Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2002), *16*, 219–227. Printed in the USA.

DOI: 10.1017/S0890060402163074

The absent interface in design engineering

PIERRE LECLERCQ AND ROLAND JUCHMES
Laboratoire d'Etudes Méthodologiques Architecturales (LEMA), Université de Liège, 4000 Liège, Belgium
(RECEIVED September 24, 2001; ACCEPTED April 12, 2002)

Abstract

In order to respond to the difficulties encountered by CAD software applications in really assisting the conceptual designer, we propose a tool that is capable of interpreting design sketches and feeding data to various project evaluators, right from the early phases in the design process. For that purpose, we use the concept of the absent interface, which is the only interface that is compatible with the cognitive process involved in sketching. In this paper, we present the principles of such an interface, illustrated by EsQUIsE, a software prototype for capturing and interpreting architectural sketches, which has been under development for several years.

Keywords: Sketch Interface; Integrated Computer-aided Design System; Virtual Modeling in Building Engineering; Knowledge Engineering and Implicit Data; Adaptative Interface; Interface Ergonomics

1. INTRODUCTION

1.1. Context

There are many powerful design assistance tools currently available in all fields of engineering, from drawing aids right up to the most sophisticated performance evaluators. However, it has to be observed that these tools fail to really help the designer in the initial design phases, in other words, at the moment when the broad outlines of the project are defined and the decisive options are chosen (see, for example, Gross [1996], Suwa & Tversky [1996], or Aliakseyeu & Martens [2001]).

We see the cause of this failure as being a problem of user interface: These tools require painstaking coding of precise data, which is only possible once the project has largely been defined. Their diagnosis only comes into effect after the costs of modifying or readapting the project have already become very high. This situation is even more critical in architecture where, with each building being effectively a unique product, the costs of correcting defects are considerable.

Reprint requests to: P. Leclerq, LEMA, Université de Liège, Chemin des Chevreuils 1, Bat B52/3, 4000 Liège, Belgium. E-mail: Pierre.Leclercq@ulg.ac.be

To remedy this situation, we proposed, in Leclercq (1994), use of the sketch to capture and evaluate the architectural project. Mathus (1994) effectively demonstrated and assessed the potential of the sketch as a means of capturing the semantics of an architectural project.

1.2. Characteristics of the sketch

Sketches are widely used at the start of the design process by designers in all fields. These drawings, initially highly 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 toward more conventional representations of the project, allowing the designer to transfer the information to other people involved.

The sketch is thus used as a graphic simulation space (Lebahar, 1983): 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.

The sketches that we are dealing with here are "design drawings," according to the Fraser and Henmi (1994) classification, rather than "presentation drawings," which are unconnected with the design process and only appear much later on.

Why do designers still prefer the hand-drawn sketch to computer-assisted design (CAD) tools at the start of the design process? McCall et al. (2001) sees three essential differences that explain the use of the hand-drawn sketch:

1. It is abstract and ambiguous. For this reason, it is well suited to the undeveloped state of the project at the sketch stage.
2. It is a nondestructive process in which the successive drawings are progressively transformed until the final solution is reached, whereas CAD tools are used to produce objects that can be manipulated (modification, destruction, etc.)
3. Finally, sketching produces a wide collection of interrelated drawings whereas CAD systems construct a single model, which is isolated from the process that brought it about.

By themselves, however, the characteristics of hand sketching do not fully explain the part played by the sketch in the early phases of the design process. In effect, the sketch is not simply an externalization of the designer's mental image, 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 phenomenon, which has been widely researched ("lateral transformation," according to Goel, 1995, or "seeing as" according to Goldschmidt, 1991), explains the role played by the sketch in the search for solutions.

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 you 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.

In this article we set out the necessary characteristics of such a user interface. Our research in this area over several years, as well as the development of our prototype system, EsQUIsE, has led us to specify various demands on the user interface of such systems. What we have called an "absent interface" is a user interface demonstrating the four characteristics we consider essential for the early stages of a design: naturalism, transparency, adaptability, and intelligence. 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 fully understand the implications of the absent interface, we 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 define the absent interface and then illustrate its characteristics in the functioning of our prototype.

2. THE SKETCHING INTERFACE

2.1. A short summary of EsQUIsE

EsQUIsE is a prototype application for the interpretation of architectural sketches. Developed in Common Lisp, it captures the lines of an architectural sketch hand drawn on a graphic 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. Figure 1 shows the screen of the EsQUIsE prototype.

EsQUIsE is made up of two modules that act consecutively. The first, the data entry module, captures and 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, which is intended to provide appropriate information to various evaluators.

