Invisible Sketch Interface in Architectural Engineering

Pierre Leclercq

LuciD Lab for User Cognition & Intelligent Design
University of Liège, Belgium
pierre.leclercq@ulg.ac.be
www.lema.ulg.ac.be/LuciD

Abstract. In this paper, we propose to discuss the concept of the "invisible interface", as an user interface compatible with the cognitive process involved in architectural sketching. We present the principles of such an interface, and illustrate them by our software prototype *EsQUIs*E.

1 The Sketch as a Design Tool

Sketches are widely used at the start of the design process by architects in building engineering. These drawings, initially abstract, gradually evolve into more geometrical representations of the desired object. Used at first to represent graphically the basic elements of the problem, they evolve towards more conventional representations of the project [16]. The sketch is used as a graphic simulation space: the basic elements of the project, set down in the earliest drawings, are progressively transformed until a complete solution to the problem is reached. Each sketch therefore represents an intermediate state between the first rough sketch and the definitive design solution (we have to underline that the sketches that we are dealing with here are "design drawings" rather than "presentation drawings" which are unconnected with the design process and only appear much later on, according to the Fraser and Henmi classification [6].

But the sketch is not simply an externalization of the designer's mental image [10,17, 18], it is also a heuristic field of exploration within which the designer discovers new interpretations in his or her own drawing, opening up an avenue to new perspectives for solutions. This fact explains the role played by the sketch in the search for solutions [1, 7, 8, 15].

The use of a sketch-based interface in a design assistance system should not be seen simply as an improvement to the interaction between user and machine, but as the means to integrate computer assistance into the very heart of the design process.

We have just seen that the sketch plays a major role in the designer's creativity. If we want to capture the project at the very moment of its conception without disturbing the course of the design process, the designer's freedom must not suffer the least hindrance. The problem of the interface is therefore a crucial one and concerns now a variety of approaches [2, 3, 4, 9].

In this article we will set out the necessary characteristics of such an user interface. Our research in this area over several years [11, 12, 14], as well as the develop-

J. Lladós and Y.-B. Kwon (Eds.): GREC 2003, LNCS 3088, pp. 353–363, 2004.

ment of our prototype EsQUIsE, has led us to specify various demands on the user interface of such systems.

We have called “invisible interface” an user interface demonstrating the four characteristics we consider essential for the early stages of a design: adaptability, naturalism, transparency and common-sense knowledge. This term expresses the fact that the system must fade completely into the background, and its presence must not be felt until the moment when the designer expressly requires its assistance. To understand fully the implications of the invisible interface we will look at the characteristics of EsQUIsE, a prototype application for the capture and interpretation of architectural sketches, in the second part of this article. In the third part we will define the invisible interface, then illustrate its characteristics in the functioning of our prototype.

2 The Sketching Prototype

EsQUIsE is a prototype application for interpretation of architectural sketches. Fully developed in Common Lisp, it captures the lines of an architectural sketch hand-drawn on a digital pad. It is then capable of deducing in real time the spaces enclosed within these lines and to associate them with characteristics appropriate to such places, by means of a character recognition module. The semantic model built up in this way is used to inform different evaluators about the performances of the building. EsQUIsE is made up of two modules that act consecutively (Fig. 1). The first, the entry module, captures then analyzes the graphic information in order to construct a geometrical model of the sketch. The second, the interpretation module, interprets the geometrical model according to the field of design in order to construct a functional model of the planned object, intended to provide appropriate information to various evaluators [13]. The figure 2 shows an example of a session screen copy.

