Nurturing Design Intuition in Energy Software

G. Z. Brown and Barbara-Jo Novitski
Department of Architecture
University of Oregon
Eugene, Oregon 97403

Abstract
This paper discusses educational software under development at the University of Oregon which helps to alleviate the difficulties of integrating technical considerations in the creative architectural design process.

Introduction
Design fields are arguably the academic disciplines in which art and science are the most closely married. Technical and social issues must be rigorously studied with left-brain attention to detail, while the right brain must be allowed flights of imagination and creativity. Frequently, architectural education has coped with the difficulties of integrating these widely different needs by teaching them separately, unintentionally encouraging students to view them as incompatible. The computer has found its niche in architecture schools over the past ten years squarely in the realm of the left brain. The vast majority of software for designers, whether analytical or presentational in purpose, requires as input a building which has already been designed, or at least developed to the point that it can be reduced to a set of clearly understood numbers or lines. This exacerbates the schism between technical considerations and the creative process because it requires users to pick up lock, stock, and media and move to a new work environment in order to change from a designing mode to an evaluating mode.

What is needed is a computer environment that allows users to sketch ideas in ways that encourage experimentation and imagination and that also lead, without cognitive interruption or a sharp break in work habits, to technical evaluation. This will tend not only to merge the left- and right-brain activities, but it will also enable designers to consider "technical" questions earlier in their design process than they otherwise could. In the case of energy, early consideration is essential because many energy opportunities are related to the fundamental building form.

We are developing software which allows users to sketch buildings at any level of detail, then provides a schematic level evaluation of its energy performance. The sketch pad environment encourages imagination because it is unbridled by rigid input requirements. Users may draw anything they choose, regardless of its relevance to energy (or even architecture), and they may design their own lines styles and symbols to do so. They may request technical evaluations as often or as infrequently as they like, and may rely on a large body of defaults to cover any gaps in their building definition. Calculation algorithms are greatly simplified, both to speed up the computer's response time and to minimize the amount of specification the user must be burdened with. By seeing evaluations of their work frequently and easily, users may develop a familiarity with and an intuition for the effect of energy considerations on their design process.

Two Modes of Thinking
In her book, Drawing on the Right Side of the Brain, Betty Edwards describes the cognitive differences between the functioning of the brain's two hemispheres. Without vouching for the anatomical accuracy of the terminology, she describes two modes of thinking that are generally attributed to the two sides. The left mode is the rational: it controls the verbal, analytical, linear, factual, objective, numerical, sequential, methodical, logical, sequenced ways of thinking. The right mode, by contrast governs the non-verbal, synthetic, subjective, holistic, rapid, intuitive, metaphorical ways of thinking. The left mode is good at figuring things out; the right mode is good at seeing new patterns and spatial relationships and suspending judgement while imaginative leaps are made.

Training both Hemispheres in Architecture Schools
Certainly in the education and practice of architecture we see a clear case for bi-modal thinking. An
individual's design process ideally jumps frequently between the wildly imaginative and the realistically sober. Spatial relationships and creative combinations of patterns can only come when the right brain is free to play unencumbered. But these ideas can only become a building after being subjected to the analytical scrutiny of the left brain and its concerns for structural soundness, social needs, economies of material, etc.

Unfortunately, it is all too common for the latter considerations (the domineering left brain) to take over in a student's design process. How many times have we seen bright students endlessly analyze problems with the hope that some kind of synthesis will result from a set of increasingly narrowed constraints? What they really need is an escape from their calculation mode into a synthetic mode of thinking. When an analysis precedes design, it should be set up pedagogically to lead naturally to synthesis. One way to do this is with a rich library of images and built examples.

The Role of the Computer in the Right-Left Battle.
So where has the computer been in this battle between the left and right modes of thinking in architecture schools? With few exceptions, it has sided with the left brain. The list of left brain characteristics (analytical, methodical, sequential, etc.) is precisely what best describes computers. Everything must be carefully defined; even graphic information is meticulously described and hard-edged. But most information is not even graphic; rather it is highly numerical and verbal. In a common scenario, the computer demands the dimensions of a building in a sequence inspired by formulas rather than human cognitive processes; this hardly encourages a flight of fancy.

Computers in fact exacerbate the dominance of left brain in architecture schools by encouraging students to think that the tightening constraints of a sophisticated analysis will lead to an optimal design solution. In fact, a thorough analysis usually tells you what not to do rather than what to do.

Suggestions for Idea Generative Computer Programs
In her essay, "Changing Pictures/Changing Minds", psychologist Kristina Hooper describes her vision of the evolving role of the computer in the generative domain of initial thinking. She describes an ideal design system as one which, by virtue of its rapidly changeable images and easily stored versions at many levels, can mirror the designer's creative processes of rapid, pictorial "self dialogues", while making clear the implications of changes in rapid evaluations. The computer's display, rather than being an end in itself, is "... the snapshot of an image idea becoming increasingly more coherent as it is worked over time."