Figure 2 shows the procedural structure of the EsQUIsE prototype. We now examine in more detail each of the processes in the chronological order in which they are applied.

2.1.1. The data entry module

The role of the data entry module consists of 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, consisting of the significant graphic elements and the relationships they maintain. Two types of act can be distinguished: capture and synthesis of the lines, which synthesize and decompose the lines from the raw data relayed by the graphic pad; and analysis of the drawing, which works out the spatial relationships between the graphic objects found on the pad (contact, adjacency, etc.). 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 graphic 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

https://doi.org/10.1017/S0890060402163074 Published online by Cambridge University Press

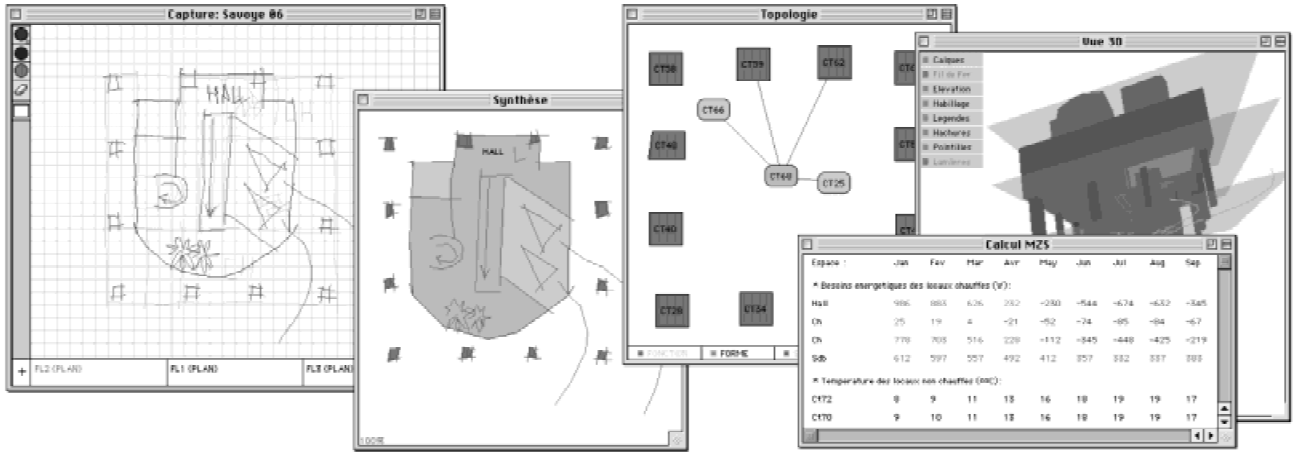

Fig. 1. A photograph of the screen of the EsQUIsE prototype. The left image is a capture of a hand-drawn sketch. On the right are the generation of the 3-D model and energy needs. In the center are geometrical and topological models created by EsQUIsE.

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 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 simplified 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. In EsQUIsE, for example, topological analysis associates the captions with the outline to which they belong, 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, parallelism, inclusion, intersection, proximity, and superposition of lines.

Because the sketch is imprecise, we have defined 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.

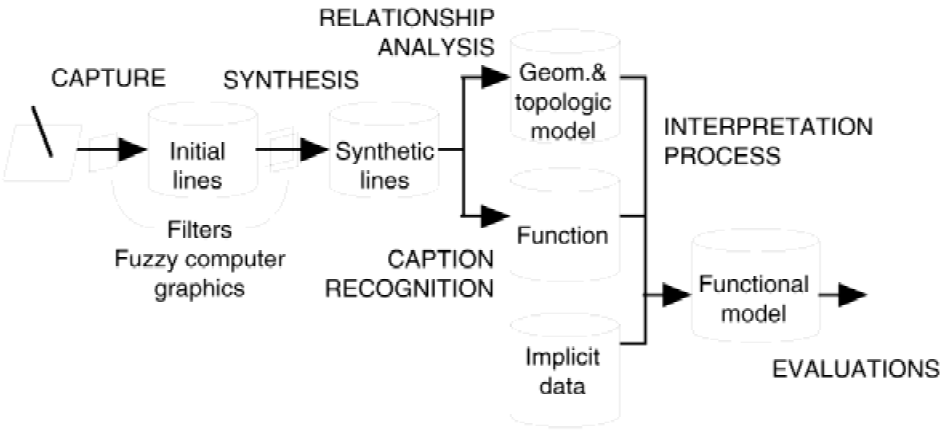

Fig. 2. The procedural structure of the EsQUIsE prototype.