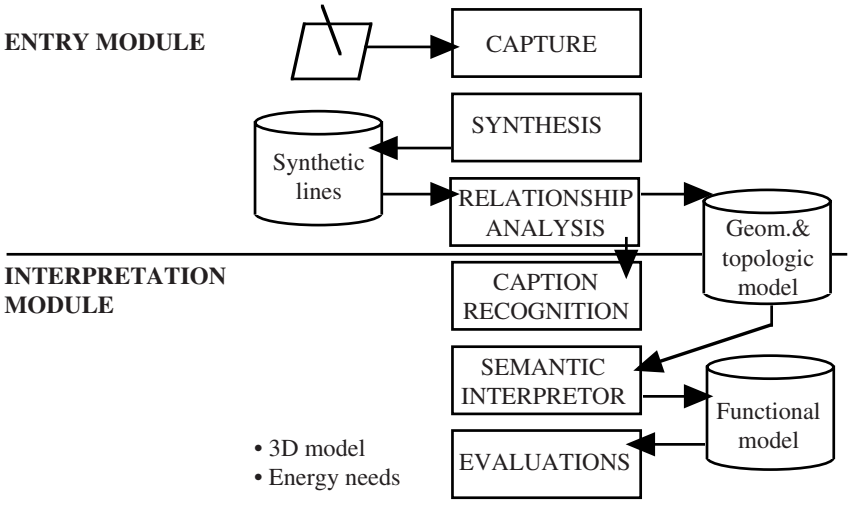

Fig. 1. EsQUIsE main modules

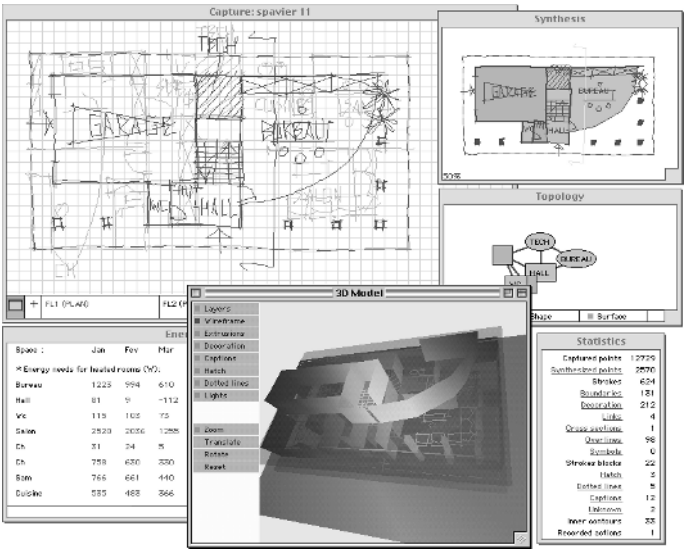

Fig. 2. Screen copy of the EsQUIsE prototype. Top-left image: capture of a hand-drawn sketch • Top-right: generation of the geometric and topologic models. Down-right: recognized items and 3D generated model • Down-left: evaluation of the model energy needs

2.1 The Entry Module

The role of the entry module consists in analyzing the drawing in order to construct the geometrical model of the sketch, in other words, the internal representation of the structure of the drawing: the significant graphic elements and the relationships they maintain.

The principal constraint on such a system is obviously the requirement that it should work in real time. Analysis therefore takes place in two phases. While the electronic stylus is being moved over the pad, the system captures the designer's movements. Then, as soon as a line is finished, the system takes advantage of the time lapse available before the start of the next line to run all the procedures to synthesize and analyze the layout.

Capture and Synthesis of Lines. The capture module receives the raw coordinates of the points relayed by the digital pad. It decomposes the sketch into lines, the first level of drawing recognition in our model. A line begins when the stylus is placed on the pad and ends when it is taken off. To limit the amount of information to be processed in later stages, an initial filtering of the data is carried out during the capture process.

The synthesis module consists of a series of successive filters intended to extract the essential characteristics of the lines, reducing by as much as possible the amount of information to be processed while conserving as faithfully as possible the appearance of the original line. To ensure that the sketch retains its ''triggering potential'',

this step is carried out transparently for the user, who only ever works on his or her initial drawing, unaware of any interpretation being made by the system.

