment.

The requirements raised for supporting generic interaction include:

- **Interaction-awareness support** Two types of interaction-awareness amongst partners have been identified -- knowing the common information objects which are being shared and being directly aware of the presence of the partners. Both should be supported, ideally in an integrated way. Furthermore, mechanisms need to be provided to distinguish and switch between private and public visibility on both data and action.
- **Supporting different interaction modes** As in the real world, computer mediated human-human interaction could take place in different modes, which could be synchronous and/or asynchronous, within local area networks and/or wide area networks, explicit(direct) and/or implicit(indirect), formal and/or informal.
- **Multi-media interaction support** Multi-media communication is very important in real world situations, as it is in the computer-mediated world. This is particularly true for the collaborative building design environment, because the domain is still difficult to be fully formalised, and thus informal representation in multi-media could greatly expand the computer's expressiveness.

Requirements raised from the building design domain include:

- **High-level information for coordination** Support is needed for the representation of organisations (such as teams, groups etc.) and activities, in addition to product modelling. The involvement of heterogeneous participants on a building project demands for some form of management at a level higher than product data for its own sake. More importantly, high level information such as the dependencies amongst activities is required as a resource for conflict detection and resolution.
- **Tool control** Apart from data management support, the involvement of large numbers of CAD tools to be used by heterogeneous collaborative users demands much more flexibility and ease of use than single user systems.
- **Support for *data locality* and *tool locality*** Collaboration in building design tends to be in coarse granularity and asynchronous, because finer task decomposition may not be feasible due to the unstructured nature of the domain. This implies that certain items of data and the corresponding tools might be monopolized by an individual user for a particular period of time. Therefore, these data items and tools should be located in close proximity to this individual at least for that period of time.

3. CONCEPTUAL MODELLING FOR COLLABORATIVE BUILDING DESIGN

Based on the functional requirements discussed in Section 2 a set of conceptual information models have been developed, as depicted in Figures 1~3, by using object-oriented methodology. The notation used is based on the Coad-Yourdon approach[7]. For simplicity, the *attributes* and *services* components of the entity box are omitted for most entities within the diagrams, with only the entity name shown in the entity box. (Throughout this paper, *entity* is used to refer to either *class* or *object*, and *type* and *class* are used interchangeably.) As will be demonstrated in the following sections, the extensibility offered by object oriented technology (notably through composing new aggregate types, deriving new types and adding new members to existing entities by introducing base classes) has greatly helped in extending the traditional building product model into an integrated collaborative building design model.

3.1. Building product model

The building product model in Figure 1 is one of those typical building models which could be found on traditional integrated building design systems (IBDS). In fact this very model was originally developed in a previous IBDS project[8], with the aim of integrating a range of CAD and building performance simulation and analysis tools for the early phases of building design. This model has been chosen as the starting point for the developing of a collaboration-aware information model.

3.2. Organization and activity models

Firstly the organization and activity models have been developed, as illustrated in Figure 2, based on the concepts of group process theories[9,10]. *Organisation* is modelled as a hierarchy of *Person* and subordinate organisations. Both the organisation and the person may be associated with one or more *Roles*. The activity model centres around the *Activity* entity, which is again defined as hierarchical to model subtasking. A collection of activities form the task *Agenda*. The *Resource* entity specifies the resource requirements such as particular CAD tools and hardware devices. Each of the activities and the tools would be related to a particular set of information data, which forms an *Area* of interest. Some of the activities may be interdependent of each other; this is modelled by the

Dependency entity, which can be used to specify pre-conditions and post-actions in terms of constraints and rules respectively. This dependency information could be utilised by the computer to monitor and automate part of the coordination process.

The three models -- product, activity and organisation -- are linked together through some interfacing entities. The *Role* links organisation model and activity model together. Compared to otherwise directly associating person/organisation to task, the use of such an interfacing entity provides more flexibility. The product and activity models are linked together by the *Area* entity, which is actually an aggregate composed of entities from the product model. At present the specification for the *Area* entities is based on the data requirements of the tools, which forms different aspect models. Note that rather than looking for a common mechanism for representing products, activities, roles and organisations, the aim of these simple models is to allow the dependencies between different activities and roles to be represented within the environment, such that the potential offered by the advanced information technology on supporting interaction and coordination could be explored.