https://doi.org/10.1017/S0890060402163074 Published online by Cambridge University Press

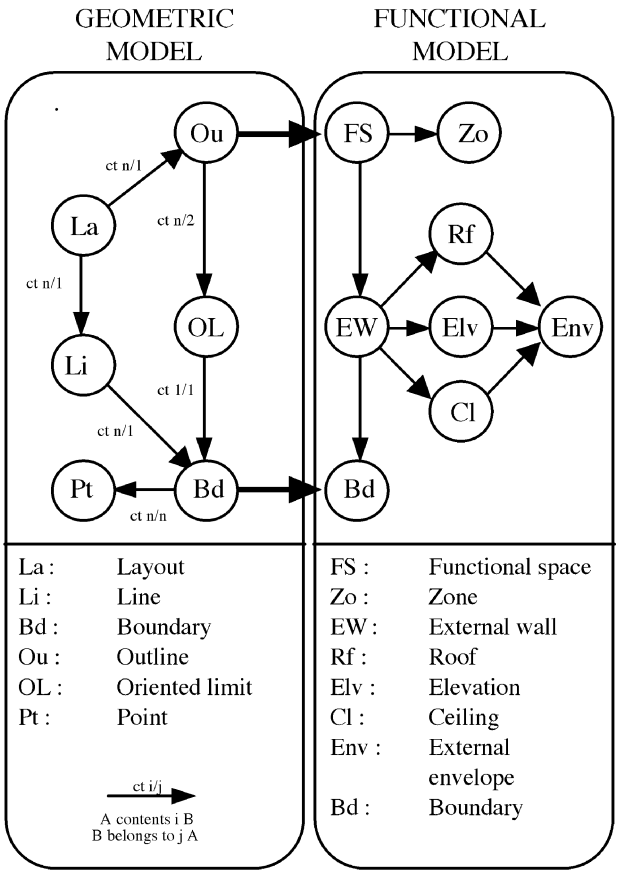

Fig. 3. The relationship between the geometrical and functional models in EsQUIsE.

2.1.2. 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 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 and functional, in this case architectural, concepts.

Inspired by Rassmussen's (1990) model, we chose to represent the process on a simplified structure diagram. The abscissa shows the level of decomposition of the planned object, and the ordinate shows the level of abstraction at which the designer sees the problem (Fig. 4, left). On the global level, the process evolves from highly abstract to more concrete, whereas on the local level, the process evolves in a much less straightforward way with the designer sometimes deciding 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

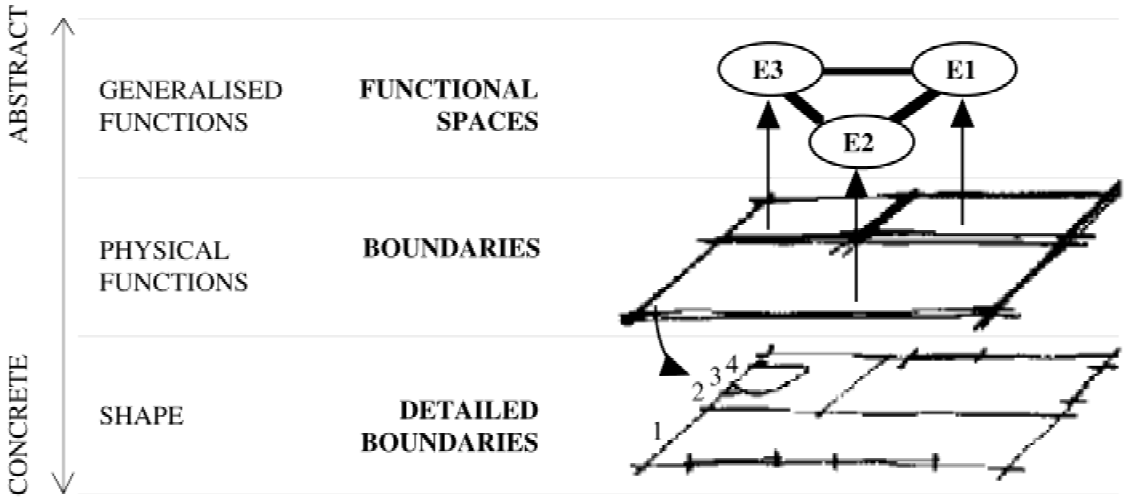

Fig. 4. The left section shows a simplified diagram of the process structure. The right section shows the functional model of EsQUIsE, which supports the different levels of abstraction of the design process.

https://doi.org/10.1017/S0890060402163074 Published online by Cambridge University Press

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. For example, searching for similar compositions in a case base would be carried out on the basis of the topology, whereas the detailed frontiers level would provide the measurements for standard CAD models. See Leclercq (1998, 1999, 2001) for more detailed descriptions of this prototype: or visit our web page at http://www.lema.ulg.ac.be/esquise.