Recognition of Captions and Symbols. Taking advantage of the fact that the synthesis module has coded the drawing, a caption and symbol recognition procedure is run as soon as a line has been synthesized. The user can thus characterize quite naturally the elements in his or her composition. Analysis of the relationships between the captions and the rest of the drawing enables the system to identify the element defined in this way. EsQUIsE associates the captions with the outline they belong to, enabling it to deduce the characteristics of the rooms.

Analysis of Relationships and Construction of the Geometrical Model. The aim of the analysis of the drawing is to weave the network of relationships between the different graphic objects it contains. Relationships include, for example, inclusion, intersection, proximity and superposition of lines and contours.

Because the sketch is imprecise, we have developed a "fuzzy graphics" approach that takes into account a considerable margin of error in the identification of points, lines and intersections. Outlines, for example, do not need to be fully closed-off in order to be recognized. By analyzing the proximity of the ends of the lines, EsQUIsE is able to identify an imprecise outline.

2.2 The Interpretation Module

The job of the interpretation module is to translate the geometrical information, produced by capturing the sketch, into a functional model of the planned architectural object.

This interpretation is highly dependent on the specific semantics of the design field: figure 3, for example, shows some of the links established by EsQUIsE's interpretation module between geometrical concepts (on the left) and architectural concepts (on the right).

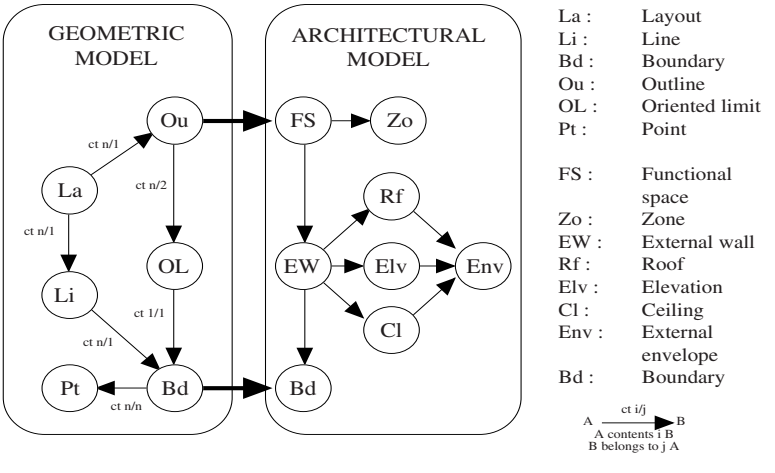

Fig. 3. Relationship between geometrical and functional models in EsQUIsE

We chose to represent the architectural model on a simplified structure diagram where the ordinate shows the level of abstraction at which the designer sees the problem (left part of Fig. 4). On the global level, the process evolves from highly abstract to more concrete, while, on the local level, the process evolves in a much less straightforward way: the designer sometimes decides to explore an idea in greater depth and at other times deciding to increase the level of abstraction so as to relax the constraints on the design.

In order to remain functional and relevant at every stage in the process, the model has to be able to adapt to these different levels of abstraction. It is therefore structured in several layers, each at a different level of abstraction.

The right part of figure 4 shows the three layers used in EsQUIsE's architectural model. In the center, the ''frontiers level'' is the model's first level of interpretation. It is deduced directly from the boundaries contained in the geometrical model. By analyzing the contact between the frontiers, the system constructs a more abstract representation of the building, made up of functional spaces and the adjacency relationships that they maintain. At the lowest - i.e. the most concrete - level there are the ''detailed frontiers'' that make up the model of the product, ready to be used in whatever ways are required during the production phase. This classification of information into different levels ensures that the model remains consistent throughout the process.

Because it is organized in layers with different levels of abstraction, the model built up in this way can support the different stages in the project being designed. This multilevel organization of the functional model also provides different access points for the project evaluators, according to their specific needs.

