ARCHPLAN - an Architectural Planning Front End to Engineering Design Expert Systems

by

G. Schmitt

EDRC-48-04-87

ARCHPLAN - an Architectural Planning Front End to Engineering Design Expert Systems

Gerhard Schmitt
Engineering Design Research Center
Department of Architecture
Carnegie Mellon University
Pittsburgh, PA 15213

1. Abstract

ARCHPLAN is a knowledge-based ARCHitectural PLANning front end to a set of vertically integrated engineering expert systems. ARCHPLAN is part of a larger project to explore the principles of parallel operation of expert systems in an Integrated Building Design Environment. It is designed to operate in conjunction with HIRISE, a structural design expert system; with CORE and SPACER, two expert systems for the spatial layout of buildings; and with other knowledge based systems dealing with construction planning, specification, and foundation design. ARCHPI.AN operates cither in connection with these expert systems or as a stand-alone program. It consists of three major parts: the application, the user interface, and the graphics package. The application offers a knowledge based approach towards the conceptual design of high rise office buildings, taking into account qualitative and quantitative considerations. Strategies used for design are prototype refinement, evaluation, and local optimization. The four major modules in the ARCHPLAN application deal with massing, building functions, vertical building circulation, and structure. The user interface provides a graphical environment for the interactive design of buildings and monitoring program states. Ihe graphics package allows the workstation to function as the external representation medium of design decisions made by the user and the application. A particular emphasis of ARCHPI AN is to explore the usefulness of object-oriented programming techniques to support the abstract representations of the design process and the resulting building.

Keywords

Object-oriented programming, prototype refinement, conceptual design process, interactive design, building performance, vertical integration.

Table of Contents

List of Figures

2. Introduction

Computer-aided drafting tools are employed to record and manipulate the results of design decisions. In that sense, the present use of computers differs only slightly from the traditional recording of design ideas and design stages on an external medium, such as paper. Only the geometric properties and a few other quantifiable attributes of design are represented by the models or abstractions used in current computer-aided design programs. This approach places heavy emphasis on the syntactic aspects of design and represents a building at a very low level of abstraction. Due to the lack of appropriate abstraction and representation methods, and their consequently missing computational counterpart, the semantic and conceptual aspects of design decisions are not sufficiently covered and must be supplied entirely by the user.

As a result, most designs created on computers are one-dimensional in their treatment of the complex issues involving the design process. Moreover, when quantifiable properties of the design are evaluated, for example the energy performance of a building, the user is forced to take descriptions and quantities from the lowest level of representation, in this case the geometric representation. Abstractions must be made on the geometric model, which in itself is an abstraction, and as a consequence, the results of evaluation are unreliable. We therefore propose to start the quantitative and qualitative performance evaluation of architectural design at a higher level of abstraction. This approach towards architectural design modeling requires a representation and abstraction concept different from traditional approaches. A hybrid system, consisting of traditional and object-oriented programs is explored to model the conceptual design of high-rise buildings.

3. Representation of Architectural Design

Over the last few centuries, design professionals have developed one of the most powerful forms of representation: the graphical image whose syntax induces semantic explanations in educated viewers. In other words, we have become so familiar with the symbols and techniques of graphical representation, that we are able to interpret meaning where the untrained eye or the computer would recognize only lines or surfaces.

Representation involves abstraction. Abstraction is the reduction of a real world object (a building, a tree, an idea) to its most important characteristics, according to a certain model. Abstraction and with it representation became necessary with the paradigm change from making buildings towards planning buildings [Schmitt86b|. It is crucial that the creator and the viewer or user of the abstraction base their work on the same model. With the introduction of computers in the design process, new forms of representation and abstraction become necessary. Several approaches were explored in the past, three of which arc of particular interest in this context:

Geometric models [Eastman751, [Eastman77|. Geometric models describe the geometric properties of a design. They are based on the assumption that the architectural design stages are representable with data structures of **varying** complexity [Mclntosh82|. The simplest data structures for two-dimensional representations are lists of points, lines, and polygons. Three-dimensional representations require more complete information, especially if realistic views of the object are a concern. Winged edge and boundary representation data structures are only two of a number of possibilities if solids are represented, and the typical set operations of union, difference, and intersection are to be performed. While the data representations are quite efficient and workable for present computer programs, they are not transparent to the designer who thinks and designs in quite different categories and operations.