3.3. Human-human interaction model

So far, the organisation, activity and product models together provide the primary means for representing the structural relationships amongst actors, roles and tasks, which is the pre-requisite for coordination and interaction amongst the collaborators. In this section a generic interaction model is described, as illustrated in Figure 3, which could be used to augment the above static information models into active, collaboration-aware ones. Additionally, it could be used to help develop general interaction utilities such as electronic meeting and multi-media communication facilities, which are complementary to the building domain-specific functions in such a collaborative design environment.

In this model, the *HHInteraction* entity models the human-human interaction as the sharing of information objects *IObject* among interacting parties (i.e., *Initiator* and *Partner*, derived from the *Person* type defined in the organisation model) in a particular interaction mode defined by *Modes*. The action leading to this sharing could be "read", "write", "create" or "send" etc. The *IObject* here is the most important entity, which acts as the interface between the *HHInteraction* entity and those which want to become interaction-aware. Apart from the association link with *HHInteraction*, *IObject* is also associated with *IHistory* and *Medium*. *IHistory* is a persistent entity, storing the collection of the past interactions associated with this object, and thus serves as a better alternative to the versioning mechanism in the conventional object-oriented database management systems (OODBMSs) to support design decision-making tracibility. In addition to the default binary format, interaction in multi-media such as plain text, PostScript file and various image formats could be supported through the *Medium* entity.

The *IObject* is a multi-media, interaction-aware entity; it tracks the current interaction status as well as the past interaction history. Through this *IObject* collaborators can become aware of the presence of each other. The *IObject* entity is defined as an abstract class, and serves as a *base* class or type to the whole system. By inheriting from this base class, any building domain-specific entity may become fully collaboration-aware. Also, general interaction utilities could be developed from scratch by deriving from this abstract class.

Note that this interaction model specifies human-human interaction in a generic way. The semantics of the interaction would mostly be determined by the specific object, itself inherited from *IObject*. For instance, when the specific object is within the organisation or activity models, the resultant interaction may be taken as a coordination-related activity. Also, a private action may be taken as a trivial interaction with zero partner.

4. COLLABORATIVE BUILDING DESIGN ENVIRONMENT PROTOTYPING

The prototype of the collaborative building design environment includes a common project workspace, general interaction utilities and multi-user interfaces, as depicted in Figure 4.

4.1. Project workspace

The project workspace is a building design domain component, serving as a common collaboration environment for the project team. It includes a project database and CAD tools. The database is an integrated implementation of the information models described in section 3, using an OODBMS called ODE[11,12]. The concurrent access by multiple users to the shared objects is managed at two levels. At the low mechanistic level, the locking mechanism provided by ODE allows multiple transactions to read and one transaction to write an object simultaneously. Also, "hypothetical transactions" are supported with ODE to allow users to test "what-if" scenarios, which is very important for supporting collaborative design activities. At the high level additional support has been implemented to help maintain the semantic integrity of the shared database, which includes the facilities for managing the activity interdependencies and user interactions,

based on the information from entities such as *Dependency* and *IHistory* etc. For example, messages could be automatically generated and sent to the affected users after each update transaction according to the activity dependency information and the interaction status. More delicate coordination for the shared access of objects mainly relies on individuals' efforts, which could be guided by collaboration-related information such as activity dependencies and interaction status. Because each of the objects inherits from the interaction-aware entity *IObject*, the collaboration-related information would be immediately available to the users whenever an attempt is made to access an object. The "tool pool" in the workspace contains tool objects. Each of the tool objects corresponds to a CAD tool, which is encapsulated with additional information such as availability status, invocation requirements, format for data input and output. A user account mechanism, based on the UNIX multi-user operating systems, is designed as the basic means for providing multi-user access to the workspace and other utilities.

In a distributed environment any of the system components may be implemented in either a centralised or a replicated way, providing that it is shared by multiple dispersed users. In a highly centralised or classic client-server implementation, a central server program handles all user input and display events, which are routed through local client programs, while local workstations act as graphical terminals and window servers. This scheme has the advantage of simplicity, but at the cost of large volumes of communication traffic and slow feedback. At the other extreme, replicated implementations maintain copies or replicas of the application on each workstation. Each replica handles access management locally and broadcasts any change in application data to all other replicas for update to maintain consistency. A replicated implementation provides fast feedback while suffering from complicated synchronisation amongst the replicas to maintain consistency. A hybrid solution seems possible, in which components of the cooperative system are either centralised or replicated, depending on the application requirements. The authors have been experimenting with several schemes for implementing the common workspace, three of which are illustrated in Figure 3. Apart from the usual trade-offs, more domain-specific advantages and disadvantages are revealed.