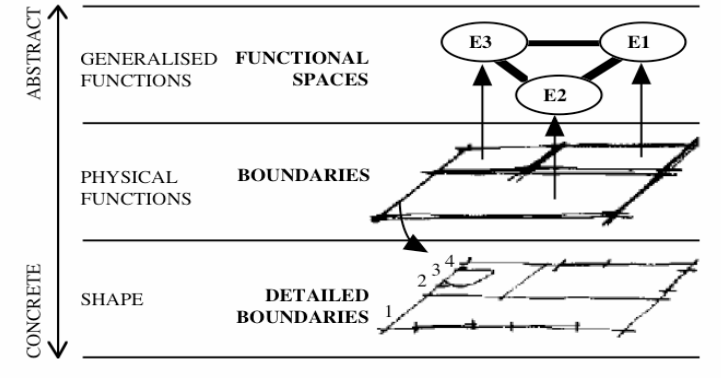

Fig. 4. The functional model of EsQUIsE, which supports the different levels of abstraction of the design process

For example, searching for similar compositions in a case base would be carried out on the basis of the topology; EsQUIsE can build the 3D architectural model with the boundaries level knowledge; whereas the "detailed frontiers level" would provide the measurements for standard CAD tools and for the energy needs evaluator.

3 The Concept of the Invisible Interface

The aim of the invisible interface is: understanding using the least possible means. It can be defined by its four characteristics: it is at the same time a adaptive, natural, transparent and intelligent interface. These terms are usually employed in the HCI domain; however, they recover different meanings according to author and context. So we suggest to define our own definitions of these terms, which we explain by the functioning of EsQUIsE.

3.1 An Adaptive System

Although every discipline uses more or less standardized graphic conventions, each architect has his or her own habits. The system must therefore be capable of supporting this unconventional dialogue mode by learning the designer's habits. For caption recognition, for example, EsQUIsE includes a learning module that builds up an alphabet for each user. In a more general way, we can say that the computer has to adapt to human behaviour, and more specifically to the fuzzy characteristics of sketches. As we saw in the previous summary of EsQUIsE, this step is managed by the synthesis module, which analyzes the hard line to build a computerized image of the project.

Figure 5 shows some examples of such mechanisms. The first window (top) shows the designer's original drawing and the second (bottom) shows the computerized version (you will recall that this second window is rendered invisible to the user who works only on his original line).

At first glance, we see that the synthesis module doesn't alter the general look of the drawing, which stays close to the original. However we see that some modifications have nevertheless been adopted. We also note that, apart from the synthesis of lines, designed to reduce the volume of computer data, the lines of the top-left corner of the rectangle has been extended, since the system considered the distance to the vertical line to be insignificant. Similarly, those parts of the lines jutting out above have not been considered to define the shape.

Unlike systems for the recognition of scanned images, the chronology of the drawing plays an essential role in our sketch recognition system. Each line is fitted into a pre-existing graphic context and is interpreted according to this order of appearance. When a designer draws two lines that are intended to meet, the contact point is never precisely positioned, the lines always being either a little too long or a little too short. It is therefore the chronology of the drawing that tells the system which line needs to be modified to preserve the consistency of the sketch. By combining chronological and geometrical information, the system can access a higher level of interpretation. For example, it can identify sequences such as captions, dotted lines, cross-hatching or blackening (figure 5), which are an aligned series of lines or graphic symbols that present the same characteristics.

For us, the adaptative interface is thus on the one hand able to adapt itself to the user himself or herself, by learning his or her working habits and, on the other hand, capable of dealing with the imprecision of human lines.