Relational Databases. Relational database management systems are important tools in business. The underlying principle of relations or tables is useful in the representation of design as well. The relational model is able to express properties (in the rows and columns of the table) and relations (through primary and foreign keys in the table) in a straightforward manner. The relational model has higher semantic quality than, for example, the hierarchical model. Although there were approaches to use the relational model for solids modeling, widespread applications of the model to represent design in its different stages are not **yet** implemented. This approach takes the existing relational view of data and extends it, treating shapes as attributes [McIntosh84J.

Frames [Minsky75]. Frames, also known as schemata and scripts, are abstractions of semantic network knowledge representation. A collection of nodes and links or slots together describes a stereotyped object, idea, or event. Frames may inherit information from other frames. Frames are similar to forms that have a title (frame name) and a number of slots (frame slots) that only accept predetermined data types. Frames are effective in expectation driven processing, a technique often used in architecture, where the program looks for expected data, based on the context [Rosenman85|.

None of the above described representation methods is ideal for describing architecture and the design process. Researchers using these methods apply existing theory from other fields to model particular aspects of design as closely as possible. Although it is possible to express particular design knowledge in other forms, such as semantic networks, predicate logic, production systems, or decision tables, all of these representations develop serious shortcomings if applied to non-trivial design problems. The reason is that design and the artifact being designed have various degrees of "softness" in the process. In the early or conceptual design stage, for example, the application of solids modeling would be too "hard" and exact a representation, whereas later in the process it is a welcome help. On the other hand, production systems that are of use in the early stages of design, supporting the user with explanations and rule-of-thumb knowledge, are of little use in a phase of design when exact analysis results are needed.

These observations lead to the search for a more flexible and less restrictive abstract representation of the architectural design process and the design artifact. The method we selected is related to the concept of object-oriented programming (OOP). OOP has two fundamental properties, encapsulation and inheritance. Encapsulation means that a user can request an action from an object, and the object chooses the correct operator, as opposed to traditional programming where the user applies operators to operands and must assure that the two are type compatible. The second property, inheritance, greatly improves the reusability of code, as opposed to traditional programming where new functionality often means extensive re-coding [Cox86]. From an architectural standpoint, object oriented programming is interesting for several reasons:

1. Objects represent data as well as operations to be performed on these data. This important property of objects is a combination of properties from geometric modeling and semantic networks. The representation of a building as an object, for example, may allow the rotation of an early concept only around the vertical z-axis, whereas a roof plane as part of the building may be rotated around all three axes.

2. Objects can represent physical objects, ideas, building functions, relations between building functions, and other real world entities. Semantic networks and frames have a similar capacity, whereas geometric modeling is less complete in this respect. In architecture, the functional diagram of a building is very important in the conceptual design phase. This diagram, developed normally from the building program, with matrices expressing relations between spaces, kinematic maps and other forms of abstractions, can be implemented as an object. The implementation of the functional module in ARCHPLAN, described in detail below, is an example for this approach. Such an object has the advantage that it can be related to other objects, that is, the error prone traditional process to translate the meaning of one representation (the kinematic map, for example) and a second representation (the adjacency matrix) into a third representation (a floor plan) is improved.

3. Objects can inherit knowledge from other objects. Class inheritance, also a property of semantic networks, allows the establishment of hierarchical and other forms of order between building elements and functional relations. This capacity is crucial in architectural design because useful spatial or functional constructs are defined once and then inherited completely or partially by other constructs on a different level of abstraction. Standard test cases are the movement of a wall containing doors and windows, and the rotation of an entire building with all its associated elements.

4. Objects can contain some form of "local intelligence". Identical messages exchanged between different objects can have different effects, and different messages exchanged between different objects can have the same effect. Through the possibility to embed decision mechanisms into each object in form of rules or type and range checking procedures, the objects can "decide" if they accept particular operations or not. The previous example of the higher degree of freedom of roof rotations versus buiding rotation applies here as well: after the building location and orientation are fixed, the degrees of freedom for the roof rotation may be reduced by a simple rule in the roof-object to the x- and y-axis.

For the above reasons, we decided to implement ARCHPLAN based on the object oriented programming

approach. Hie language is lisp, with the object-oriented extensions supplied by Hewlett Packard [Hewlett-ftickani86].

