A Hypermedia Model for Supporting Energy Design in Buildings

Khaled A. Al-Sallal
Larry O. Degelman
Texas A&M University
Texas A&M University

Several studies have discussed the limitations of the available CAAD tools and have proposed solutions [Brown and Novitski 1987; Brown 1990; Degelman and Kim 1988; Schuman et al 1986]. The lack of integration between the different tasks that these programs address and the design process is a major problem. Schuman et al [1986] argued that in architectural design many issues must be considered simultaneously before the synthesis of a final product can take place. Studies by Brown and Novitski [1987] and Brown [1990] discussed the difficulties involved with integrating technical considerations in the creative architectural process. One aspect of the problem is the neglect of technical factors during the initial phase of the design that, as the authors argued, results from changing the work environment and the laborious nature of the design process. Many of the current programs require the user to input a great deal of numerical values that are needed for the energy analysis. Although there are some programs that attempt to assist the user by setting default values, these programs distract the user with their extensive arrays of data. The appropriate design tool is the one that helps the user to easily view the principal components of the building design and specify their behaviors and interactions. Data abstraction and information parsimony are the key concepts in developing a successful design tool.

Three different approaches for developing an appropriate CAAD tool were found in the literature. Although there are several similarities among them, each is unique in solving certain aspects of the problem. Brown and Novitski [1987] emphasize the learning factor of the tool as well as its highly graphical user interface. Degelman and Kim [1988] emphasize knowledge acquisition and the provision of simulation modules. The Windows and Daylighting Group of Lawrence Berkeley Laboratory (LBL) emphasizes the dynamic structuring of information, the intelligent linking of data, the integrity of the different issues of design and the design process, and the extensive use of images [Schuman et al 1988]; these attributes incidentally define the word hypermedia. The LBL model, which uses hypermedia, seems to be the more promising direction for this type of research. However, there is still a need to establish a new model that integrates all aspects of the problem. The areas in which the present research departs from the LBL model can be listed as follows:

it acknowledges the necessity of regarding the user as the center of the CAAD tool design;

it develops a model that is based on one of the high level theories of human-computer interaction; and

it develops a prototype tool that conforms to the model.

Objective

The general objective of this research is to develop a model that can facilitate the interaction between the designer and the CAAD tool based on what activities should be delegated to the computer for automation and what activities should be performed by

...
the designer. This research concentrates on the design of the user interface as opposed to other research in the field that may concentrate on the efficiency of algorithms or authentic CAAD drawing capabilities. The main concern is to improve the design environment to be more desirable to architects.

Theoretical Background

The most important consideration in the design of the EnerFace model is the ability to describe (or sketch) a building in a physical way that harmonizes with the user's thinking and not to burden him/her with tasks that might disturb the flow of the design process. The tools that provide the quickest and easiest ways for describing building shapes and characteristics are the preferred ones. This is especially important in the conceptual stage of the design when the designer needs to explore many ideas through the process of synthesis and analysis. One should realize that the design process in the early stages involves feelings about what the design could be rather than concrete facts of what it is or what it should be. With this consideration in mind, the EnerFace software deliberately avoids procedures that look like detailed "drafting" procedures. Typical drafting software would have the user define a shape point by point by moving a mouse and clicking on its button while observing the coordinate fields on the screen. These programs not only hinder the design process but they might also degrade the design quality. This happens when users find themselves involved in highly detailed procedures that make them lose their imagination of the entire picture of their design.

Since architects deal with building objects (floors, walls, windows, etc.), at least in their imagination, as means for dividing and forming spaces, they should manipulate effective representations of these objects and not primitive shapes such as the circles used in the drafting packages (e.g., lines, squares, and circles). Effectiveness in the context of this section means less mental effort on the part of the user to understand the conceptual model of these representations. This follows the cognitive approach to human-computer interaction that was described by Eberts and Eberts [1989] as follows:

The overall goal of the cognitive approach can be stated pictorially in Figure 1 (adapted from Norman, 1983). Three concepts are important. First, the conceptual model is a description of the computer system in engineering terms so it is accurate, consistent, and complete. The mental model is the model the user forms of how the computer system works; this mental model guides the user's action and behaviors. The mental model is built up through interactions with the display representation which provides a picture of the conceptual representation. The goal is to try to choose the information to represent on the display to enable the mental model, like the conceptual representation, to be accurate, consistent, and complete.

![Conceptual model](image1)

**Figure 1:** Relationship between the conceptual model, the display representation, and the mental model. (From Four Approaches to Human Computer Interaction)

With this theoretical background in mind, the authors established the basis for the development of the EnerFace model following the basic approach of Zeigler [1976] to modeling and simulation.

