Supporting heterogeneous distributed substantiation of common generic structures in collaborative design

C. Peng
EdCAAD
University of Edinburgh
20 Chambers Street
Edinburgh EH1 1JZ
United Kingdom

Abstract

This paper presents an analysis of collaborative design activity, explaining how design participants collaborate to achieve integrated design on the basis of sharing and substantiating common generic structures with domain design developments. Referring to a previous observation of teamwork in architectural modelling, an overview of the *structuralist* approach to collaborative design is firstly introduced. The structuralist scenario is then classified into the aspects of model construction, model-constructing constraints, and modelling acts. By examining the properties of different types of design representations and the systematic relations among them, the *constraints on collaboration* are identified; a logic of collaborative design is found in the necessity of maintaining a dual correspondence between the evolution of *common generic structures* and the development of *domain design solutions* distributed over several sites. Following the constraints derived, a discussion of the basic requirements for computer-based tools to support collaborative design is given. The paper concludes with how the current work can be related to the research carried out in other areas.

Keywords: collaborative design, common generic structures, heterogeneity, distributed substantiation, computer support requirements.

1 Introduction

Research on designing computer-based tools to support group design activity has recently become active, attracting the attention of researchers working in various fields. Unlike conventional computer-aided design tools, the new tools are expected to support *collaboration* among designers participating in project work. This paper attempts to establish a conceptual framework for

*Current address: Information Technology Research Institute, University of Brighton, Lewes Road, Brighton BN2 4AT, U.K.

the development of a computer-based modelling environment that can facilitate collaborative design activity.

Initial solutions to the design of communicating and computing tools that can be supportive for people involved in collaborative design can be seen in the field of Computer-Supported Cooperative Work (CSCW). In a recent bibliographical survey of the research in CSCW, Greenberg [Greenberg, 1991] has introduced the key phrase "shared workspace". It is noticeable that a large portion of the experimentation on shared workspaces has to do with building prototypes of "shared drawing space". These group drawing tools were based on earlier observational studies of working group graphics and shared drawing space activities (see, for instance, [Lakin, 1983] [Bly, 1988] [Tang & Leifer, 1988]).

However, it is evident that, to different CSCW research groups and system developers, 'design' has different meanings. As revealed in the published prototypes, design activities have been described and analyzed according to various research interests and perspectives. Due to the differences in studying group work in design, there appears diversity in understanding of what constitutes a workspace for collaborative design; and the different understandings of 'how drawings and drawing activity are related to design' have resulted in varied system solutions to 'what is to be facilitated by a shared drawing tool'.

Our current enquiry into computer-supported collaborative design has an emphasis on *design thinking* in a teamwork context. For this purpose, it is considered that a move into the study of *design modelling activity* can open up a more appropriate research vantage. Seen from design modelling, a designer not only performs design actions but also considers *design representation* which is the designer's concerning with the ways of generating and modifying design results. We believe that an understanding of teamwork in design modelling can contribute to a unified framework for eliciting and organising both representational and communication requirements in collaborative design. In particular, we are interested in giving a structural account of the collaborative design activity that has the following features:

- *Integration* — The teamwork has a general goal of achieving single integrated designs that satisfy all participating design views;
- *Distribution* — The teamwork is participated by multiple individuals who hold different design perspectives and work in heterogeneous design worlds;
- *Evolution* — The teamwork is situated in the interplay between *integration* and *distribution* (i.e., no fixed routes of integration or of distribution are set in advance).

In an earlier case study of participatory architectural modelling, reported in [Peng, 1992c] [Peng, 1992b], a distinct teamwork approach to design modelling is characterised as "structuralist[1]".

[1] "Structuralist" is the term coined by the author to characterise the observed pattern of teamwork in archi-

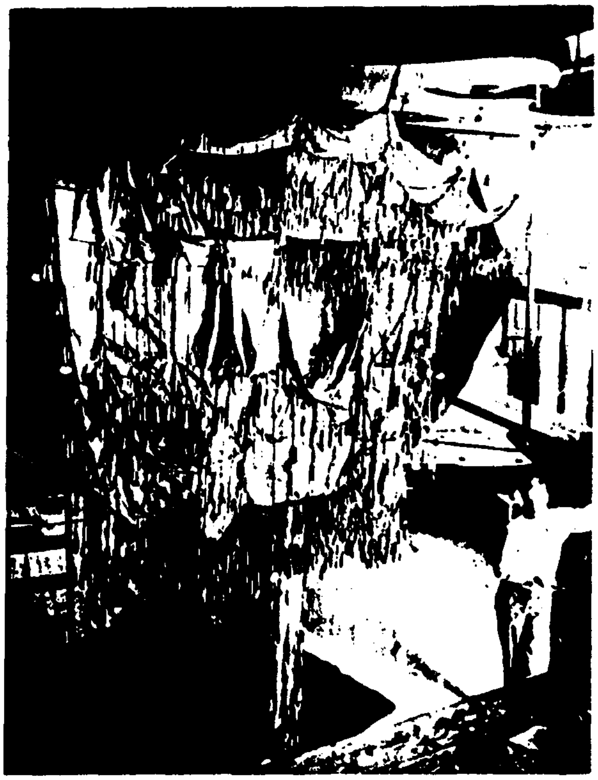

Figure 1: The construction and sharing of the funicular structure in the Colonia Güell church design project.

Our abstraction of the structuralist approach is mainly based on a study of the funicular model constructed in the Colonia Güell church design project [Martinell, 1979, p.335] [Collins & Nonell, 1983, pp.31- 35] (See Figure 1). This approach presents an interesting feature of group design: some *common generic structures* are created collectively but viewed and substantiated differently by members of a design team.

The remainder of the paper is organised as follows. An overview of the structuralist approach is introduced in the next section. Given the structuralist scenario, Section 3 presents a classificatory scheme, individuating the basic constituents of the collaborative design activity. Within the framework, the constraints on collaboration are derived in Section 4. Following the constraints presentation, an informal specification of the requirements for computer support is presented in Section 5. The paper concludes with a discussion of related research and a direction to further work.

tectural design. As it will be shown later, the term is purely used to indicate the construction of "structural objects" in group modelling processes. It is not intended to relate to other notions of "Structuralism" developed by, e.g., Levi Strauss, Derrida or others.