In his essay, "The Interface to Design", Eric Hulteen describes the need for an "idiosyncratic" interface, one which responds to an individual designer's unique language, preference for graphic vs. verbal, degree of computer involvement in the process, and needs for detailed references.

In Nicolas Negroponte's 1975 classic, Soft Architecture Machines, he raises many of these concerns. "A straight line 'sketch' on a cathode-ray tube could trigger an aura of completeness injurious to the designer as well as antagonistic to the design." His ideal computer watches over the designer's shoulder from the very early stages of conception and recognizes the intentions of a designer's marks rather than the designer being responsible for defining everything. It allows "... a wide variety of approaches (some contradictory)..." It draws high-level inferences from nebulous ideas described by wobbly lines, collections of overtracings, and a rich accumulation of exploratory "freewheeling, equivocal, and inaccurate" marks that have meaning primarily in the visual feedback the designer is giving herself or himself. This allows the designer to stay in the right state of mind to do the design.

To achieve these ends, Negroponte and his research team at MIT wrote the program, HUNCH, which examined sketches and tried to "infer" meaning. At the lowest level, of translating sketchy lines into hard-lined graphic primitives, they were partly successful; at the highest level, of interpreting architectural meaning from the graphic symbols, they were not. Indeed, it is questionable if researchers in "artificial intelligence" will ever get to the point of being able to program a computer to 'do what I mean, not what I say.'

While his intentions were admirable, Negroponte's solution was too ambitious to be realistic.
assuming that the computer must be wholly responsible for the interpretation in order to be unobtrusive in the designer's process. We have proposed a less ambitious solution, which is somewhat more obtrusive, but more doable.

Nevertheless, we agree with his basic assertions: that "(1) it is crippling to force an explicitness in contexts where the participant's equivocations are part of the function of design; (2) the tedium of overt, categorical exchange is counterproductive, unfulfilling for the speaker, and boring; (3) constructive and exciting responses are often generated by twists in meaning that result from the personal interpretation of intentions and implications."

How our Software Nurture Intuition
In an attempt to answer many of the concerns that researchers have been voicing for over a decade, we have worked out a prototype program which both provides a set of technical (energy) evaluations and an environment that fosters a highly visual, right-brain mode of thinking. This has been done in a variety of ways.

The Graphic Interface
A sketchpad environment allows the user to doodle in any style with any content, "mindlessly", without having to worry about defining attributes or naming parts. This is similar to many paint programs except that it includes several features that are specific to architectural drawing habits. Users can design their own line styles, private symbol systems, and pictorial languages. A designer can communicate a lot to him or herself through these personal symbol systems and when initial ideas are too tentative to translate into a hard-edged, well-defined CAD notation.

Figure 1. Personalized symbol systems. A simple floor plan drawn by three designers, each with a different set of graphic conventions for describing enclosure and opening.

Special tools include graphic scales and triangles that look and behave more like the physical objects they represent than the mathematical equivalents found in most CAD programs.

![Scale Selection](image)

Figure 2. A scale selection scheme designed to resemble the physical devices found on a designer's conventional drawing board.
Rich Imagery
Besides the drawing tools and palettes, there is a library of historical examples which users can call up at any time. These historical buildings are shown in any of several energy contexts, or simply as undocumented drawings. They are indexed by building, architect, climate location, or the design strategy they illustrate, so that they are accessible from any number of widely differing needs or interests.

Figure 3. A drawing index. Library images are referenced by means of a set of miniatures.

Small openings were used to let in sunlight which was then reflected to the ceiling, which in turn reflected it onto the walls and floor.

Perforated louvers also let some daylight directly into the space to reduce glare.

Figure 4. The Kimbell Art Museum used to illustrate daylighting in a hot climate.

The output is also graphic. By this we mean not only that the numeric data is displayed as graphs, but we also add an innovative meaning to this assertion. Evaluations are displayed in the same medium as the input they respond to. For example, if a user has drawn the windows so that they are too small to accomplish solar heating, the window sizing evaluation will redraw them at a more appropriate size. The user thereby gets the benefit of feedback on a technical question without ever leaving the highly visual environment of the sketchpad.
Minimizing Numeric Input
To satisfy the algorithmic need for eventual definition of the building characteristics, we append a "takeoff" section that allows the left brain to temporarily intercede and make its measurements on the right brain's doodle. In practice, the information that is eventually taken off will be a small fraction of the sketching that the user will want to engage in for a "self dialog". The program's technical content (eg. energy) is not allowed to overwhelm and dominate the more inclusive design process.

In the illustration below, the user is "taking off" measurements from a south elevation for solar sizing evaluation. The "Window Size Aid" at the lower right represents the optimum window size (according to pre-specified parameters) and is filled with a grey pattern to match the area of glass which the user has measured. By watching the fill pattern grow, the user can decide when a building has enough glass for solar heating, without ever seeing window requirements or results numerically.