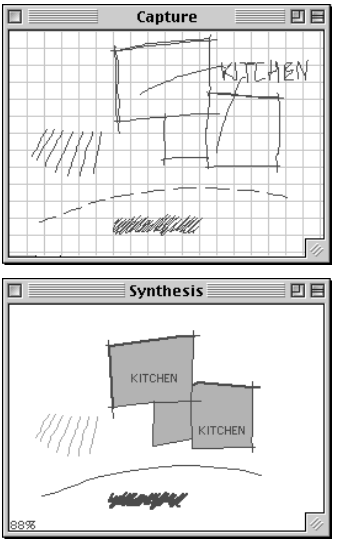

Fig. 5. Adaptive and Transparent interface in EsQUIsE

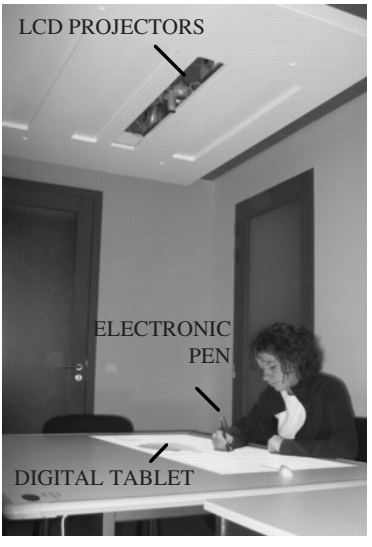

Fig. 6. The Natural interface. and the virtual desk

3.2 A Transparent Interface

A transparent interface means that the system does not require a pre-established dialogue procedure. One of the main arguments we have with a lot of sketch recognition systems is the fact that they impose a certain design method, which bears no relation to user's habits. In certain cases, the system imposes the symbols design order and, in others, the designer has to indicate when he or she is starting to draw a symbol and when it is finished.

In our system, the designer creates freely and the IT application monitors his or her actions. The context enables the system to identify the action carried out rather than the designer making use of a predefined function. A designer can thus take the tool in hand without any knowledge of its functioning. However, our system is not a recognition system suited for each user but is based on design habits peculiar to the architectural discipline. In EsQUIsE, the concept of transparent interface manifests itself in different ways. The first window (top) of figure 5, for example, shows a designer drawing sketched using EsQUIsE. The second window (bottom) shows the computerized image after synthesis. Although we note that the designer used only one colour and one thickness for his lines and doesn't give any instructions to the computer, EsQUIsE has been able to distinguish the various kinds of lines (the walls, the legends and their connecting lines) by examining chronological and geometrical relationships between user actions.

Using EsQUIsE does not require any use of the keyboard. The designer, who draws walls or writes captions, never specifies the significance of his or her actions. The system interprets the lines drawn on the pad in function of the context.

The transparent interface, perhaps more than the natural, is one of the necessary preconditions that enables a system to function at the conceptual design phase. Indeed, as we saw in the first part of this article, it is essential not to interrupt the design process by entering into dialogue with the computer.

3.3 A Natural Expression

All a designer needs to sketch a project is a piece of paper and a pencil. The aim of the natural interface is to conserve the simplicity of these tools, while at the same time achieving the same exceptional versatility.

Up to now, EsQUIsE has employed a screen pad (digital capture on an LCD screen) together with an electronic stylus as unique input device. This installation has already performed better than the traditional graphic tablet since in the latter case the parallax, which appears when a designer draws on the tablet and checks the on-screen results is eliminated. However, this system still had some drawbacks of which the most significant was undoubtedly the tablet's dimensions (18"), which limited the sketches' format and scale.

To remedy this situation, we are currently in the process of implementing a ''virtual desktop'' enabling the handling of much larger documents. In addition to this improvement, the virtual desktop extends the analogy to traditional working method by enabling multi-document handling on the desk (photographs, previous designs, etc.). We have carried out a prototype of the desk made up of a video projector with a mirror, enabling the screening of computer displays on a digital A0 tablet or on a 56" picture (Fig. 6).

Even though no scientific study of user behaviour in this new installation has been yet carried out, since this is still a prototype, we can however note two interesting results:

- the user's handle seems much more natural. Even though he or she mistrusts the apparently fragile screen pads, he or she soon feels comfortable on the virtual desktop;
- even though the desk's resolution is much lower than that of the screen pad, since it is limited by the projection, the system acquires details to which it didn't have access, as the scale chosen by the user is much greater.