2.2. Related works

Since the early 1960s various researchers have looked into sketching interfaces. Apart from those systems "with 'sketch' in their names," Do (1998) distinguishes two categories: systems that convert sketches to objects and systems that support the sketching activity. It is clearly those systems from the second category that interest us the most here. The SILK system, for example, described in Landay and Myers (1995) and Landay (1996*a*, 1996*b*), allows for the simple and speedy creation of user interfaces through the sketching by hand of interface objects (buttons, dialog boxes, etc.) and then by allowing the user to interact with these objects.

In the architectural world the Electronic Cocktail Napkin and other associated applications (e.g., Gross, 1994, 1996; Do & Gross, 1996) allow recognition and interpretation of diagrams and sketches. This system is used, for example, in easily accessing case bases (Gross et al., 1994).

These systems operate in a very different way from ours because they are based on the recognition of predefined shapes or gestures and consequently analyze the relationships between these elements. EsQUIsE employs similar elements for the recognition of legends or symbols, but this procedure is not applicable for the interpretation of architectural plans because the design is composed of more than just symbols. An architect cannot effectively build a plan by starting with one room, then proceeding to the next, then adding a door to separate the rooms, and so on. For example, he or she will trace a long line to depict an axis and then retrace certain parts of the line, thus signifying that these parts relate to the walls, or it can serve as a support when designing a door. In short, those systems based on the recognition of symbols are well suited to the recognition of diagrams (which are another type of graphic representation used by designers at the start of a project), whereas we are interested instead in the interpretation of plans.

The ASSIST prototype described in Alvadaro and Davis (2001*a*, 2001*b*) offers a sketching interface for the capture of mechanical sketches, which can then be converted into CAD in order to simulate the behavior of a given mechanism.

The common point between the ASSIST system and ours is the fact that they both have their own knowledge base to simulate the behavior of the object under design. The use of sketching, however, is limited in the former case to the natural acquisition of information. The system "corrects" the design, thereby abandoning the imprecision and ambiguity needed during the design process.

Saund and Moran's (1994) Perceptually Supported Sketch Editor (PerSketch) system interprets an image by making a variety of shapes appear, rather than analyzing relationships between symbols. This system can access knowledge independent of the symbols' construction order but cannot model the object under design because it analyzes only shapes.

In a different domain, EsQUIsE could also be related to the tools for recognition for architectural drawing. Dosch et al. (2000) and Ah-Soon and Tombre (1997, 2001) analyze scanned building plans and are thus able to construct a 3-dimensional (3-D) model. This is not, however, a design aid, because it applies solely to precise CAD drawings.

The HDICAD system described by Lladós et al. (1997, 1998) can also interpret architectural plans but is nevertheless able to support hand drawing imprecision. As in the previous system, it simply analyzes drawings from scanned images off-line.

The prototypes we have just described do not reach the objectives we set out for EsQUIsE. Indeed, they each have interesting characteristics and functions, but we consider none of them to be well suited to assist the designer in a real work situation.

In the rest of this article, we define the characteristics that any tool capable of being employed during the early stages of designs should possess and show how we attempted to apply these principles to EsQUIsE.

3. THE CONCEPT OF THE ABSENT INTERFACE

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

3.1. A natural interface

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 a LCD screen) together with an electronic stylus as a unique input device. This installation has already performed better than the traditional graphic tablet because 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, which limited the sketches' format and scale.

https://doi.org/10.1017/S0890060402163074 Published online by Cambridge University Press

To remedy this situation, we are currently in the process of implementing a "virtual desk," enabling the handling of much larger documents. In addition to this improvement, the virtual desk extends the analogy to traditional working methods by enabling multidocument handling on the desk (photographs, previous designs, etc.).

We have carried out a first simplified prototype of the desk made up of a video projector with a mirror, enabling the screening of computer displays on a traditional desk, and an infrared and ultrasound positioning system for the stylus (Fig. 5).

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

1. 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 desk.
2. Even though the desk's resolution is much lower than that of the screen pad, because it is limited by the projection, the system acquires details to which it did not have access, as the scale chosen by the user is much greater.

