Distance Collaboration. A Comparative Analysis of Tools and Procedures

Giuseppe Pellitteri, Benedetto Colajanni, Salvatore Concialdi
Dipartimento di Progetto e Costruzione Edilizia, University of Palermo
pellitt@unipa.it; bcolajan@unipa.it; concialdi@unipa.it

Abstract. *Besides design theory and practice, curricula of architectural students should include some experiences referring to professional situations. Among these experiences, Collaborative Design is nowadays somewhat frequent. It is normally practised by large professional studios, using expansive software which is beyond what they can afford on average. Much academic research on the topic has also been carried out often resulting in the proposition of new and too complex description models of the building object. We think that students should instead get acquainted with such a design process: an experience has been planned and carried out in our Department for the purpose of practising the possible paradigm in a more ordinary context. Its purpose was threefold. First, making the students grasp the method's potentialities and learn the right approach. Second, testing the practical suitability of the most widely used software. Third, comparing their relative efficiency. The software we used was: Architectural Desktop, AutoCad Revit, ArchiCad for Teamwork. We focused special attention on how representing and managing restraints, since they are the main source of conflicts. This was the hardest topic to manage. The results were partly positive inasmuch as the experience showed that it could be possible to adopt the Collaborative Design paradigm which is also used in the AEC field. The drawbacks emerged from the analysis of non-dedicated software are: a relative slow process for the lack of certain specific tools; a subsequent necessity of integrating them with different communication software; the difficulty of managing hard and soft restraints. However, in the final analysis, the experience can be considered as positive.*

Keywords. *Collaborative Design, Architectural teaching.*

Introduction

Architectural training has to deal not only with the changing technology and formal trends but also with the changing design procedures for buildings. Indeed, Collaborative Design is likely to be the paradigm which architects will have to cope with from the beginning of their professional career.

Buildings are getting more and more complex. Performance requirements are becoming more and more demanding. Consequently, the design

process involves a substantial and growing number of partners. Hence, this increases the difficulty of adopting the traditional way of collaboration among partners: there are sessions when all of them are present or at least the ones whose fields of competence concern the decisions to be taken in that given session. “Collaborative Design” is a way to overcome such a difficulty; even if it is too general a definition, it is widely used to refer to distance collaboration. Many researchers have transformed this operative procedure into a theoretical topic. They have developed complex behaviour models, whereby building components and relationship protocols between the design partners are represented. Things may not be so complex, indeed distance collaboration can be carried out also in a simple way using more widely-used software.

Training architecture students to these procedures is an important educational task. Students are familiar with the most diffused architectural design software. Thus, it is appropriate that the first experiences in Collaborative Design are made using it, avoiding specialised, complex and expensive software. If it is tested with more than one software, a kind of “home-made” procedure allows students to explore possible different solutions to the problems of distance collaboration.

For this reason, we carried out such an experience in our Department with the ultimate aim of verifying the feasibility of inserting it in the current teaching program. The experience involved designing a simple residential building. It has been repeated three times using “Architectural Desktop”, “Autodesk Revit” and “ArchiCad for Teamwork”. The experience began in 2003, with the releases existing in that year. Since then, new useful tools have been added to the software in use thus providing further help: for instance, in “Architectural Desktop 2006” a new tool deals with beam and columns layout.

The procedural structure

A Collaborative Design process develops a complete model of the architectural object to be later built through successive phases.

In order to achieve this result, the operative procedure has to:

1. let each partner work on the building parts pertaining to his/her competences;
2. assemble each partner’s work into an overall and consistent model of the building. Hence, this task entails the use of specific tools for:
3. assessing the appropriateness of each proposed change to the development of the project;
4. detecting the possible source of conflicts and/or inconsistencies;
5. resolving them;
6. managing times, sequences and ways of coordinating phases 1 and 2, as well as subtasks 3, 4 and 5.

Task 1 can be performed in many ways, according to the conceptual division of design competences among the design partners, i.e., either according to types of building elements (see ISO Norm 6290) or geometrical-topological building parts.

As for task 3 and 6, only one member of the design team should be entrusted with them: the coordinator. A certain degree of automatism is desirable in task 4. Task 6 could involve a hierarchical structure within the design team.

The most frequent conflicts are the geometric-topological ones. The most frequent cases are two:

1) Two objects, belonging to two different fields of competence, are proposed and take up the same space, e.g. the structural engineer wants to place a column where the architect designed a window or a door.