4. The ARCHPLAN Concept

ARCHPLAN is a conceptual tool for the design of high-rise buildings and has four major purposes:

- To provide a graphical feedback and representation of decision processes in the conceptual design of high-rise buildings.
- To provide a general graphical front end for a set of engineering design expert systems.
- To describe the desired attributes of a high-rise office building in different decision-making domains.
- To create a building design according to this description that will satisfy the requirements either through interactive design or partially automated or optimized decisions.

The first purpose deals with the visualization of analysis and decision processes and the implementation of an appropriate graphics package. The second purpose deals with the development of a general user friendly graphical interface. We selected the StarBase graphics package which provides Common lisp language interfaces [Starbase85). Purposes 3 and 4 deal with the implementation of a particular design application. The design strategy we chose to simulate with ARCHPLAN is that of rational decision making, which breaks down into four steps [Akin87]:

- the generation of alternatives,
- the prediction of consequences for each alternative,
- the evaluation of each alternative, and
- the selection of an alternative for implementation.

This conceptual strategy determines the ARCIIPI AN architecture and the types of abstractions needed. For interactive generation, analysis, evaluation, and selection of alternatives a modular structuring approach is best suited. These activities take place in each of the decision making domains, which are at the moment

- Site, Cost and Massing (SCM). After a site is chosen, preliminary design starts on developing a massing model that will fit a given budget. Cost and massing options arc inter-dependent variables based partially on site characteristics.
- Function. Examples for building functions are office, retail, and parking space. Fach function has particular requirements and affects the layout, appearance, and cost of the building.
- Circulation. Vertical circulation in high-rise buildings is part of a central service core or externally attached to the building. As it is a spatial and structural vertical continuum through the entire building, its size and location are important for design.
- Structure. The structural system of a building is affecting architectural expression, functional layout, and cost. In some cases, design develops after the structural system has been determined, in other cases the structural system is the result of design decisions.

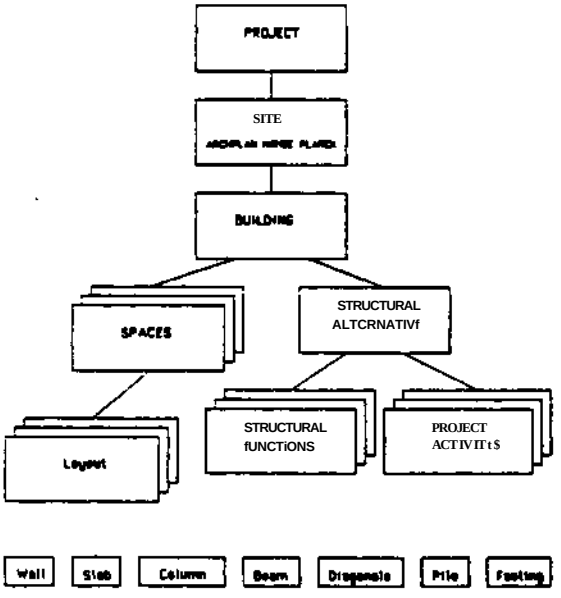
Each of these domains is responsible to a general building database, implemented as an object, whose responsibility is to maintain the high level consistency of the building abstraction, to warn the user if the consistency is violated, and to direct control to the appropriate decision making domain to correct the problem. The decision making domains are responsible for local decisions which will be of no concern to the general database unless they violate important parameters.

The overall model to simulate the decision processes is best described as prototype refinement on a global level, and of simulation and optimization on a local level. Prototype refinement means that a typical prototype for the particular building type is chosen at the beginning of the design process which is subsequently changed and refined [Gero87c]. Simulation includes operations on an abstract model of the design to predict consequences of design decisions [Schmitt86b|. Optimization involves finding optimal solutions for one or more pre-defined design parameters [Radford86|.

In the context of ARCHPLAN the above described decision making domains are implemented as four separate modules. Once the user has established a building prototype in the SCM module, all other modules can be visited and consulted in arbitrary order. Their responsibility is to refine the preliminary building description. All modules and HIRISE are accessed by selecting a menu item from the top left window provided by the user interface (see Figure 1).

In the global context of the Integrated Building Design Environment (IBDE), ARCHPLAN's main responsibility is to establish a site and architectural building description, based on client's needs (see Figure 1). This description is posted as a data base frame on a blackboard which is accessible by the engineering expert systems HIRISE, FOOTER, SPEX, and PI ,ANEX, and by IOOS1, the architectural layout generator. As an option, HIRISE can be accessed directly from ARCIIPI,AN by selecting the appropriate menu item on the top level user control screen (see Figure 1).