Scheme (a), the centralised (classic client-server) model, is a straightforward extension of the traditional integrated design system with interaction mechanisms augmented to allow multiple users to share data and tools by means of TCP/IP streams. This is the cheapest scheme from the perspective of the client workstations, and a good way to share expensive packages. But the server could be overloaded, and data and tool localities are not supported.

Scheme (b), the augmented client-server model, is an extension of (scheme (a)) by adding a data cache mechanism and local tools on the client workstations. This is a partial replication scheme, with only those data within the area of interest replicated. With such augmentations the user could choose to carry out some of the tasks locally by downloading the relevant data from server to the local data cache, running local tools and finally uploading results back to the server. Localities are catered for with the cost of increased complexity for replication and synchronisation. It is also expensive in terms of resources.

Scheme (c), the distributed model which has evolved from (scheme (b)), takes a more extreme step towards localisation. Either part or the whole of the project workspace can be distributed and permanently residing on client workstations or any other sites. Information data and tools are both modelled as autonomous objects providing services, which may be requested by other objects. Instead of keeping replicas of the distributed objects, the central workspace keeps pointers to the actual object implementations through a directory mechanism, which results in a *symbolic* project workspace. Each object implementation could actually be an autonomous database or tool server. This approach represents a very flexible way for the modelling of complicated heterogeneous distributed systems and is particularly powerful for integrating together naturally distributed sub-systems and organisations. Each of the previous schemes could be perceived as a special case of this model. Through creating and deleting object implementations preferred set-ups could be configured dynamically to suit changing needs with the evolution of the project. Also advantageous with this approach is its consistency with the international standardisation on developing interoperable object-oriented systems. In fact, a CORBA (Common Object Request Broker Architecture)[13] package called ORBeline[14] has been deployed to implement the distributed object management system.

4.2. General interaction utilities

Interaction through sharing the common workspace tends to be formal, asynchronous and building domain-specific. To accommodate a variety of interaction patterns as listed in Section 2, a set of general interaction utilities have been designed. Combining the both together, an integrated collaborative working environment would yield.

A messaging system, with the conventional e-mail integrated in, is designed to support asynchronous, explicit interaction. Real-time synchronous interaction support includes electronic meeting and conversation facilities. Multi-media interaction support includes image-based communication facilities, which could help generate, transmit (in either real-time or asynchronous mode) and visualise images in several formats.

4.3. Multi-user interface

At present, only a text-based user interface is in operation, but graphics-based CAD tools on the remote workspace can be invoked and displayed on the local client workstation through X window mechanisms. A X11-based WYSIWIS (what-you-see-is-what-I-see) multi-user interface has been under development, which includes a project database area, a resource area, a monitoring area, and a working area. Image icons are used to symbolise the data objects, tools and participants present in the common workspace. Menu bars are designed, which classify the user commands into different categories. Like the common workspace, the multi-user interface could be implemented in either a centralised or a replicated way in a distributed environment. For simplicity a centralised approach has been adopted for implementing the WYSIWIS multi-user interface.

5. CONCLUSIONS

This paper has described a research project involving the development of an integrated collaborative design environment. The aim of this environment is to achieve a high degree of human synergism by making use of high level information such as organisations and activities within an interaction context. Object-oriented technology has been employed throughout the system development process to increase flexibility and extensibility and to assist incremental prototyping.

The prototype has been developed with the construction industry as the focal point, but it is recognised that this could be extended in the future to other sectors of commerce and industry.

6. REFERENCES

1. Y.V.R. Reddy, K. Srinivas et al (Eds), Special issue on "Computer Support for Concurrent Engineering", IEEE Computer, January 1993.

2. K. Schmidt and L. Bannon, "Taking CSCW Seriously. Supporting Articulation Work", Computer Supported Cooperative Work, Vol. 1, Nos. 1-2, pp. 7-40, 1992.

3. IEEE Computer, Special issue on Computer-Supported Cooperative Work, May 1994.

4. M.L. Maher, J.S. Gero and M. Saad, "Synchronous Support and Emergence in Collaborative CAAD", In: U. Flemming and U.V. Wyk (Eds), CAAD Futures '93, North-Holland, 1993.