2 The Structuralist Approach: An Abstract

By referring to our previous observation of the funicular modelling case, the structuralist approach to teamwork modelling can be described concisely as follows:

> At the inception of a design project, members of a design team work jointly in constructing a single spatial framework or skeleton as a *Common Generic Structure* in a *Group Modelling Space*. When applying projective devices onto a state of a common structure, *Derivative Structures* can be produced and then imported to *Individual Modelling Spaces* that are set up and used by different participating designers.
>
> By taking derivative structures as design referents, participants carry out separate strands of *Domain Design Developments* by substantiating (parts of) the generic structure into specific design expressions in individual modelling spaces. In the course of elaborating design developments, however, any participants may work up to the need to change parts of the derivative structures in use; to fulfill the intended changes, the individuals manipulate and modify parts of the common structure. The changes thus proposed by one individual can subsequently cause further changes to be made in the derivative structures used by other participants.

3 A Classification of the Structuralist Scenario

The special terms introduced in the above abstraction denote some of the primary concepts of the structuralist approach to collaborative design. In this approach, team members are enabled to coordinate domain design developments through the sharing of the generic structures constructed by themselves. The concepts arose here are important in soliciting the requirements for representing and communicating design intents in collaborative design. To analyse the scenario further, a more elaborate account of the concepts abstracted above is given below.

3.1 Common Generic Structures

Common Generic Structures ($\mathcal{CGS}$) are 2-D or 3-D generic objects, representing kinds of spatial frameworks or skeletons that are constructed and used by all participants working in different aspects of a design project. When created and evolved in a group modelling space (see below), parts of a common structure can be manipulated directly by participants working in different design domains. There are the following general properties of $\mathcal{CGS}$:

- *Deformability.* $\mathcal{CGS}$ are made as instances of model constructs that are connected in a field of physical forces or formal constraints. Being constructed and shared by all participants,

$\mathcal{CGS}$ are meaningful and useful in revealing certain spatial forms or geometrical shapes. Changes in forces/constraints applied or in values of constructs may *deform* $\mathcal{CGS}$ into different states. The deformability of $\mathcal{CGS}$ entails that the construction of the generic objects is based on certain *physical* or *formal models* which behave in certain systems of constraint satisfaction or equilibrium of forces.

- *Multiple-viewpoint.* Parts of $\mathcal{CGS}$ can be manipulated by participants from multiple points of view for different design reasons. The multiplicity is firstly achieved by participants' introducing types of model constructs that correspond to various "perspectives" of design modelling (e.g. site, structure, enclosure, opening etc, in building design). Secondly, there are multiple ways allowed to assemble or detach model constructs while modifying parts of $\mathcal{CGS}$. This multiplicity lies in a range of *connecting devices* that can be used to associate various types of model constructs introduced in the first place.

- *Derivability.* A state of $\mathcal{CGS}$ can be applied with *projective devices* as intended by any individual designers. Valid projections of $\mathcal{CGS}$ can generate instances of derivative structures (see below) that can be further transported to individual workspaces for domain uses. The derivability of $\mathcal{CGS}$ can allow participants to establish *referencing* relations between individual design developments and the shared generic structures.

3.2 Group Modelling Space

The term *Group Modelling Space* (GMS) refers to a modelling space[2] created by designers participating in project work. One of the key functions of a GMS is its use by all participants in modelling $\mathcal{CGS}$. The basic constitution of a GMS may include the following components:

- *Model constructs* — a collection of elementary objects that can be instantiated[3] to represent what participants think of corresponding elements in the the real world.

- *Form-giving forces* or *constraints* — fields of physical forces or formal constraints that participants choose to shape or deform parts of $\mathcal{CGS}$. In designing buildings, examples of form- giving physical forces are gravity, acoustics, light etc, and certain spatial/shape grammars or schemas can act as systems of formal constraints. Given a constraining field in a GMS, all participants are concerned with if a state of $\mathcal{CGS}$, as manifested in a configuration of model constructs, is *equilibrium* or *satisfactory* to the forces or constraints applied.

[2] In its simplest constitution, we may think of a modelling space as a collection of model constructs and modelling operations that can be used to generate and modify design expressions.

[3] By "instantiation", we mean the assigning of particular values (e.g. values of length, weight, spatial position etc.) to types of objects.

- *Manipulative operations* — operations that enable participants to displace, transpose, or aggregate etc. instances of various types of model constructs such that common structures can be created and evolved.

- *Projective operations* — operations that allow participants to perform certain spatial actions, such as sectioning, projecting, tracing, truncating, developing etc., so that "secondary" structures can be deduced.

3.3 Derivative Structures

Derivative Structures ($\mathcal{DS}$) are 2-D or 3-D pictorial objects representing kinds of spatial frames or skeletons. Images of $\mathcal{DS}$ are produced by participants' applying projective devices onto a state of $\mathcal{CGS}$. Once imported into individual workspaces, instances of $\mathcal{DS}$ can serve the individuals as referents in modelling domain design developments.

In the course of developing domain designs, there may be a need to manipulate the underlying referents. The manipulation may be direct or indirect. In the case of indirect manipulation of $\mathcal{DS}$, the imported referents are "frozen" images, and they cannot be manipulated in parts but only as a whole. To effect changes in the referents, designers make changes in $\mathcal{CGS}$ and then re-derive $\mathcal{DS}$ containing the intended changes.

In the case of direct manipulation of $\mathcal{DS}$, participants may have $\mathcal{DS}$ *re-interpreted* so that domain-specific transformation can be defined and operated with. However, in maintaining consistency in both cases, changes in $\mathcal{DS}$ should trigger corresponding changes to be reflected in $\mathcal{CGS}$.

3.4 Individual Modelling Spaces

Individual Modelling Spaces (IMSs) are modelling spaces created and used by individual participants for the development of domain design solutions. In general, designers set up IMSs to include the following components in dealing with individual design tasks:

- *Individual design world* — a designer's design resource, consisting of (domain-specific) notations and tools for coding, visualising, manipulating and evaluating design expressions.