2) An object fulfils more than one function which belongs to different fields of competence;

the designer in charge of the different competences wants to provide the object with different characteristics. An example: a structural engineer suggested a concrete wall of considerable thickness to be fully functional while the architect thought of it as a visible-brick wall built as a partition. An oriented software compatible with multi-user collaboration usually assigns each specialist designer only one part of the whole project. However, the automatic detection of this kind of conflicts is practically impossible while the geometric-topological ones have a graphical expression and, thus, can be detected more easily.

The experience with Architectural Desktop

The experience is very simple. Only two partners were simulated: the architect and the structural engineer. The former was also the coordinator. Indeed, the absence of at least a third team member has left out any possible experiences of direct relationship between partners, without the coordinator's intervention. Some conflicts could be resolved through direct negotiation of the interested partners and then notified to the coordinator. Our intention is the experience with more than two partners. Let us now examine the way how the different tasks have been dealt with.

The division of competences between architect and structural engineer can account as a specimen from the wider division of competences according to the typology of building elements. "Constructs" can be used to develop a further information organization with geometric-topological criteria.

The easiest way of creating independent "work spaces" for each partner is to structure the information model in layers. On his/her computer, each partner has a model including all the partners' layers. All the partners can only work with the layers they are entrusted with and can see the others loading them with "Xref". Non-graphical information is exchanged either by e-mail or, preferably, by "Microsoft NetMeeting".

Two procedures have been experienced.

Procedure A: the coordinator's computer is the server containing the general model. As the process starts, simplified plans drawn by the architect are sent to the other members who will load them into their computers using "Xref". At the beginning, layers pertaining to the other partners are empty. During the following phases, each partner suggesting a change always keeps a backup copy of his/her previous solution. He/she then informs the coordinator about the planned adjustment. At this point, the coordinator loads the proposing partner's layers with "Xref", assessing whether the proposal is consistent with the purpose of the project and preventing possible conflicts and/or inconsistencies. Meanwhile, those partners who may be interested have been informed of a possible change by a signal for a "Xref" modification appeared on their notifying bar. If the suggested change is approved, the coordinator informs all the partners that they can load the new "Xref".

As the collaboration phase is asynchronous, information is exchanged via e-mail. In case of more than two partners, the conflict could concern only a couple of them. After informing both of them, the coordinator should charge them with solving the problem. Obviously, the coordinator should supervise the whole trade off in order to prevent any possible conflict with other partners' tasks.

Conflicts rise if restraints are not respected. However, they can be early detected if restraints are shown as soon as possible, i.e. in the very moment of their inclusion. Geometric conflicts occur when two building elements occupy the same space. Spatial restrictions can be either hard or soft and regard position and/or dimensions. Hard restrains do not permit any change pertaining to both position and dimensions. Soft constraints allow some changes: indeed the software is provided with a tool which states the allowed amount of variations. Hard constraints can be clearly visible on drawings: the restrained elements are described in an

acknowledgeable way, for instance with a different colour. Here, they are marked in red. Besides graphical identification, soft constraints need some alphanumerical information.

Here, we chose hypertextual links with text files or other annotation files stored in the server as attributes.

Other conflicts may regard alphanumeric characteristics as density, heat conductivity, permeability, colour. For this kind of constraints, no resolution method has been tested because both partners were mainly interested in geometric-topological decisions. However, solutions can be found only by direct negotiation via e-mail or "NetMeeting".

Procedure B aims to facilitate the relationship between coordinator and team members. The coordinator has two files containing the model: the "official" one is accessible to all partners while the second is not. The coordinator alone can load any modification proposed by the partners into the second model. When a team member devises an adjustment, he/she then informs the coordinator who, after having loaded the modified layers on the second model, checks their appropriateness and consistency. If the change is approved, the coordinator allows the proponent to load the modified layers on the "official" model and informs the other partners about the new "Xref". The advantage of this procedure is that non-interested members are left out of the possible negotiations.

Although our remarks do not refer to the updated version of "Architectural Desktop", our experience emphasized some difficulties in using it for distance collaboration, namely:

- interactions between objects should be enhanced;
- imposing restraints is impossible;
- geometric-spatial conflicts are not automatically detected by the software.

We found an autonomous but partial solution to the last two drawbacks: restraints should be shown (either by means of differentiated graphic

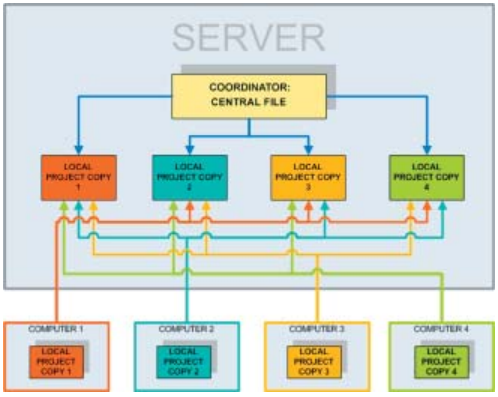

Figure 1. Relationship between "Xref" files in Autodesk and Architectural Desktop (Procedure A).