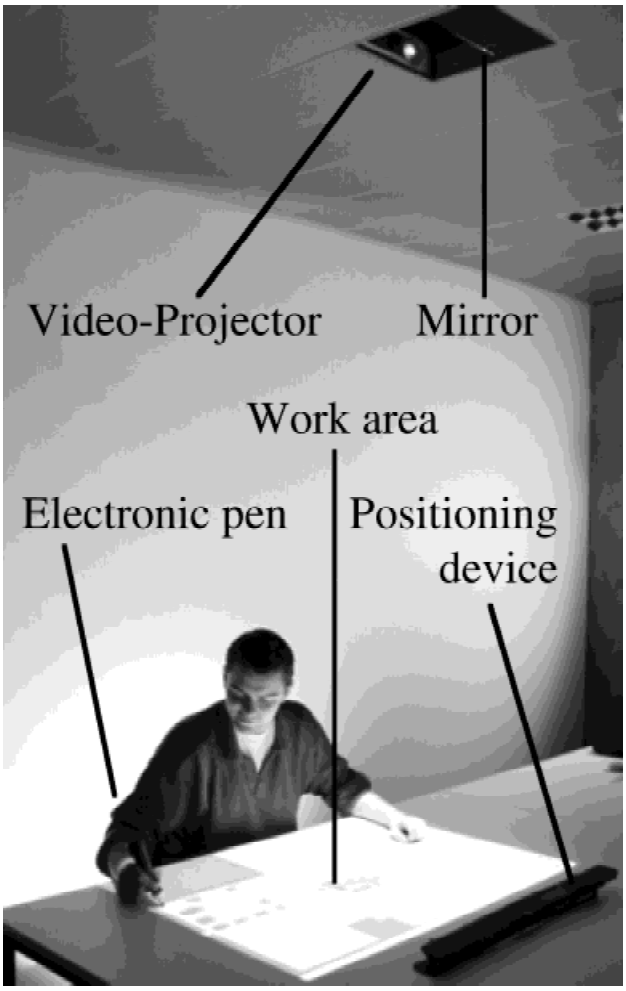

Fig. 5. The natural interface in EsQUIsE.

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.

3.2. A transparent interface

A transparent interface means that the system does not require a preestablished dialogue procedure. It is the machine that should adapt to human behavior, rather than the other way around.

One of the main arguments we have with many sketch recognition systems is the fact that they impose a certain design method, which bears no relation to the user's habits. In certain cases, the system imposes the symbols' design order, whereas in others the designer has to indicate when he or she is starting to draw a symbol and when it is finished (e.g., Forbus & Usher, 2002).

In our system, the designer creates freely and the information technology (IT) application monitors his or her actions. This is the context that enables the system to identify the action being carried out rather than requiring the designer to make 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 each discipline. Indeed, each discipline has its own habits and standards of design: architects and engineers, for example, do not construct their plans in the same way. Therefore, the software has to take into account these differences. Up to now, our studies have mainly focused on architectural design, but we are thinking of adapting EsQUIsE's functions to other disciplines in the near future.

In EsQUIsE, the concept of transparent interface manifests itself in different ways. The first window of Figure 6, for example, shows a designer drawing sketches using EsQUIsE. The second window shows the computerized image after synthesis. Although we note that the designer used only one color for his lines and does not give any instructions to the computer, EsQUIsE has been able to distinguish the various kinds of lines (the walls, the legend, and its connecting line) by examining chronological and geometrical relationships between lines. Using EsQUIsE does not require any use of the keyboard. The designer, who draws walls or furniture or writes captions, never specifies the significance of his or her actions. The system interprets the lines drawn on the pad as a function of the context.

The transparent interface, perhaps more than the natural interface, is one of the necessary preconditions that enables

https://doi.org/10.1017/S0890060402163074 Published online by Cambridge University Press