- *Derivative structures base* — an information space for storing, sorting, and displaying the images derived from $\mathcal{CGS}$.

- *Communicating tools* — communication channels for receiving and issuing confirmations or disagreements of making changes in $\mathcal{CGS}$, as manifested in imported states of $\mathcal{DS}$, among team members.[4]

[4] It is supposed that individuals working in different design aspects may be located remotely from each other,

3.5 Domain Design Developments

As described, the structuralist approach to collaborative design starts with the the construction of shared generic structures. However, final design products often go well beyond the design of general structural skeletons. An equally important part of collaboration is concerned with how the generic structures can be substantiated with more specific substances or properties. In this respect, domain design developments are the specialisation processes that are normally carried out by multiple parties with different design expertises. Typically, there are the following modelling acts involved in developing domain design solutions:

- *Constructing domain expressions* — In their own ways, individual designers are contributing different aspects of design solutions that can be attributed to particular parts or layers of the common structures. An important common factor is that all domain designs are developed on the basis of $\mathcal{DS}$. That is, designers construct Domain Design Expressions ($\mathcal{DDE}$) by taking $\mathcal{DS}$ of interest as underlying design referents.[5]

- *Reviewing design consequences* — Since each $\mathcal{DDE}$ is developed in relation to what $\mathcal{DS}$ is underlaid, and any instance of $\mathcal{DS}$ is a projection of $\mathcal{CGS}$, the resultant $\mathcal{DDE}$, as viewed and judged by its author from a particular design perspective, is the *consequence* of $\mathcal{CGS}$.

- *Evolving shared* $\mathcal{CGS}$ — When an individual develops domain designs to a certain extent, on seeing the resultant $\mathcal{DDE}$, he or she may conclude that the underlying design referents are not satisfactory. To explore other possibilities, the individual searches for modifications in $\mathcal{DS}$. These intended changes will further affect the state of $\mathcal{CGS}$.[6] In this respect, participants' developing domain designs in distributed IMSs may actually contribute to the evolution of $\mathcal{CGS}$.

3.6 A Classificatory Scheme

In giving a more elaborate account for the structuralist concepts of collaborative design, we have started with an explanation of, mainly, what artifactual aspects are involved in the teamwork activity. However, as we may find, some notions emerged from our discussion need a better classification. That is, notions like *design constraints*, and *modelling actions* are in fact independent of the existence of the representational artifacts. A larger classifactory scheme for organizing these notions is needed.

and the "individuality" of the communicating tools lies in the "addresses" of the participating IMSs such that communications can be directed effectively across distributed work sites.

[5] We may better characterise this individual modelling act as "constructing domain design expressions by *referencing* to parts of shared $\mathcal{CGS}$."

[6] In Section 4.2, we shall give more explanations of why this is always the case.

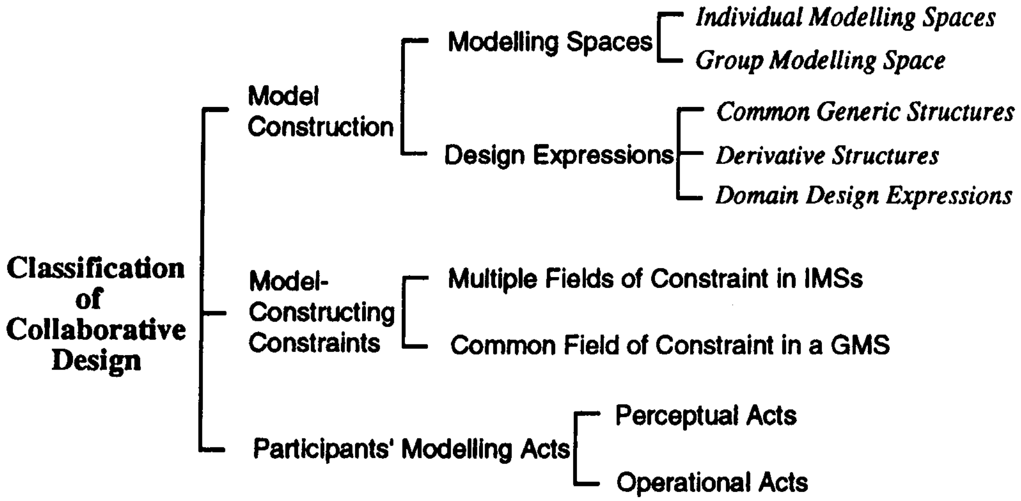
Classification of Collaborative Design
Model Construction
Modelling Spaces
Individual Modelling Spaces
Group Modelling Space
Design Expressions
Common Generic Structures
Derivative Structures
Domain Design Expressions
Model-Constructing Constraints
Multiple Fields of Constraint in IMSs
Common Field of Constraint in a GMS
Participants' Modelling Acts
Perceptual Acts
Operational Acts

Figure 2: A classificatory scheme of collaborative design.

Figure 2 thus presents a classificatory scheme that individuates the complex group design activity into a collection of simpler components. Seen in this scheme, design expressions are constructed and modified (by participants' modelling acts) in modelling spaces. When introduced into modelling spaces (group as well as individual ones), model-constructing constraints (e.g. gravitational forces, systems of spatial grammars etc.), can be applied to *deform* design expressions until an equilibrium (or satisfied) state is reached.

Members of a design team may perform two kinds of modelling acts. By *perceptual* acts, we mean that an individual's acts of constructing or manipulating parts of expressions is motivated (or caused) by his or her *perceiving* (or seeing and understanding) states of expressions. An individual can also perform *operational* acts to derive referential information from $\mathcal{CGS}$, or, more fundamentally, to update the constitution of modelling spaces.

Figure 3 illustrates how the components classified can form an abstract platform of collaborative design. The diagram summarises several features of teamwork in design that we have emphasised:

- *Heterogeneity in IMSs* — In modelling domain design solutions, participants employ individual modelling spaces, often equipped with domain-specific systems of design constraints. To serve individual purposes, the IMSs in use can be highly heterogeneous to each other.