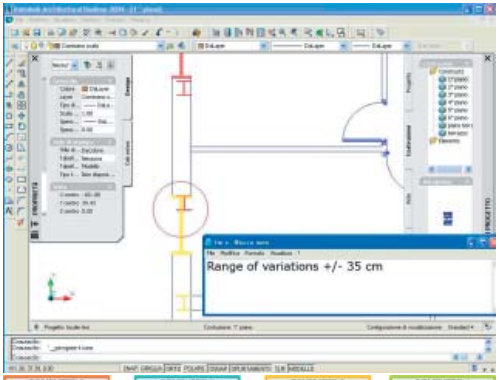

Figure 2. Example of soft constraints and hypertextual annotations in order to resolve geometric conflicts .

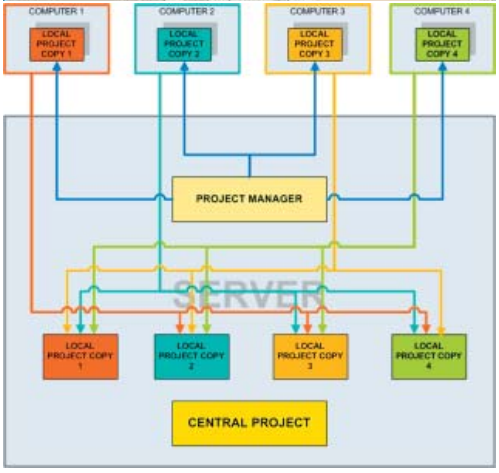

Figure 3. Relationship between "Xref" files in Autodesk and Architectural Desktop (Procedure B) .

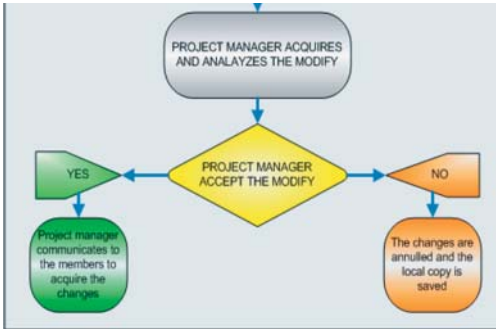

Figure 4. Flow diagram of the Procedure A – Left; and the Procedure B - Right.

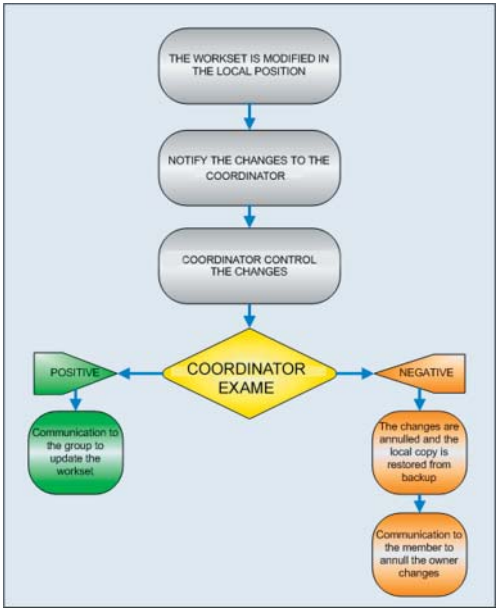

Figure 5. Flow diagram of the proposed procedure in Autodesk Revit.

signs and/or of hypertextual links) as soon as they become essential to the project.

The experience with Autodesk Revit

Characters and limits of this experience are the same as in the previous paragraph.

The architect is the coordinator; his/her computer functions as the server where the "Central File" is stored.

"Revit" is a powerful and sophisticated software. Its main characteristics are a parametric management of the objects and a strong network of functional relationships linking them. The consistency of the project is maintained by those features. Furthermore, "Revit" is provided with the "workset", a specially-tailored software tool for multiple collaboration.

A workset can be considered as a "design space" which is assigned to each partner. The "central file" is the sum of all the worksets. Each team member in charge of a workset is the only one who is allowed to work on it.