Real System

Papamichael [1991] defined design as the direct control of the values of design variables in order to indirectly control the values of performance variables. In discussing the activity of design description as part of the design process, Papamichael [1991] wrote "Control variables are usually specified in an object-oriented fashion. The will-be situation is defined in terms of objects, which have attributes and may be children or parents of other objects, and can be seen as hierarchical, treelike structures." Applying this to the energy building design, one can say that the real system consists of objects such as spaces, walls, windows, floors, and roofs. Each object has a set of variables (or attributes) to which designers assign values in order to describe it as part of the whole.
system. The number of these variables can be extremely large and the relationships among them can be complex. Some of these objects are subsets of others — like a wall as a subset of a space and a window as a subset of a wall. In this treetlike structure, the lower level objects cannot exist unless the higher level ones exist (e.g., no window without a wall). Although a space is not a real object, it is an imaginary one that is formed by other objects such as walls, roofs, floors, and windows. In real life we do not manipulate its geometrical or physical characteristics directly; rather, we manipulate the characteristics of the other objects that affect its characteristics.

Thus, the hierarchical structure is appropriate for organization of the real system information. This is even more supported by the theory of information chunking that was discussed by Akin [1986]. The human memory organizes pieces of information into chunks that have common relationships binding them together. The chunking of information allows larger numbers of units of information to be processed at one time. A primary strength of this theory is that it accounts for the limitations of short term memory span and supports the hierarchical and multirelational organizations of information. In task domains with a predominantly spatial problem spaces (chess and recall of graphic elements), chunks are the most robust information structures which have been shown to account for memory functions.

In discussing the degrees of strength for the validity of models, Zeigler [1976] considered that the strongest degree of validity concerns the relation between the structure of the model and the internal workings of the real system. He wrote: “A model is structurally valid if it not only reproduces the observed real system behavior, but truly reflects the way in which the real system operates to produce this behavior.”

The hierarchical structure, which is suggested by the theory of information chunks, establishes a very significant point for the design of appropriate user interfaces of CAAD systems. As mentioned in the beginning of this chapter, the goal of the cognitive approach to human-computer interaction is to try to choose the information to represent on the display to enable the mental model, like the conceptual representation, to be accurate, consistent, and complete. Although it is hard to predict the user’s exact mental model of a computer system, one can argue that users in the same domain have a good chance to have roughly similar mental models. This is because of the similarity of experiences they encounter every day in their domains, which essentially come from the nature of the real system. Accordingly, in order to improve the user’s mental model of the computer program, the conceptual model and its display representation should support the hierarchical organization that exists in the nature of the real system. Since this hierarchical structure allows multiple chunking of information, it can improve the user ability to recall large amounts of information such as those encountered by architectural designers.

Model Components

EnEface attempts to solve the problem of the user interface by representing the basic components of the design with objects and encapsulating their characteristics and methods within these objects. In this way, the user can easily view the principal components of the system and specify their behaviors and interaction. Objects provide both data abstraction and information parsimony that help to modularize a problem in its earliest stages of analysis. Another important point in the design of EnEface is the graphical representation of its components. Due to the power and ease with which graphics convey information, EnEface is totally dependent on graphics. EnEface supports easy and effective user interface by providing natural pictorial (iconic) representation of real objects in the physical environment. Icons which look like the components of the system being modeled are placed on the screen to show the spatial relationships. This is especially important in modeling architectural systems where the spatial relationships are as important to designers as the internal information of objects.

An object consists of data, data structures, and procedures for manipulating that data. Objects communicate with other objects by passing messages. An object associates a certain procedure with the message and on receiving the message executes the appropriate procedure. When variables and data change
to provide existence to an object, it is called an instance. EnerFace is comprised of various types of objects which model the entities involved in the energy design process. Encapsulation is a mechanism that allows an object to interact (or communicate) with other objects in a standardized and abstracted manner so that the internal components of modules can change without affecting the interfaces to those components. This design strategy allows one to implement or re-implement modules without affecting the operation of the remaining system components [Laskov 1987]. With this definition of encapsulation, each instance of a class is a separate encapsulation or component in a problem solution [Lee 1991].