- *Distributedness* — Each participant's modelling domain design solutions can be geographically and logically separate from the rest of team members'.

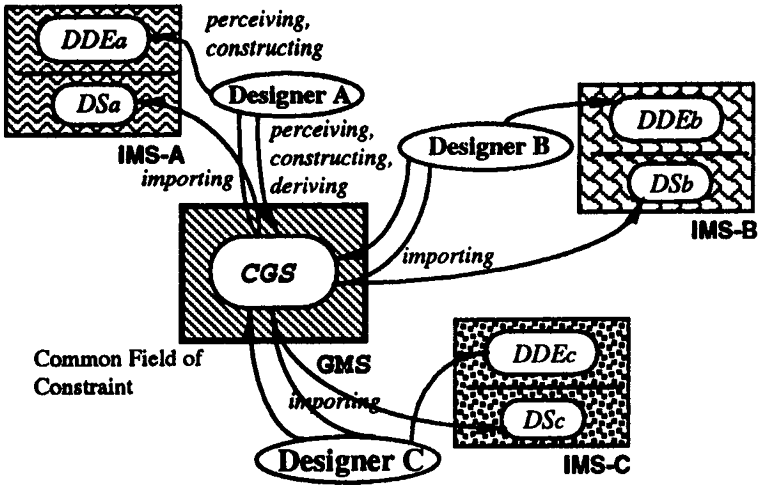

Figure 3: An abstract platform for collaborative design. (The number of participating designers is not definitive.)

- *Structure sharing* — All participants share the states of $\mathcal{CGS}$ modelled in a GMS. The sharing of common structure is manifested in participants' capabilities of (a) accessing to any parts of the structure, (b) changing the state of $\mathcal{CGS}$, and (c) extracting partial design structures for whatever design purposes.

The descriptions given in this section can be considered as an articulation of the 'infrastructural' aspects of teamwork in design. Given the rather static components arrived at, it's our next task to spell out the 'logical' aspects of collaborative design. In searching for the logic of collaboration in the next section, we aim to get a clearer picture of the more dynamic aspects of group design activity.

4 Constraints on Collaboration

In developing a semantic theory of natural language and information, Barwise expresses the following view [Barwise, 1989, p.52]:

> "When we search for the "logic" of some activity, what we are after is the collection of constraints $S \Rightarrow S'$ [7] that govern the activity. For example, the logic of perception consists of the set of constraints that govern perception."

[7] In Situation Theory [Barwise & Perry, 1983], this is read as actual situations of type S which involve there being actual situations of type S'.

Following the situation-theoretical stand-point, it seems plausible for us to think of the logic of collaborative design activity as a set of constraints[8] which governs collaboration. In the search for the constraints on collaboration, we find that a further examination of the properties and the systematic relations among different types of representations (i.e. $\mathcal{CGS}$, $\mathcal{DS}$, and $\mathcal{DEE}$) can tell us more about what is actually involved in collaboration.

4.1 Shareability of $\mathcal{CGS}$

The property of being "shareable" of common generic structure is a critical indicator of the continuing of teamwork. The shareability of $\mathcal{CGS}$ indicates the status of common understanding and judgement achieved and maintained by team members. In certain circumstances, $\mathcal{CGS}$ may become not shareable, hence teamwork cannot continue, due to the following reasons:

- Deformability — $\mathcal{CGS}$ may not be sustainable in a GMS because an equilibrium state of the structure under modelling cannot be reached;
- Multiple-viewpoint — $\mathcal{CGS}$ may not be accessible to some participants in the course of modelling because of the lack of certain types of model constructs or connectors;
- Genericity — $\mathcal{CGS}$ may not be usable to some members because of its derivative structures are not generic on a right level to serve the purposes of domain-related substantiations.

4.2 Consistency among $\mathcal{CGS}$, $\mathcal{DS}$, and $\mathcal{DDE}$

In our classification scheme, we have identified three different types of design expressions that are constructed by participants in different modelling spaces and serve for various purposes. However, in the course of constructing expressions, there exist certain "operational" relations such that the consistency of design information contained in different expression types needs to be maintained. More explicitly, consider the following operational relations:

1. $(\mathcal{CGS})R_d(\mathcal{DS})$ — Derivative structures are extracted expressions from a state of common generic structure. Therefore, $\mathcal{CGS}$ always stands in a relation, denoted as R_d, to $\mathcal{DS}$. The type of relation R_d can be characterised in terms of the derivative devices (methods) used and the spatio-temporal locations (relative to the $\mathcal{CGS}$ in a GMS) of applying the devices.
2. $(\mathcal{DS})R_f(\mathcal{DDE})$ — Domain design expressions are constructed by participants with reference to underlying derivative structures. Therefore, $\mathcal{DS}$ always stand in a relation, denoted as R_f, to $\mathcal{DDE}$. The type of relation R_f can be characterised in terms of '*referring to*', '*instantiating of*', or '*substantiating with*'.

[8] Note that the notion of 'constraints' here has a different nature from that of design constraints described in Section 3.6. Model-constructing constraints are targeting at design problems framed in workspaces. Constraints on collaboration are more to do with the problems of inter-personal communication.

3. $(\mathcal{CGS})R_d(\mathcal{DS})R_f(\mathcal{DDE})$ — Logically, due to the *information flow* between the types of expressions, a complex relation among $\mathcal{CGS}$, $\mathcal{DS}$, and $\mathcal{DDE}$ can be formed. Moreover, instances of the expression types can have different states in the courses of design modelling. As long as the relations R_d and R_f are in operation respectively, there can a problem of maintaining consistency. Consider further two types of design events:

$$\begin{aligned} \mathcal{CGS} \rightsquigarrow \mathcal{CGS}'\ (1)\ ; &\quad \text{(some participant's act causing state changing in } \mathcal{CGS}) \\ \mathcal{DS} \rightsquigarrow \mathcal{DS}'\ (2)\ ; &\quad \text{(because a } R_d \text{ is in operation)} \\ \mathcal{DDE} \rightsquigarrow \mathcal{DDE}'(3)\ ; &\quad \text{(because a } R_f \text{ is in operation)} \end{aligned}$$

Or the event type can be the other way around

$$\begin{aligned} \mathcal{DDE} \rightsquigarrow \mathcal{DDE}'\ (1)\ ; &\quad \text{(some participant's act causing state changing in } \mathcal{DDE}) \\ \mathcal{DS} \rightsquigarrow \mathcal{DS}'\ (2)\ ; &\quad \text{(because a } R_f \text{ is in operation)} \\ \mathcal{CGS} \rightsquigarrow \mathcal{CGS}'\ (3)\ ; &\quad \text{(because a } R_d \text{ is in operation)} \end{aligned}$$

Sharability of $\mathcal{CGS}$ and consistency maintenance among types of model expressions contribute to the main resources that give rise to constraints on collaboration in group modelling activity. On the basis of the above properties and relations examined, we spell out two constraints on collaboration in the following.