5. R.R. Bhat, J. Gauchel and S. van Wyk (1993), "Communication in Cooperative Building Design", In: U. Flemming and U.V. Wyk (Eds), CAAD Futures '93, North-Holland, 1993.

6. Y.Z. Chen, I. Frame, and T.W. Maver, "The architecture of a computer-mediated collaborative product development environment", Intelligent Manufacturing '95, SPIE Proceedings Series, SPIE Press, Bellingham, WA, USA, (In press).

7. P. Coad and E. Yourdon, Object-Oriented Analysis (2nd edition), Yourdon Press, NJ, 1991.

8. T.W. Maver, J.H. Rutherford and Y.Z. Chen, Final report for SERC project: "Knowledge-based design decision support (KNODES)", University of Strathclyde, 1993.

9. S. A. Olsen (Ed), Group Planning and Problem-solving Methods in Engineering Management, John Wiley& Sons, 1982

10. R. S. Baron et al, Group Process, Group Decision, Group Action, Open University Press, UK, 1992.

11. S. Dar N.H. Gehani and H.V. Jagadish (Eds), ODE: Object Database & Environment, AT&T Bell Laboratories, Murry Hill, NJ, USA, 1992.

12. R. Arlein, A. Biliris et al, Ode 3.0.3 User Manual, AT&T Bell Laboratories, Murry Hill, NJ, USA, 1994.

13. OMG, The Common Object Request Broker: Architecture and Specification, OMG Document Number 91.12.1, (Revision 1.1), December 1991.