Getting the most natural interface is obviously the primary aim of every pen computing system. However, current systems are far from being in competition with the traditional pen and paper. Carrying out a good natural interface involves giving consideration to both hardware and software parameters.

3.4 An Intelligent Interface Supplied with an Common-Sense Knowledge

A sketch is, by definition, incomplete. The designer only uses it to represent essential information, that which is specific to the current project. He or she focuses on certain problems in succession, postponing any decisions concerning other elements of the design. It is therefore quite common to come across one element that is fully defined in both shape and dimensions when the rest of the drawing is still very sketchily drawn. This way of working enables the designer to deal with the complexity of the

project, going by his or her own experience to hierarchize the sub-problems that need to be resolved. The designer is only able to work in this way because he or she knows that the information that is not directly focused on is not going to cause difficulties later, or at least is only going to have a limited influence on the element being designed [11].

To fill in the information not specified by the designer, the system must be able to identify the context of the design being carried out. It is therefore capable of selecting the most relevant information in function of this context, rather than blindly settingstandard values for all its parameters. In order to feed appropriate information to the different evaluators while the project is still at the gestation stage, the system must be assisted by an implicit database specific to the particular field of design.

This omitted information must therefore be included in order to make up the functional model. By sharing this common-sense knowledge with the designer, the system can assign appropriate parameters to a design element well before these data are explicitly indicated – or even considered – by the designer. The use of this.implicit knowledge enables the system to construct a sufficiently complete model very early in the design process. The system adapts its data as and when the successive sketches are drawn, i.e. as the designer's model becomes increasingly precise.

In architecture, for example, the designer may draw a room and put a window in it. The software would search in its database and assign his/her usual sill height to the window. The designer might go on to call the newly created space ''bath room''. The software would consult its database and revise its decision, assigning the window a higher sill height, which would be more appropriate to the intimacy of its newly designated function. We organized this implicit database, till now by a direct coding in three hierarchically layers (Figure 7). The first layer consists of the designer's personal references: previous projects and design habits. The second layer is made up of the rules of good practice, European standards, norms, etc. Finally, the third layer contains universal references, which are independent of any context: characteristics of materials, density, conductivity, strength, etc.

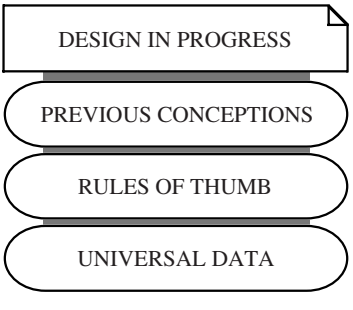

Fig. 7. Implicit knowledge of EsQUIsE

Because it is organized in layers with different levels of abstraction, the model built up in this way can support the different stages in the project being designed. Assisted by the implicit database, the system is capable of maintaining the consistency of the model, despite the incompleteness of the information it receives.

4 Conclusion – Summary

Our experience of designing sketch-based tools for architectural design support has led us to set out various demands on the user interface of such systems. It must be:

- adaptive (adapt its behaviour to the user)
- transparent (not impose a fixed dialogue protocol)
- natural(not change the habits of the designer)
- supplied with a common-sense knowledge (able to choose pertinent information according to context).

In our opinion, these four characteristics of the invisible interface are the necessary preconditions that enable a system to function at the conceptual design phase. The EsQUIsE prototype, which was developed according to these principles, has demonstrated the validity of such an approach in the field of architectural design. EsQUIsE works in two step phases. In the first step, it constructs the geometrical model of the sketch by detecting the relationships between the different elements that make up the drawing. Because this step is independent of any semantics specific to a particular field, it can be adapted to any discipline. Next, the interpretation module analyzes the geometrical model that has been built up, in order to give meaning to the sketch and constructs the semantic model of the architectural project. Thanks to an implicit knowledge base, belonging to each design discipline, this model could then provide very effective assistance to the design process because of the pertinent way it represents the object.