Suppose, at some design stage, a participant (say, Designer A) decides to make some changes in $\mathcal{DS}s_A$ (i.e. the set of derivative structures used by Designer A) to maintain (or validate) an intended domain design solution. Consequently, A's changing $\mathcal{DS}_A$ leads to a changing state of the $\mathcal{CGS}$ which is shared by other participants (say, Designers B and C). Owing to the deformability of the $\mathcal{CGS}$, the derivative structures, $\mathcal{DS}_B$ and $\mathcal{DS}_C$, used by B and C respectively may get changed in order to maintain the derivative relations introduced previously. This gives rise to at least two circumstances calling for communication among A, B and C:

1. [Constraint $Coor$: $Collaboration \Longrightarrow Coordination$]
   Coordination is involved if A's changing $\mathcal{DS}_A$ is seen in $\mathcal{CGS}$ and judged desirable by B and C, as they inspect the consequent states of their own derivative structures. Under this circumstance, B and C need to coordinate A's proposals by making further developments in their domain design solutions in respect of the changed $\mathcal{DS}_B$ and $\mathcal{DS}_C$.

2. [Constraint $Nego$: $Collaboration \Longrightarrow Negotiation$]
   Negotiation is involved if A's changing $\mathcal{DS}_A$ is judged undesirable by B and/or C, as they inspect the differences occurred in the changing states of $\mathcal{DS}_B$ and/or $\mathcal{DS}_C$. Under this circumstance, A needs to negotiate with B and/or C by either dropping completely the intended changes in $\mathcal{DS}_A$, which requires A to develop his or her domain design in

a different direction, or requesting B's and/or C's suggestions of the extent to which the changes in $\mathcal{DS}_A$ are acceptable.[9]

5 Basic Requirements for Computer Support

Given the constraints on collaboration arrived at the above, a natural move forward is to specify requirements for computer supports, aiming at an ultimate design of a computer-based modelling environment that can support collaborative design activity. We may rightly ask "What do these constraints tell us about how collaborative design, as described in this paper, can be *computer-supported*?". However, we do not intend to present a formal and complete requirement specification at present. Instead, we now discuss some of the requirements, as prompted by our current analysis, for identifying areas of prospective computer support which have not been fully addressed or interconnected with other related research work.

(Issue 1) Support for the Construction of a GMS

As clearly revealed in the structuralist scenario, collaborative design begins with the construction of a common modelling space. Given the initial demand, a computer-based design environment may have to provide, in the first instance, representation supports in the users' construction of a GMS.

[1.1] *Representation of multiple viewpoints.* — More specifically, this requirement can be subdivided into the representation of two kinds of model objects:

[1.1.a] *Types of model constructs* — Participants working in various aspects of a project need to introduce types of model constructs that he or she considers pertinent representations of design elements. Different viewpoints in a GMS may be better represented by various types of *model constructs.* Instances of model constructs can interact with form-giving forces or constraints applied in a GMS and exhibit certain behaviours of deformation.

[1.1.b] *Types of model connectors* — When types of model constructs are introduced by participants, model connectors, the devices to connect or disconnect instances of the constructs, are essential. Types of connectors are used by participants to define and effect ways of manipulating parts of the common structures for various reasons. Note that model connectors are *neutral* objects in a sense that they do not represent any specific design elements in the real world.

[9] Note that A's receiving the suggestions made by B and/or C is same as how B and/or C may recognise A's intention expressed in changing $\mathcal{DS}_A$; the cognitive basis for A to do so is, again, the deformability of $\mathcal{CGS}$ and the relations between $\mathcal{CGS}$ and $\mathcal{DS}$.

[1.2] *Representation of constraint system for shaping $\mathcal{CGS}$.* — Design participants are not expected to build up, computationally, a common constraint system by themselves for modelling $\mathcal{CGS}$, since this demands highly technical knowledge. It would be a task for system engineers to develop computational models that can interact with instances of model constructs and connectors introduced by participants. The design of constraint systems of a GMS can be of the following nature:

[1.2.a] *General constraint systems* supporting physical (or, more broadly, environmental) laws such as gravity, thermal energy, or acoustics etc.

[1.2.b] *Specific constraint systems* supporting intentional laws such as particular systems of spatial or shape grammars.

[1.3] *$\mathcal{CGS}$ is pictorial and generic.* — Representation of $\mathcal{CGS}$ requires to be graphical and, at the same time, generic for the following two reasons:

[1.3.a] In serving *all* members of a design team as a common (global) representational medium, $\mathcal{CGS}$ is essentially *pictorial*, or, at least, *diagrammatical.* This implies that the construction of $\mathcal{CGS}$ has to be based on graphical objects so that all participants of different backgrounds can feel relatively easy to be familiar with.

[1.3.b] $\mathcal{CGS}$ is essentially generic in order to be enriched or refined to different levels of specificity. Therefore, its representation requires, perhaps, a higher order of genericity to support the following flow of information:

$$\mathcal{CGS} \xrightarrow{instantiation-of} \mathcal{DS} \xrightarrow{substantiation-with} \mathcal{DDE}$$

(Issue 2) Support for the Construction of IMSs

In teamwork, a participant's development of domain design solutions is not less important than that of common structures. To carry out more technical modelling tasks, participants need to work with *personal* workspaces which are not necessarily known and accessible to others. The problem is how to have a system capable of interacting with a user and generating an IMS which he or she thinks pertinent to the design tasks at hand. This requirement gives rise to the following sub-issues:

[2.1] *Representation of individual design worlds* — This includes, firstly, a set of personal design constructs for generating and manipulating domain design expressions, secondly, domain-oriented constraint systems employable in shaping domain design developments.