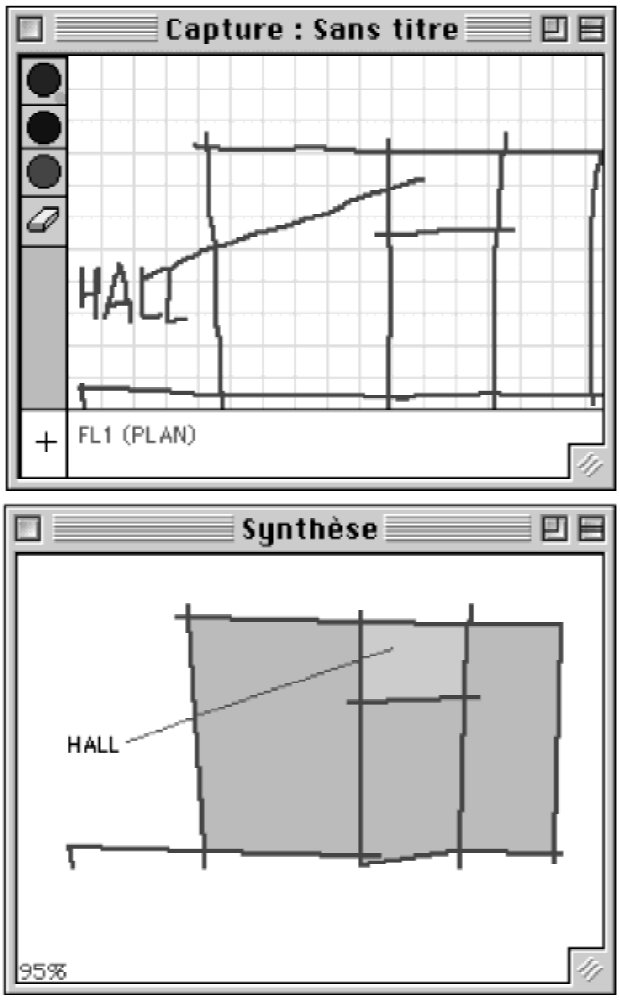

Fig. 6. The transparent interface in EsQUIsE.

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. An adaptive interface

Although every discipline uses more or less standardized graphic conventions, each designer has individual 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 behavior, and more specifically, to the fuzzy characteristics of sketches. As we saw in the 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 7 shows some examples of such mechanisms. The first window shows the designer's original drawing and the second shows the computerized version. You will recall that this second window is rendered invisible to the user, who works only on the original line.

At first glance, we see that the synthesis module does not alter the general look of the drawing, which stays close to the original. However, some modifications have nevertheless been adopted. Moreover, apart from the synthesis of lines, which is designed to reduce the volume of computer data, the horizontal line in the center of the window has been extended, because the system considered the distance to the vertical line to be insignificant. Similarly, those parts of the lines jutting out above have been deleted.

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 preexisting 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

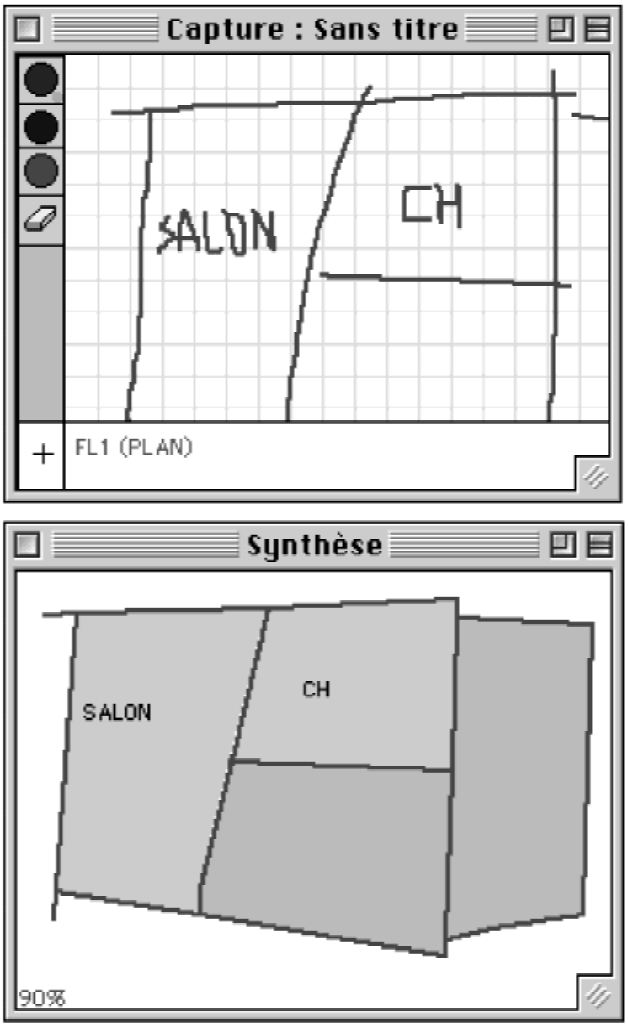

Fig. 7. The adaptative interface in EsQUIsE.

https://doi.org/10.1017/S0890060402163074 Published online by Cambridge University Press

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. In accordance with the designer's intention, it is generally the second line that is modified because that is the line that has to fit into an already existing drawing.

By combining chronological and geometrical information, the system can access a higher level of interpretation. For example, it can identify sequences such as words, dotted lines, or cross-hatching, which are an aligned series of lines or graphic symbols that have the same characteristics.