14. PostModern Computing Technologies, ORBeline Reference Manual, CA, USA, 1994.

7. FIGURES

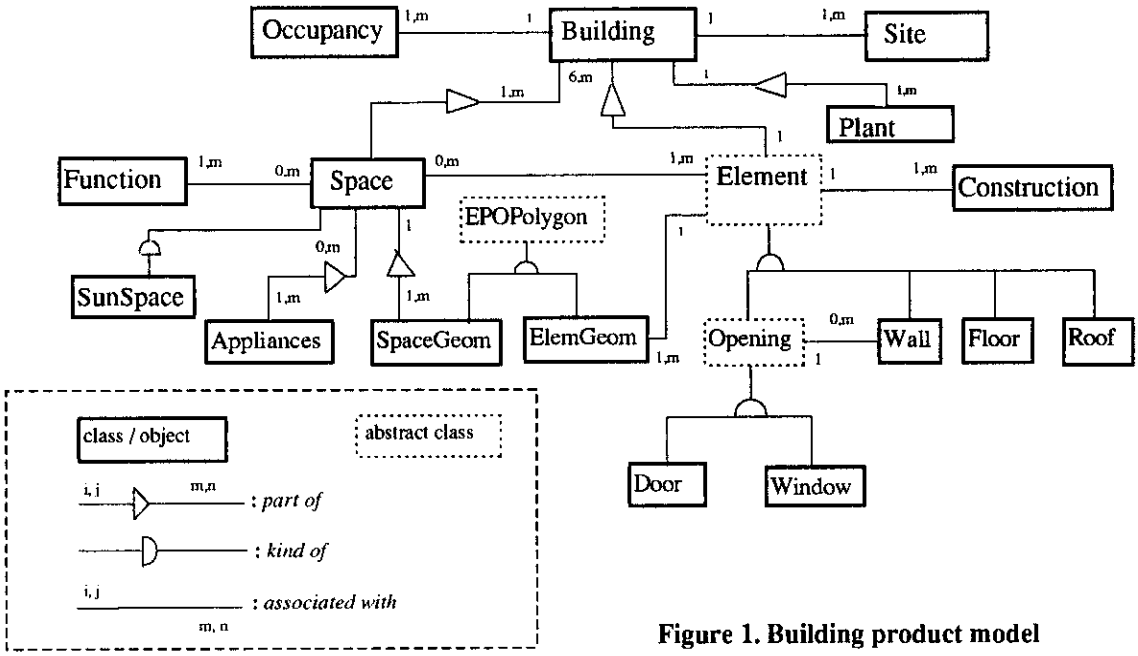


Figure 1. Building product model

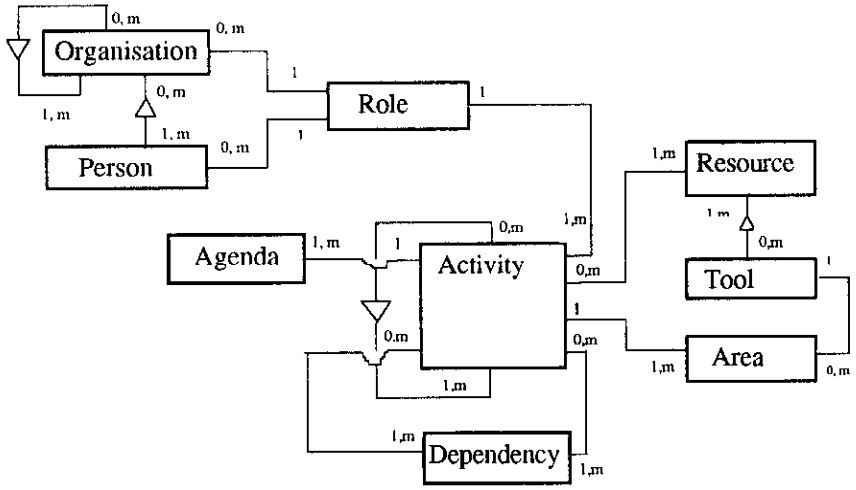


Figure 2. Organisation and activity models

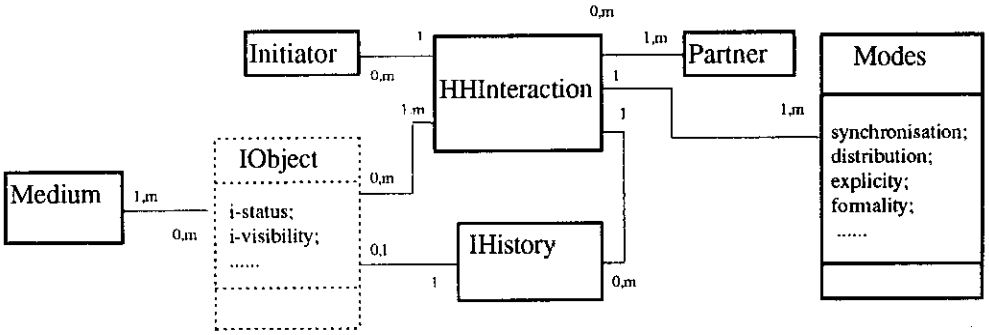
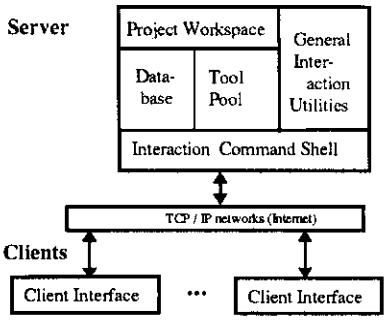
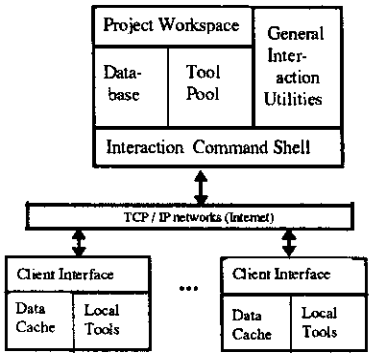


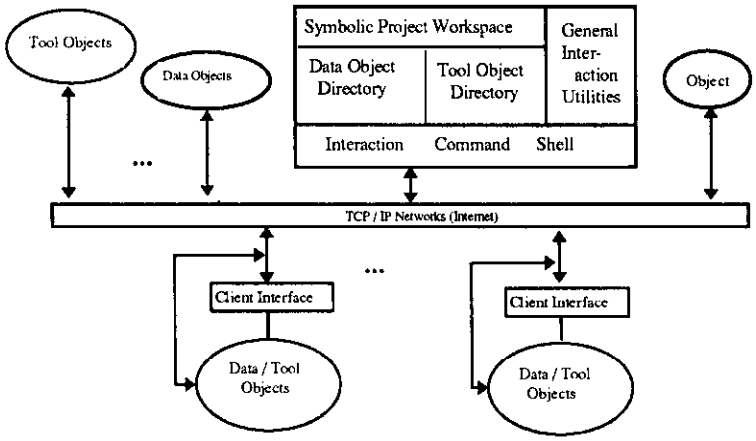
Figure 3. Interaction model



(a). Centralised model



(b). Augmented client-serve model



(c). Distributed model

Figure 4. Implementation models