[2.2] *Support for the construction of $\mathcal{DDE}$ with reference to $\mathcal{DS}$* — The spaces for constructing $\mathcal{DDE}$ is required to be *overlapped* or *juxtaposed* with the spaces for holding $\mathcal{DS}$ as design referents.

[2.3] *Support for the construction of $\mathcal{DDE}$ by substantiating $\mathcal{DS}$ with domain design elements (substances)* — This is a user's need for direct use of $\mathcal{DS}$ imported from a GMS. The type of construction process involves enriching or refining $\mathcal{DS}$ into $\mathcal{DDE}$ filled with more domain design details.

(Issue 3) Support for Coordination and Negotiation

The representations in a GMS and multiple IMSs discussed above are the infrastructural supports for the users' setting up group as well individual workspaces. Given the infrastructures outlined, we are in a position to spell out further system requirements of more dynamic features. The third issue is concerned with system ability to support *coordination* and *negotiation* among design participants. In accordance with the constrains on collaboration explained in Section 4, the following communication requirements are expected to be fulfilled in the development of a collaborative modelling environment:

[3.1] *Detection of state change in $\mathcal{CGS}$* — It is clear to us that $\mathcal{CGS}$ is a dynamic object subject to participants' manipulations from different viewpoints. It is the evolving of a common structure that gives rise to the dynamism of teamwork. For a design environment to fit into the dynamic situation, it has to be concerned with the facts about state change in $\mathcal{CGS}$. But how do we define such a state change?

[3.1.a] A state of $\mathcal{CGS}$ is defined by a two or three dimensional deployment of instances of model constructs and connectors under the influence of a global constraint system activated in a GMS.

[3.1.b] A state change in $\mathcal{CGS}$ can therefore be defined as a change in (parts of) an existing deployment (or, a better word, configuration) resulting from a net effect of some participant's or participants' modelling actions together with the constraint influence.

A system's ability to keep track of state change in $\mathcal{CGS}$ lies in if the system can generate information about the configuration differences between two $\mathcal{CGS}$ states given at a time. This bit of information is essential for the system to trigger further communication mechanisms, such as the maintenance of R_d and messages delivering for users' maintaining R_f (see below). Seen in this requirement, a detection mechanism, so to speak, is needed.

[3.2] *Maintaining the relation R_d in $(\mathcal{CGS})R_d(\mathcal{DS})$* — In developing domain design solutions, participants need to extract derivative structures from a state of $\mathcal{CGS}$ as design resources or references. Since the state of $\mathcal{CGS}$ may keep changing, it is a useful support for participants if a system can inform the users timely the changing states of $\mathcal{DS}$ in use, arising from state change in $\mathcal{CGS}$. This requires a system to keep a record of the relation between $\mathcal{DS}$ and $\mathcal{CGS}$ and compute updated states of

$\mathcal{D}S$ whenever $\mathcal{CG}S$ gets changed. Apart from the state of $\mathcal{CG}S$, two representations are necessarily involved in a system's maintaining the relation R_d:

[3.2.a] *Representation of derivative actions* — To derive a $\mathcal{D}S$, users require to perform certain *spatial operations*, such as *projecting, subdividing, or slicing* etc., upon $\mathcal{CG}S$. Taken as a bit of information, a *derivative action* thus consists of the performer and the spatial operation performed.

[3.2.b] *Representation of location of deriving* — The information about the time and position (relative to $\mathcal{CG}S$ modelled in a GMS) in which a derivative action takes place is also relevant in keeping a R_d.

[3.3] *Messages delivery for maintaining the relation R_f in $(\mathcal{D}S)R_f(\mathcal{DDE})$* — Standing in a domain design perspective, a participant shall perceive his or her development of domain design solutions as design consequences in relation to a state of $\mathcal{CG}S$. By judging the development resulted, any participants may well be motivated to make changes in $\mathcal{DDE}$. This kind of design change activity gives rise to a second dynamism to the course of teamwork. As explained before, there exists the systematic relation, R_f, between $\mathcal{D}S$ and $\mathcal{DDE}$. Given a change in $\mathcal{DDE}$ desired by some individual, a R_f will not be sustainable if state changes in $\mathcal{D}S$, and hence in $\mathcal{CG}S$, are not reflected correspondingly.

A usable collaborative modelling environment should, therefore, not only allow for participants to freely make changes in $\mathcal{DDE}$ in their IMSs, but also assist the individuals in dealing with the problem of maintaining R_f. To support this communication need, two functionalities are considered necessary:

[3.3.a] *Detection of state change in $\mathcal{D}S$* — A detection mechanism similar to that of detecting $\mathcal{CG}S$ state change is needed. But the detection functions need to be installed locally as IMSs may be distributed over a number of separate working sites.

[3.3.b] *Sending the change message to GMS* — When a state change ($\mathcal{D}S \leadsto \mathcal{D}S'$) is computed, a message is sent to GMS for activating corresponding state change in $\mathcal{CG}S$.

When GMS receives and processes the message sent from IMSs, a change in $\mathcal{CG}S$ will be implemented by the system, resulting in $\mathcal{CG}S'$. Owing to the mechanism of maintaining R_d described in [3.2], further messages (containing the information about $\mathcal{D}S \leadsto \mathcal{D}S'$) sending from GMS to IMSs shall naturally follow so that other participants involved shall be informed. The detection and message delivery mechanisms described here seems to suggest a *local management agent* be set up in an IMS which is the sole information space for the agent to serve.