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

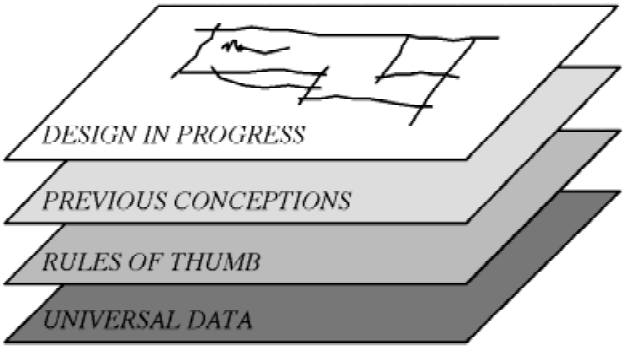

Fig. 8. The intelligent interface in EsQUIsE.

3.4. An intelligent interface

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 for the function of this context, rather than blindly setting standard values for all its parameters. In order to feed appropriate information to the different evaluators while the project is still in the gestation stage, the system must be assisted by an implicit database specific to the particular field of design.

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 subproblems 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 (Leclercq, 1994).

This omitted information must therefore be included in order to make up the functional model. By sharing this implicit 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, that is, 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 a standard sill height to the window. The designer might go on to call the newly created space "bathroom." 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 database in three hierarchically superimposed layers (Fig. 8). The first layer consists of the designer's personal references: previous projects and design habits. The database is obviously made more effective through the system's knowledge of the designer and his or her strategies for resolving problems. The second layer is made up of the rules of good practice, European standards, norms, and so on. Finally, the third layer contains universal references, which are independent of any context: characteristics of materials, density, conductivity, strength, and so forth.

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

Our experience of designing sketch-based tools for design support has led us to set out various demands on the user interface of such systems. It must be natural and not change the habits of the designer, transparent and not impose a fixed dialogue protocol, adaptive and adapt its behavior to the user, and intelligent and able to choose pertinent information according to context. In our opinion, these four characteristics of the absent 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 steps. 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 seman-

https://doi.org/10.1017/S0890060402163074 Published online by Cambridge University Press

tics specific to a particular field, it can be adapted to any discipline. In the second step 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 project. This model can then provide very effective assistance to the design process because of the pertinent way it represents the object. This step is only feasible thanks to an implicit knowledge base belonging to each design discipline.

To follow on from this, we are offering to apply the concept of the absent interface to other disciplines. We are currently preparing a research project with a view to applying the sketch interpretation technique to the field of mechanics.

REFERENCES

Ah-Soon, C., & Tombre, K. (1997). Variations on the analysis of architectural drawings. *Proc. 4th Int. Conf. Document Analysis and Recognition*, Ulm, Germany, pp. 347–351.

Ah-Soon, C., & Tombre, K. (2001). Architectural symbol recognition using a network of constraints. *Pattern Recognition Letters 22*, 231–248.

Aliakseyeu, D., & Martens, J.-B. (2001). Physical paper as the user interface for an architectural design tool. *Proc. Interact 2001*, Tokyo, pp. 680–681.

Alvarado, C., & Davis, R. (2001*a*). Preserving the freedom of paper in a computer-based sketch tool. *Proc. HCI International 2001*, pp. 687–691.

Alvarado, C., & Davis, R. (2001*b*). Resolving ambiguities to create a natural sketch based interface. *Proc. IJCAI-2001*, pp. 1365–1371.

Do, E. (1998). *The Right Tool at the Right Time—Investigation of Freehand Drawing as an Interface to Knowledge Based Design Tools*. PhD Thesis. Atlanta, GA: Georgia Institute of Technology.

Do, E., & Gross, M. (1996). Drawing as a means to design reasoning. In *Visual Representation, Reasoning and Interaction in Design: Workshop Notes, Artificial Intelligence in Design '96 (AID '96)*, Stanford University, Stanford, CA.

Dosch, P., Tombre, K., Ah-Soon, C., & Masini, G. (2000). A complete system for analysis of architectural drawings. *International Journal on Document Analysis and Recognition 3*, 102–116.

Forbus, K., & Usher, J. (2002). Sketching for knowledge capture: A progress report. *IUI'02*, January 13–16, San Francisco, CA.

Fraser, I., & Henmi, R. (1994). *Envisioning Architecture: An Analysis of Drawing*. New York: Van Nostrand Reinhold.

Goel, V. (1995). *Sketches of Thought*. Cambridge, MA: MIT Press.