This window sizing procedure is one of several intermediate evaluations that can be performed on building components. The optimal window size is determined by criteria which the user selects (solar heating, cross-ventilation, etc.). Not until the final evaluation are the windows considered in the context of the overall, year-round energy balance.

Figure 5. With graphic "takeoffs", building areas can be measured for evaluation without requiring numeric input.

Procedural Flexibility
Like most software with a "Macintosh interface", the user is permitted, with some exceptions, to do anything, at any time. This caters to the fanciful right brain by not requiring the user to proceed in an ordered fashion. The user may opt to do a thumbnail design and get a rough evaluation or a full blown building with a complete annual energy evaluation. In either case, or in any intermediate case, the computer provides a complete set of default data to fill in anything that is "missing". This is very important, as it allows a quick and dirty evaluation for, say, general orientation of the building, without the user having to specify building data that is more particular than what has yet been designed or even considered. The defaults are based on minimum code requirements and simple judgements about "typical" materials and building shapes. The defaults are completely and
easily replaceable and serve only to fill in gaps in a design.

Simplified Algorithms
The technical algorithms for the program have been greatly simplified both to speed up the iterative nature of the design process, but, more importantly, to simplify the user interface so that the goal of presenting design alternatives is not overwhelmed by meticulous detail.

![Wall Type 1 (NEW)](image)

**Figure 6.** Defining a wall's thermal characteristics. Any part of the building database may be defined or modified numerically, if desired.

The final evaluation output consists of graphs and tabular reports on the heat loss/gain balance for a 24-hour day for each of four seasons. In addition to conventional heat loss and gain calculations, we also include the reduction in gain from electric lights in the event of effective daylighting, the hourly loss to, or gain from, thermal mass, and heat transfer between thermally unbalanced zones of the building. The user can determine from these reports how design changes to one or more components will change the overall building performance.

Very specific details that do not influence building form (e.g., the emissivity of materials) are generally not included in user input procedures. Because of this simplification, we are careful not to imply more significance than is warranted by our schematic level energy evaluation. For instance, we will make sure that the output format is readily convertible to become input to more sophisticated analysis programs such as DOE-2.

Interactive Learning
The transition to "technical" material is smooth and graphic so that the user is not aware of a cognitive shift of focus. A major part of our software features a set of simulations, or "learning labs", that provide on-screen tutorials to teach what is normally considered to be "technical material" when the student user finds a gap in his or her understanding of the vocabulary or principles. These tutorials are highly graphic, interactive, self-contained modules that illustrate physical phenomena such as the sun's movement over the sky dome and the flow of heat through a wall. They are animated in real time to demonstrate the dynamic character of the phenomena in question, and, more importantly, they react to student input, so the student must be actively involved in the exploratory simulation. As an example, the user can specify a season and latitude and then view the sun moving, hour by hour, through a day's path. Since the input is provided by pointing at options, there is no need for left brain distractions such as typing or naming. Since there is very little that is structured or "required" of a student in this learning mode, there is no distraction from an exploratory behavior that is capable of building intuition.
Figure 7. A learning lab for sun position. The user changes the time, date, and latitude settings on the right, and the display indicates sun and shadow positions in plan, elevation, and axonometric view.

Mimicking Familiar Processes
Other features are under development that extend this software to maximize its smooth compatibility with traditional design processes. One example of this is a "Yellow Trace" function. Most graphic software has cut, copy, move, scale, and paste functions, but none of these, even in combination, really match the cognitive complexity that human beings express in and through the medium of tracing paper. In addition to cut, copy, move, scale, and paste, designers use tracing paper to reshape, make more specific, make less specific, recombine several drawings, keep track of old versions, apply proportional systems, use early versions as guide lines, combine several layers and then selectively redraw, and selectively modify only part of a drawing at one time. Our goal is to simulate the variety and complexity of the traditional tool.

Conclusions
For the last 20 years or so, architectural software has been making great advances in the general direction of trying to pack more information into fewer lines of drawing. All of the left-brain characteristics of the computer have been exploited, and progress will continue on this front for years to come. However, it's now time for software developers to also concentrate on the right-brain region of design, in a mode which enhances the creative aspects of the design process by providing a nurturing environment that fosters those creative states of mind.

Our software, of which a prototype is scheduled for completion by September, 1987, will incorporate a number of features that go in this direction. These include a low-pressure sketch environment, graphical drawing aids that resemble real tools, visual as well as numeric evaluations, graphic simulations of physical phenomena, and a library of on-screen illustrations.

References

The work on this project has been supported by the U.S. Department of Education's Fund for the Improvement of Post-Secondary Education, the University of Oregon Department of Architecture, and Apple Computer Inc.