[3.4] *Communication channels for resolving conflicts manifested in* $\mathcal{CGS} \rightsquigarrow \mathcal{CGS}'$ — If a coordination situation, as described previously on page 11, cannot be reached, negotiation among the individuals involved in the disagreement is needed to resolve the conflict. Since the situation is a highly non-deterministic one, a system is not expected to automatically detect the arising of a conflict and resolve it. In principle, this should be left to the participants to decide if coordinating or negotiating. In coordination, there is no need for participants to express individual judgements of the state of $\mathcal{CGS}$, and corresponding changes in $\mathcal{DVE}$ shall be carried out in IMSs separately.

More problematically, in negotiation, participants need to express disagreement to one another[10]. This demands a system to provide users with communication channels with which they can discuss, directly or indirectly, and resolve the differences in recognising the state of $\mathcal{CGS}$ until the sharability is re-established among members of a design team.

6 Related Research and Further Work

To investigate the possibility of computer-supported collaborative design, we have started from a study of the structuralist approach to teamwork in architectural modelling. By carrying out a natural analysis of the structuralist scenario, a classification scheme that explains the constitution of collaborative design activity is presented. Guided by an examination of the properties of the types of representation and the systematic relations among them, a logic of collaboration in teamwork is found. The constraints spell out what is involved when members of a design team co-work on the substantiation of a common generic structure with heterogeneous design developments in a distributed manner. Following the constraint presentation, we then give a discussion of the basic requirements for prospective computer supports.

For the purpose of drawing up a promising strategy for a further exploration, we have some readings from other researchers. In relation to our current enquiry, the following collection of research references are of a particular interest:

1. In their search for what makes research on Computer Supported Cooperative Work (CSCW) a unique research field, Schmidt and Bannon propose a general conceptual framework for CSCW (see [Bannon & Schmidt, 1991, Schmidt & Bannon, 1992]). In particular, they identify that the priority of computer support should be given to supporting a group of users for *articulation work* and the *construction of a common information space.* Our

[10] Again, to use the negotiation situation described on page 11, this is to say that B and/or C must find a way to let A know that A's intention in making the change in $\mathcal{CGS}$ is not acceptable.

findings in supporting the structuralist approach to collaborative design appear in tune with the Schmidt-Bannon framework.

2. Based on analyses of organizational problem solving in scientific communities, Leigh Star derives the concept of *boundary objects* and suggests the concept would be an appropriate data structure for Distributed Artificial Intelligence (DAI) [Star, 1989]. Star identifies four types of boundary object which are considered as a major method of solving heterogeneous problems. Notably, the properties of boundary objects bear a close relation to those of our common generic structures [Star, 1989, p.46]:

   > "Boundary objects are objects that are both plastic enough to adapt to local needs and constraints of several parties employing them, yet robust enough to maintain a common identity across sites. They are weakly structured in common use, and become strongly structured in individual-site use."

   We suggest that the $\mathcal{CGS}$ in our case can be another candidate for a type of boundary object to be used in collaborative design but with a generic-specific structural adaptation instead of a weak-strong one. Though the general properties of boundary objects are researched, no computational representations of the objects have been proposed.

3. Research on computer graphics models, which can respond to in a natural way to applied forces or constraints, has shown us the technical possibilities of representing $\mathcal{CGS}$ graphically in computers. In particular, three research results worth noting: the theory of elasticity was employed by Terzopoulos *et al.* to construct elastically deformable models [Terzopoulos *et al*, 1987]; Witkin and others explored the representation of (geometrical) constraints as energy functions that behave like forces pulling and deforming parts of the model into place [Witkin *et al*, 1987]; three force-based constraint methods were explored by Platt and others to add several desirable properties into flexible models [Platt & Barr, 1988].

   It certainly remains to be seen how the kinds of graphics model achieved above can serve in a collaborative design context, satisfying the demands for being generic and manipulable for multiple design purposes. As for representing design constraints on a smaller scale, Gross and others developed constraint-based design environments as separate specialized design "Labs" (see [Gross *et al*, 1988] for detail). We see this work as a precedent experiment to the setting of private constraint systems in distributed IMSs.

4. System research and developments on computer-supported human communication in cooperative work have presented two distinct approaches: one is in favour of supporting informal interaction among users through the design of *shared virtual workspaces*; the other focuses on supporting formal interaction mediated by *communication protocols*. Research

prototypes of shared drawing systems have demonstrated a range of technological means to re-create face-to-face communication where users are actually separated geographically (see, for example, [Bly & Minneman, 1989, Lee, 1990, Tang & Minneman, 1991, Lu & Mantei, 1991, Ishii *et al*, 1992] among many others, and [Peng, 1992a] for a more detailed survey).

Along with the second line, several computer-based coordinating protocols have been implemented. The building of these mechanisms is mainly based on a formal representation of either particular work procedures or special knowledge involved in the design tasks. In the domain of designing computer configuration for buildings, for example, the knowledge-based design environment *NETWORK* was implemented to test the idea of integrating the domain knowledge of configuring computer network and the communication between designers [Fischer *et al*, 1992, Reeves & Shipman, 1992]. Bond and others developed a set of ***rules of interaction***, arising from "an organisationally agreed sequence of commitment steps", to model the collaboration among specialists in aircraft design [Bond, 1989, Bond & Ricci, 1992].

It remains questionable, however, if the knowledge-based approach to the design of computer-based coordinating mechanisms can satisfy the needs of less stabilised group practice in design. The encapsulation of specific knowledge about artifacts or procedures can be problematic to collaborative design that demands unique solutions to every single project. Obviously, it is not very sensible to design a collaborative design environment centred on the funicular structure shown in the Colonia Ǵuell church project; as we know, there are always innovative building structures being developed.

The above overview of related research shows the complexity involved in developing a realistic collaborative design environment. It covers a very wide spectrum of conceptual and technical issues. To further our current work, we choose to focus on constructing a coordinating theory that is in tune with the representational and communication requirements elicited in this paper.

Acknowledgements This work was mainly carried out when the author was staying at EdCAAD, which was partially supported by the UK ORS Awards. The author wishes to thank the joint guidance from John R. Lee and Aart Bijl in developing the paper. Thanks also to the comments and supports from John Downie, Lyn Pemberton, Simon Shurville, and Donia Scott of the CSCD Group at the University of Brighton.