The "central file" contains the advances made in each partner's work. Since it is continuously updated and visible to all the team members, it also shows the new relationships that any "workset" modification establishes between the already existing building components. Thus, it is also the space where conflicts may be visible. Given that it is possible to define some objects as non-modifiable, hard constraints can be described and managed rather easily. As for extra "Revit" software ("NetMeeting" or similar), it is very useful to deal with soft constraints, which, here too, have been shown by graphical signs. Furthermore, a very interesting tool which did not exist at the time of our experience is the "element borrowing". It can be very effective in solving inconsistencies arising from different qualities given to the same object by two partners. That can be alternately modified until a common solution is obtained.

When a new "workset" is published every member sees it but the copy of the "central file" stored in his/her computer remains unchanged. The coordinator assesses appropriateness and consistency of the new "workset". After that, if the adjustment is accepted, he allows the other partners to load it into their individual files; otherwise he restores the previous workset version into the "central file".

The experience with Archicad for Teamwork

Framework and starting point for this experience are the same as the previous two.

"Archicad for Teamwork" is a specific software for multi-user distance working, equipped with multiple tools which are missing from other similar software. Here too, a central file contains all the team members' work. However, one of its peculiarities is the hierarchical organisation within the design team: establishing some rigid roles such as "Teammate", "Team Leader", "Administrator" and "View Only" only works in highly structured and large professional studios while in smaller ones "Administrator" and "Team Leader" can often be the same. Moreover, if it is not balanced very carefully, the distribution of power can lead to some inconsistencies in "Archicad for Teamwork". Indeed, while on the one hand each "Teammate" is free to define his/her workspace and link it with the central file, on the other, the "Team Leader", having considerable power, can influence the very structure of the individual workspaces. Hence, some preliminary coordination would be very useful. Here too, the same solution as for "Revit" has been adopted, unifying the two roles of "Administrator" and "Team Leader". Moreover, "Net-Meeting" sessions have been held to deal with two kinds of decisions. First, whether to reject or approve the new versions of workspaces proposed by team members and, consequently, to allow the upgrading of individual copies into the central file.

Conclusions

The environment where our experimentation has been carried out is significantly different from the ones for which the tested software has been created. As it has been said before, a reliable assessment of the software's appropriateness and efficiency was not our ultimate goal. In fact, the starting point was an experience of distance collaboration, which although simplified, should be included in the educational curriculum of any student of architecture. From the educational viewpoint, it is extremely important that students develop the skill to understand and learn the behavioural approach needed during a process of Collaborative Design. For this purpose, entrusting the students with the responsibility of analysing and choosing the best tool is more efficient than proposing them a given software. Indeed, one of the most satisfactory outcomes of this experience was the development of this ability. Our Students succeeded in developing and operating small pieces of procedure as well as in adapting the software to a real even if marginal situation. Furthermore, our experience has proved valuable also in terms of the later improvements made in the software that we tested. The updated versions are aimed at overcoming the drawbacks emerged during our analysis.

References

Mao-Lin, C. and Ju-Hung, L.: 2005 Information and IN-formation Information mining for supporting collaborative design, Automation in Construction, 14(2), pp. 197-205.

Liu, H.; Tang, M. and Frazer, J. H.: 2004 Supporting dynamic management in a multi-agent collaborative design system, Advances in Engineering Software, 35(8-9), pp. 493-502.

Craig, D. L. and Zimring, C.: 2002 Support for collaborative design reasoningin shared virtual spaces, Automation in Construction, 11(2), pp. 249-259.

Colajanni, B.; Pellitteri, G.; Concialdi, S.: 2002 Intelligent Structures for Collaborating with the Architect, in K. Koszewski. S. Wrona (eds.), Connecting the Real and the Virtual - Proceedings of the 20th Conference on Education in Computer Aided Architectural Design in Europe, Warsaw, pp. 360-364.

Shungo, W.; Eunjoo, L.; Tsuyoshi, S.:2001 The multiuser workspace as the medium for commu-

nication in collaborative design, Automation in construction, 10(3), pp. 303-308.

Peña-Mora, F.; Hussein, K.; Vadhavkar, S.; Benjamin, K.: 2000 CAIRO: a concurrent engineering meeting environment for virtual design teams, Artificial Intelligence in Engineering, 14(3), pp. 203-219.

Colajanni, B.; Concialdi, S.; Pellitteri, G.: 1999 CoCoMa: a Collaborative Constraint Management System for the Collaborative Design, in A. Brown, M. Knight and P. Berridge (eds.), Architectural Computing from Turing to 2000 [eCAADe Conference Proceedings], Liverpool, pp. 364-369.