![Diagram of building components hierarchy]

**Figure 2: Class hierarchy of building components**

A class is the structured knowledge describing one or more similar objects. A class description involves defining all the properties and behaviors of any object that is an instance of that class. The lower levels in a class hierarchy, subclasses, usually represent an increased specialization, and higher levels, superclasses, represent generalization. The object-oriented system provides a range of capability relating to the extent to which objects of a subclass may inherit, extend, or override the characteristics of the superclass. Classes of the objects that are needed for the energy building design as well as their attributes should be specified up front as the basic components of the conceptual model (see Figure 2). These are as follows:

**Building**

A building is defined as an object existing in the topmost level of a hierarchical structure. It is formed by a group of lower level objects (spaces, walls, doors, windows, and skylights) and has attributes, called building attributes, that provide a description of its status. Each object, however, has its own attributes, called object attributes, that provide more specific description of the object status (see next section). The attributes that describe a building object are name, site, building type, location, and orientation.

A building descendant is generally defined as an object that is represented by a rectangle in the computer screen and is used to manipulate, as well as to access, the values of its attributes through the processes: move, change size, delete, and get info. There are five classes of building descendant objects that exist in EnerFace which can be listed and defined as follows:

**Space**

A space is defined as an object existing in the first level of the building components' hierarchy structure that has attributes, called space attributes, for controlling the space design variables and providing access to their values. The attributes that describe a space object are dimensions, ceiling height, roof construction, slab construction, space type, internal thermal mass, thermostat temperature, occupancy, ventilation rate, hot water usage and temperature, infiltration, and lighting density.

**Wall**

Wall can be defined as an object existing in the second level of the building components' hierarchical structure that has attributes, called wall attributes, for controlling the wall design variables and providing access to their information. The attributes that describe a wall object are dimensions and construction.

**Door**

Door can be defined as an object existing in the third level of the building components' hierarchical structure that has attributes, called door attributes, for controlling the door design variables and providing access to their information. The attributes that describe a door object are dimensions and construction.
Window or skylight
Window (or skylight) can be defined as an object existing in the third level of the building components' hierarchical structure that has attributes, called window attributes (or skylight attributes), for controlling the window (or skylight) design variables and providing access to their information. The attributes that describe a window (or skylight) are dimensions and construction.

Model Operation
In the EnerFace model, new objects can be formed by extending functionality and data attributes supported by existing object definitions. Similarly, new object definitions can serve to refine or tailor existing object behavior in order to support specific applications. This process is called Inheritance in the object-oriented programming terminology and provides a framework through which object behavior and/or data can be utilized by other objects in the system [Kacmar 1990]. EnerFace uses this process to reduce the mental load on the user by helping him/her to form the objects while avoiding highly detailed procedures. Inheritance also helps to avoid risky designs that cannot result from inexperienced users by generating the design from prototype objects whose attributes were chosen according to results of previous research.

The process of assigning values to the attributes of objects goes through three levels. The first level starts when the user selects a building type from the BuildingInfo window (see Information Processes section). When the user does this, the program sets one of its thirteen space-type prototypes as the default space object. The user probably chooses the prototype space based on the most prevalent space type. When the user generates new spaces, these generated spaces inherit their attributes and behaviors from the default prototype space. The second level starts if the user finds that the type of one (or more) of these spaces is different from the prevailing one. If this happens, he/she at any time can change its type to another prototype from the SpaceInfo window by selecting another space type. Similarly to the first level, the different space inherits its attributes and behaviors from the new prototype space. The third level starts when the user wants to change the value of a specific attribute. The user can do this by entering a new value for the desired attribute. EnerFace uses a process called Polymorphism.

This process allows a common message to be sent to all objects within a class and its subclasses such that each object knows how to respond to this message. For example, when the message change_size is sent, an object, such as a space, will change in width or length whereas another object, such as a wall or a window, will change in only the length (assuming that the thickness of the wall or window is constant.) The ability to use the same message for a similar operation on different kinds of objects is consistent with the way human beings think about solving problems. Hence, this process can contribute to improving the user interface.

Creation Processes
As stated previously, the overall goal of the cognitive approach is to try to choose the information to represent on the display to enable the mental model, like the conceptual representation, to be accurate, consistent, and complete. The intelligent interface reduces the semantic distance [Norman 1987] across the cognitive interface by providing system output in a form that corresponds closely to the user's model of the task, and by allowing the user to express intentions in the language of that user model. The hierarchical representation of the building components' information, by corresponding to the real system, can provide the right structure to develop display representation (system output) that is amenable to cognitive analysis by the user. Using this cognitive structure, one should create a space before creating its walls or windows. Since a building is a group of objects and not an object by itself, one should not expect to create a building before creating its spaces. Accordingly, the EnerFace object creation processes should follow and support this hierarchical representation in order to always keep the mental model of the user accurate, consistent, and complete.

derives from the memory and processing capabilities of the computer. In effect, computers add external loops to our basic communication with ourselves. Researchers in computer applications have pursued diverse approaches including modeling of quantitative measures and dynamic processes, form grammars, and simulations. These may be viewed both as structured information or as less
The user creates a descendent object (e.g., a wall is a descendent of a space and a window is a descendent of a wall) by following this general form process:
- Click once on the tool button that has the icon of the desired object in the Tools palette.
- Click once on the parent object that will have the desired descendent object. In the case of space object, skip this step.
- Release the mouse button.