References

[Bannon & Schmidt, 1991] Bannon, L. J. and Schmidt, K. (1991). CSCW: Four characters in search of a context. In *Studies in Computer Supported Cooperative*

Work: Theory, Practice and Design, pages 3–16. North-Holland: Elsevier Science Publishing Company, Inc.

[Barwise & Perry, 1983] Barwise, J. and Perry, J. (1983). *Situations and Attitudes.* The MIT Press.

[Barwise, 1989] Barwise, J. (1989). *The Situation in Logic.* CSLI Lecture Notes No. 17, Center for the Study of Language and Information, Stanford.

[Bly & Minneman, 1989] Bly, S. L. and Minneman, S. L. (1989). Commune: A shared drawing surface. Technical Report SSL-89-86, System Sciences Laboratory, Palo Alto Research Center.

[Bly, 1988] Bly, S. L. (1988). A use of drawing surfaces in different collaborative settings. In Greif, I., (ed.), *Proceedings of the Conference on Computer- Supported Cooperative Work (CSCW'88)*, pages 250–256. ACM Press.

[Bond & Ricci, 1992] Bond, A.H. and Ricci, R.J. (1992). Cooperation in aircraft design. *Research in Engineering Design*, 4(4):115–130.

[Bond, 1989] Bond, A. H. (1989). The cooperation of experts in engineering design. In Gasser, Les and Huhns, M. N., (eds.), *Distributed Artificial Intelligence Vol. 2 (Research Notes in Artificial Intelligence)*, pages 463–483. London: Pitman.

[Collins & Nonell, 1983] Collins, G. R. and Nonell, J. B. (1983). *The Designs and Drawings of Antonio Gaudí*, pages 31–35. Princeton, N.J. Guildford: Princeton University Press.

[Fischer *et al*, 1992] Fischer, G., Grudin, J., Lemke, A., McCall, R., Ostwald, J., Reeves, B. and Shipman, F. (1992). Supporting indirect, collaborative design with integrated knowledge-based design environments. *Human-Computer Interaction.* (To appear in Special Issue on Computer Supported Cooperative Work).

[Greenberg, 1991] Greenberg, S. (July 1991). An annotated bibliography of Computer Supported Cooperative Work. *SIGCHI Bulletin*, 23(3):29–62.

[Gross *et al*, 1988] Gross, M.D., Ervin, S.M., Anderson, J.A. and Fleisher, A. (July 1988). Constraints: Knowledge representation in design. *Design Studies*, 9(3):133–143.

[Ishii *et al*, 1992] Ishii, H., Kobayashi, M. and Grudin, J. (1992). Integration of inter-personal space and shared workspace: ClearBoard design and experiments. In Turner, J. and Kraut, R., (eds.), *Proceedings of CSCW 92*, pages 33–42. ACM Press.

[Lakin, 1983] Lakin, F. (1983). Measuring text-graphic activity. In *Proceedings of the GRAPHICS INTERFACE '83*. Edmonton, Alberta.

[Lee, 1990] Lee, J. J. (1990). Xsketch: A multi-user sketching tool for X11. In *Proceedings of the Conference on Office Information Systems*, pages 169–173.

[Lu & Mantei, 1991] Lu, I. M. and Mantei, M. (1991). Idea management in a shared drawing tool. In Bannon, L., Robinson, M. and K., Schmit, (eds.), *Proceedings of ECSCW'91*, pages 97–112. Kluwer Academic Publisher.

[Martinell, 1979] Martinell, C. (1979). *Gaudí: his Life, his Theories, his Work*, page 335. Barcelona Editorial Blume.

[Peng, 1992a] Peng, C. (June 1992). A Survey of CSCW Designs in Shared Drawing Space. Working paper, EdCAAD, University of Edinburgh. vv In submission.

[Peng, 1992b] Peng, C. (September 1992). Exploring communication in collaborative design: A cooperative architectural modelling perspective. Working paper, EdCAAD, University of Edinburgh. To be published in the Journal of *Design Studies* on a special issue on collaborative design, 1993.

[Peng, 1992c] Peng, C. (1992c). Participatory architectural modeling: Common images and distributed design developments. In *Proceedings of the Participatory Design Conference 1992 (PDC'92), Kresge Auditorium, Massachusetts Institute of Technology, Cambridge MA US, November 6-7 1992*, pages 171–180. Computer Professionals for Social Responsibility (CPSR).

[Platt & Barr, 1988] Platt, J.C. and Barr, A.H. (1988). Constraint methods for flexible models. *ACM Computer Graphics*, 22(4):279–288.

[Reeves & Shipman, 1992] Reeves, B. and Shipman, F. (1992). Supporting communication between designers with artifact-centered evolving information spaces.

In Turner, J. and Kraut, R., (eds.), *Proceedings of CSCW 92*, pages 394–401. ACM Press.

[Schmidt & Bannon, 1992] Schmidt, K. and Bannon, L. (1992). Taking cscw seriously: Supporting articulation work. *Computer Supported Cooperative Work*, 1(1-2):7–40.

[Star, 1989] Star, S. L. (1989). The structure of ill-structured solutions: Boundary objects and heterogeneous distributed problem solving. In Gasser, Les and Huhns, M. N., (eds.), *Distributed Artificial Intelligence Vol. 2 (Research Notes in Artificial Intelligence)*, pages 37–53. London: Pitman.

[Tang & Leifer, 1988] Tang, J. C. and Leifer, L. J. (1988). A framework for understanding the workspace activity of design teams. In *Proceedings of the Conference on Computer- Supported Cooperative Work (CSCW '88)*, pages 26–28. ACM Press.

[Tang & Minneman, 1991] Tang, J. C. and Minneman, S. L. (1991). Videowhiteboard: Video shadows to support remote collaboration. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 315–322. ACM Press.

[Terzopoulos *et al*, 1987] Terzopoulos, D., Platt, J., Barr, A. and Fleischer, K. (July 1987). Elastically deformable models. *ACM Computer Graphics*, 21(4):205–214.

[Witkin *et al*, 1987] Witkin, A., Fleischer, K. and Barr, A. (July 1987). Energy constraints on parameterized models. *ACM Computer Graphics*, 21(4):225–232.