Goldschmidt, G. (1991). The dialectics of sketching. *Design Studies 4*, 123–143.

Gross, M. (1994). Recognizing and interpreting diagrams in design. In *Advanced Visual Interfaces '94: AVI '94* (Catarci, T., Costabile, M., Levialdi, S., & Santucci, G., Eds.). New York: ACM Press.

Gross, M. (1996). The Electronic Cocktail Napkin—Working with diagrams. *Design Studies 17*, 53–69.

Gross M., Zimring C., & Do, E. (1994). Using diagrams to access a case base of architectural designs. In *Artificial Intelligence in Design AID '94* (Gero, J.S., Ed.), pp. 129–144. Dordrecht: Kluwer.

Landay, J., & Myers, B. (1995). Interactive sketching for the early stages of user interface design. *Proc. Human Factors in Computing Systems, ACM CHI '95*, pp. 43–50.

Landay, J. (1996*a*). *Interactive sketching for the early stages of user interface design*. PhD Thesis. Pittsburgh, PA: Carnegie Mellon University.

Landay, J. (1996*b*). SILK: Sketching interfaces like krazy. *Proc. Human Factors in Computing Systems (Conf. Companion), ACM CHI '96*, pp. 398–399. [Video program].

Lebahar, J.C. (1983). *Le Dessin d'Architecte—Simulation Graphique et Réduction d'Incertitude*. Paris: Éditions Parenthèses.

Leclercq, P. (1994). *Environnement de conception architecturale préintégré. Éléments d'une plate-forme d'assistance basée sur une représentation sémantique*. PhD Thesis. Liège, Belgium: University of Liège.

Leclercq, P. (1998). Application d'un outil d'interprétation architecturale. *La Lettre de l'IA, Number 135, Complex Systems, Intelligent Systems and Interfaces*. Paris: EC2 & Développement.

Leclercq, P. (1999). Interpretative tool for architectural sketches. In *Visual and Spatial Reasoning in Design* (Gero, J., & Tversky, B., Eds.), pp. 69–80. Sydney, Australia: Key Centre of Design Computing and Cognition.

Leclercq, P. (2001). Programming and assisted sketching. *Proc. Ninth Int. Conf. CAAD Futures 2001* (de Vries, B., Leeuwen, J., & Achten, H., Eds.), pp 15–32. Dordrecht: Kluwer.

Lladós, J., Sánchez, G., & Martí, E. (1997). A system to understand hand-drawn floor plans using subgraph isomorphism and Hough transform. *Machine Vision and Applications 10*, 150–158.

Lladós, J., Sánchez, G., & Martí, E. (1998). A string-based method to recognize symbols and structural textures in architectural plans. In *Graphics Recognition, Algorithms and Systems* (Tombre, K., & Chhabra, A.K., Eds.), pp. 91–103. Lecture Notes in Computer Science 1389. New York: Springer–Verlag.

Mathus, P. (1994). *Analyse d'esquisses architecturales*. PhD Thesis. Liège, Belgium: University of Liège.

Mccall, R., Ekaterini, V., & Zabel, J. (2001). Conceptual design as hyper-sketching. *Proc. Ninth Int. Conf. CAAD Futures 2001* (de Vries, B., Leeuwen, J., & Achten, H., Eds.), pp. 285–298. Dordrecht: Kluwer.

Rassmussen, J. (1990). Mental models and the control of action in complex environments. In *Mental Models and Human–Computer Interaction 1* (Ackerman, D., & Tauber, M.J., Eds.). Amsterdam: Elsevier.

Saund, E., & Moran, T.P. (1994). A perceptually-supported sketch editor. *ACM Symposium on User Interface Software and Technology*, Marina del Rey, CA.

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*, Vancouver, Canada, 191–192.

Pierre Leclercq received his MScEng degree in architectural engineering (Ingénieur Civil Architecte) in 1987, and his PhD degree in Applied Sciences from the University of Liège in 1994. From 1988 to 1994 he was researcher and assistant of Prof. A. Dupagne in the Laboratory of Architectural Methodology, University of Liège. He is currently a professor in the Department of Architecture at the University Liège. In 2001 he created the Group Laboratory of User's Cognition and Intelligent Design (LuciD). Dr. Leclercq's current research interests are in design process analysis, implicit knowledge management, the sketch interface, and fuzzy computer graphics.

Roland Juchmes received his MScEng degree in architectural engineering (Ingénieur Civil Architecte) in 1999 from the University of Liège. He is currently a member of the LuciD Group. His research interests include sketch-based interfaces and CAD tools for early architectural design.