Once the user releases the mouse button, and depending on which tool button was clicked on in the first step, one of the following will result:

**Space**
A new space object appears on the upper left side of the screen (see Figure 3). The program sets the horizontal and vertical dimensions of this space to the default values that exist in the default settings window and can be changed by the user at any time. Since EnerFace does not support numerical input, due to drawbacks that were mentioned previously, the numerical input style is not active unless the user requests it (see Get_Object_Info and Change_Object_Attribute(s) sections). In other words, changing the dimensions of the space (or any other object) will depend essentially on the manipulation processes which are explained later. The current version of EnerFace supports only rectangular shapes; however, we intend to include other shapes in the future.

**Wall**
A new wall object appears on the edge of the space that was chosen in step the first step. The new wall extends along the entire chosen edge of the space without any windows. The program sets the thickness of this wall to the default value that exist in the default settings window and can be changed by the user at anytime.

**Door**
A new door object appears on the wall that was clicked on. The new door width is less than its wall width by a default value that can be changed any time by the user. The program always sets the thickness of the door to the thickness of its wall.

**Window**
A new window object appears on the wall that was clicked on (see Figure 4). The new window width is less than its wall width by a default value that can be changed at any time by the user. The program always sets the thickness of the window to the thickness of its wall.

**Skylight**
A new skylight object appears on the space that was clicked on. The program sets the horizontal and vertical dimensions of the skylight to the default values that exist in the default settings window and can be changed by the user at any time.

---

**Figure 3: Creating a new space**

**Figure 4: Creating a new window**

---

Howard's End

Structured Processes
Serendipity

ACADIA 1994
Manipulation Processes

The hierarchical representation of the building components' information can be a structure equally useful to the manipulation processes as it is to the creation processes. In other words, it can provide the right structure to develop display representation (system output) that is amenable to cognitive analysis by the user. Following this cognitive structure, any manipulation process will affect not only an object but also its descendants. For example, when the user moves a space, the walls, doors, windows, and skylights of this space will move a distance that is equal in value and direction to the distance that the space moved. Accordingly, the manipulation processes will follow and support this hierarchical representation in order to always keep the mental model of the user accurate, consistent, and complete.

The user manipulates (move, change size, or delete) an object by following this general form process (see Figure 5 and Figure 6):

- Click on the tool button that has the action icon of manipulation (e.g., moving).
- Click on the object that you want to manipulate. In case of move or change size processes, keep the mouse button down, and drag it to the desired location, by moving the mouse.
- Release the mouse button.

Move

The move process has an effect on the objects building, space, wall, door, window, and skylight. This process can move the object space and its descendants in horizontal and vertical directions whereas it can move the objects wall, door, and window and their descendants in only one direction, either horizontal or vertical, which was parallel to the direction of the selected object.

Change Size

The change size process has an effect on the objects space, wall, door, window, and skylight. This process can change the size of an object space in horizontal and vertical directions, whereas it can change the size of the objects wall, door, and window in only one direction, either horizontal or vertical, whichever is parallel to the direction of the selected object. Changing the size of an object requires one to move one or two of its edges. The object's descendants that lie on the moved edges move to the new edge location. Also, the sizes of its descendants change so that they cannot exceed the size of the parent object.

Delete

The delete process has an effect on the objects space, wall, door, window, and skylight. When the user deletes an object, all of its descendants will also be deleted.
Interactions of Objects

The basic differences among the object classes are in the names, number, and values of attributes that describe each one of them. However, in some cases, an attribute of an object could carry the same name, and possibly the same value, as another in a different class (e.g., the wall width could be the same as the space width). In another case, an attribute of an object should carry the same name and the same value as another in a different class (e.g., the window thickness should be the same as the wall thickness). On the other hand, there are cases in which an attribute value of a particular object should not exceed the value of the corresponding attribute in another object (e.g., the window width should not exceed the wall width). Without rules that control the interactions among the different objects, the validity and reliability of the EnerFace model could be threatened. Examples of these rules can be listed as follows:

- If an object is moved, or its size is changed, it should not overlap another object of the same class.
- If an object is moved a distance, move all its descendents with the same value and direction of that distance.
- If a space edge is moved as a result of changing its size, move the descendents of the space that lie on that edge to always follow the space edge.
- If an object is moved, or its size is changed, it should not exceed the boundaries of its parent.
- If an object is deleted, delete all its descendents.