References

1. Ah-Soon C., & Tombre K. (1997). Variations on the Analysis of Architectural Drawings. *Proc. 4th International Conference on Document Analysis and Recognitio*n, pp. 347–351. Ulm, Germany.
2. Alvarado C., & Davis R. (2001a). Resolving ambiguities to create a natural sketch based interface. *Proc. IJCAI-200*1, pp. 1365–1371.
3. Alvarado C., & Davis R. (2001b). Preserving the freedom of paper in a computer-based sketch tool. *Proc. of HCI International 200*1, pp. 687–691.
4. Do, E. (1998). *The Right Tool at the Right Time - Investigation of Freehand Drawing as an Interface to Knowledge Based Design Tool*s. Ph.D., Georgia Institute of Technology, USA.
5. Forbus, K., & Usher, J. (2002). Sketching for knowledge capture: a progress report. *IUI'02, California, USA.*
6. Fraser, I., & Henmi, R. (1994). *Envisioning Architecture: an analysis of drawin*g. Van Nostrand Reinhold. NY.
7. Goel, V. (1995). *Sketches of though*t. MIT Press, Cambridge, MA.
8. Goldschmidt, G. (1991). The dialectics of sketching. *Design Studies vol.* 4, pp. 123–143.
9. Landay J. (1996a). SILK: Sketching Interfaces Like Krazy. *Proc. of Human Factors in Computing Systems (Conference Companion), ACM CHI '9*6, pp. 398–399. Formal video program.

10. Lebahar, J.CH. (1983). *Le dessin d'architecte – Simulation graphique et réduction d'incertitude*. Éditions Parenthèses, Paris.
11. Leclercq, P. (1994). *Environnement de conception architecturale pré-intégré. Éléments d'une plate-forme d'assistance basée sur une représentation sémantiqu*e. Doctoral thesis in applied sciences, Faculty of applied sciences, LEMA, University of Liège, Belgium.
12. Leclercq, P. (1999). Interpretative tool for architectural sketches. In *Visual and Spatial Reasoning in Desig*n, (Gero, J., & Tversky, B., Eds.), pp. 69–80. Key Centre of Design Computing and Cognition, Sydney, Australia.
13. Leclercq, P. (2001). Programming and Assisted Sketching. *Proc. Ninth Int Conf. CAAD Futures 2001*, (de Vries, B. et al., Eds.), pp 15–32. Kluwer Academic Publishers, Dordrecht, The Netherlands.
14. Leclercq, P., Juchmes, R., (2002). The invisible Interface in Design Engneering, in AIEDAM, Artificial Intelligence in Engineering Design & Manufacturing, Special Issue: Human-computer Interaction in Engineering Contexts, Vol.16, No. 5, Nov 2002, Cambridge University Press. (original and complete paper, synthesized here for GREC'03).
15. Mathus, P. (1994). *Analyse d'esquisses architecturales*. Thesis in architectengineering, LEMA, University of Liège, Belgium.
16. Mccall, R., Ekaterini, V., & Zabel, J. (2001). Conceptual Design as Hypersketching. *Proc. Ninth Int Conf. CAAD Futures 2001*, pp. 285–298. Kluwer Academic Publishers, Dordrecht, The Netherlands.
17. Rassmussen J. (1990). Mental models and the control of action in complexenvironments. In *Mental models and human-computer interaction 1* (Ackerman D., & Tauber M.J. Eds.), Elsevier Science Publisher B.V., Holland.
18. Suwa, M., & Tversky, B. (1996). What Architects See in Their Sketches: Implications for Design Tools. *Proc. CHI'96 Conference on Human Factors in Computing Systems*, pp. 191–192. Vancouver.