Get Object Info

In the case of any other object (space, wall, door, window, and skylight) information, the general form of this process is as follows (see Figure 7):

- If the ObjectInfo window is not on the screen, click on the tool button that has the icon of a magnifier in the Tools palette to activate/deactivate it.
- Drag the mouse into the object boundary that you want to get or modify its information. A window pops up showing all the attributes of the object.

![Figure 7: The BuildingInfo window is used to access information tools such as the orientation](image)

Change Object Attribute(s)

After displaying the object information window, the user wants to change one or more of the object attributes. The general form of this process is as follows:

- Move the mouse to the attribute's value control (this can be any kind of control to change the value of an attribute such as a field, a sliding bar, or a pop-up list) whose value you want to change (or just check) and change it. Repeat this step for other attributes, if needed.
Get_Object_Info and Change_Object_Attribute(s) are general forms that can represent more specific processes, rather than being processes by themselves. Each can work with any attribute of any object. For instance, when one replaces the word "Object" with "roof" and the word "Attribute" with "Construction" in Change_Object_Attribute(s), it becomes the process Change_Roof_Construction. Depending on the type of the attribute's value control, their specific forms will vary. To give the reader the whole spectrum of these variations, two specific processes are explained here and can be used as examples to illustrate the concept.

Change_Building_Location
The user changes the building location by following this process (see Figure 8 and Figure 9):
- Click on the button that has the icon of the US map in the BuildingInfo window. The program opens the graphical weather database displaying a full-screen image of the US map with state names on it.
- Click on a state name. The program pops up a list of cities for this state.
- Click on the desired city.
- To view the weather data, click on the magnifier icon; otherwise, click on the EnerFace program icon to go back to the main program.

The program stores this location name (city and state) to be used later in retrieving its weather data for running the energy analysis.

Change_Roof_Construction
The user changes a roof construction by following this process (see Figure 10 and Figure 11):
- Click on the pop-up list button that has the label "Construction" in the SpaceInfo window and hold the mouse button down. The program pops up a list of roof constructions.
- Select the desired roof construction, then release the mouse button. The program displays the last one chosen, indicating that it has picked up this roof construction.
- To view its information, click on the button "Database". The program opens the graphical roofs database, displaying an image of the roof with its thermal data. At this point, the user can go back to the main program or browse the database to search for another roof.
- When the appropriate roof is found, click on the button "OK" to go back to the main program. The program displays the roof of the last screen in the pop-up list button, indicating that it has picked up this roof construction.

Figure 8: First screen of the weather database

Figure 9: Example of weather data for a specific city

Summary and Future Work

The paper began by discussing the limitations of the available CAAD tools and the current approaches for improving them. It emphasized two key concepts, data abstraction and information parsimony, that need

Ecological communication is already a powerful force in design. Satellite generated imagery has revealed the impact of human encroachment, radically changing our view of future challenges for architecture. Pioneers in the study of sustainable natural and human environments initiated a study of the implications for architectural education and practice in the Cooper Landing, Alaska.
to be considered in the development of such tools. It presented a rationale for the need for a different philosophy with which to develop these concepts, and it emphasized the significance of using one of the human-computer interaction approaches. The cognitive approach was the one adapted for this research. It proposed using the hierarchical structure as an appropriate structure for organization of energy design information. Based on two theories in design process and information chunking, the paper supported the authors' view by showing the cognitive benefits that could be gained when using the hierarchical structure. It also showed how the object-oriented model can be beneficial in implementing this structure. The paper then defined the different objects that make up the model and the attributes of each object. It also described the operation of the model based on the two important concepts inheritance and polymorphism and it showed how these concepts can improve the mental model of the user.

After developing the prototype tool, the authors will evaluate it based on a conceptual definition of usability that relates to the ease and effectiveness of using it. The purpose of this evaluation is to provide feedback as part of a comprehensive methodology for advancing the state of the art of CAAD tools. The evaluation criteria will include performance and subjective measures that operationally define usability.

Acknowledgments

The authors wish to thank CRSS, Inc. for their continuous generosity to support the College of Architecture, Texas A&M University, and in particular, their support for this work through the Caudill Fellowship. Thanks are also due to Dr. Ronald Zellner, Dr. Lester Boyer, and Dr. Valerian Miranda for their valuable inputs.
References


Conference and the ACSA Teachers Seminar, held earlier this year under the umbrella of the EASE (Educating Architects for a Sustainable Environment) project organized by Marvin Rosenman at Ball State University. A broad community of scientists, engineers, architects and environmentalists are examining the composite effects of the processes of human habitation on our planet and