This section must end with a disclaimer: we are aware of the extremely complex interactions in the human design process and do not suggest that ARCHPLAN will be able in the near future to simulate or improve all of them. Therefore, the set of decisions in ARCHPI AN are a subjective selection. The criteria for selection were the ease of formalization techniques available, and the expertise of specialists in the particular areas of interest.



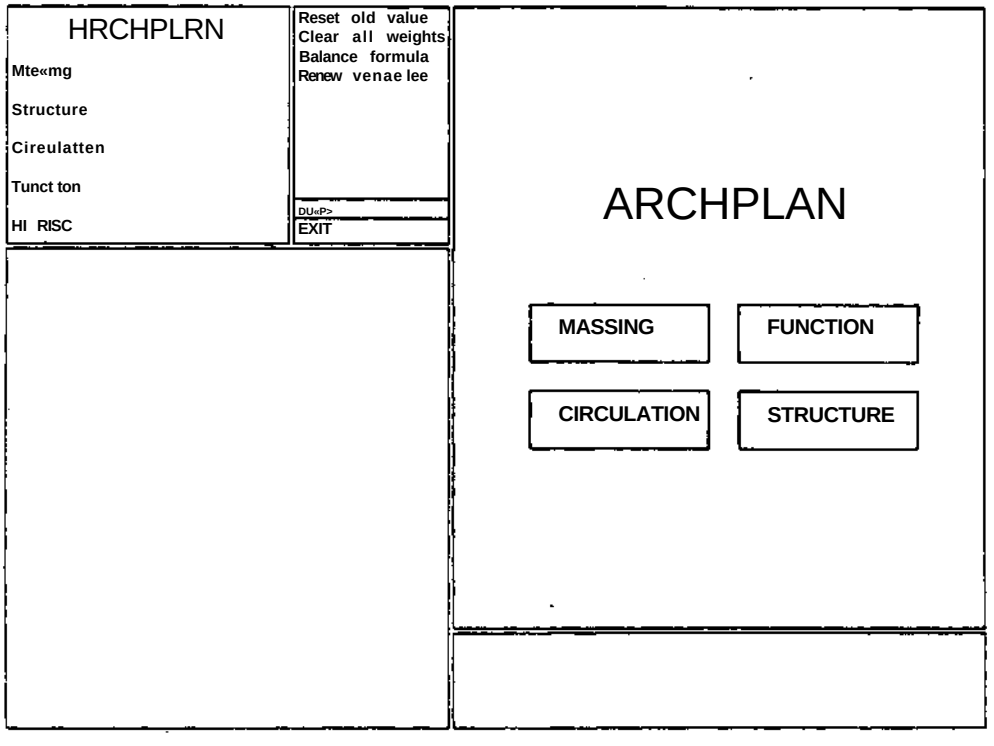


Figure 1: Top: IBDE - Integrated Building Design Environment. Bottom: ARCH PLAN modules and top level user interface. The top left window gives access to the four modules and to HIRISE. The top right window is the graphic window. The bottom left window is reserved for alpha-numeric display and interaction, the bottom right window displays messages.

5. The ARCHPLAN Modules

The four presently implemented decision making modules will be described in detail. Each of the modules contains algorithms, rules and weighting factors to determine the importance of decisions and parameters. The common abstraction for all modules is that of objects. The exchange of information is achieved through the passing of messages. This also applies for the general design description which is an object with slots for the most important building characteristics. There is no predetermined order in which the modules must be accessed and executed, which allows idiosyncratic design interaction. The exception is the SCM module, which must execute first to establish the basic parameters for the following session.

5.1 The Building Object

The general database is an object which contains information about the crucial parameters of the building and a set of actions to protect this database from becoming inconsistent through the decisions of the other modules. The object resembles a frame with slots for the various parameter values that are filled and changed according to the degree of the building design completion. Because ARCHPLAN is also producing output for the other expert systems in the integrated building design environment, the central building design description contains slots with additional information. The building description data base frame has the following (simplified) form:

```
SITE  INFORMATION
_____________________________________________

SLOT                      VALUE  (default)

site_longitude            60 degrees
site_latitude             40 degrees
site_rotation_angle       0 degrees
site_x                    300  feet
site_y                    200  feet
degree_days               6600
max_wind_load             120 raph
max_wind_dir              60  degrees

BUILDING  INFORMATION  FRAME
_____________________________________________

SLOT                      VALUE  (default)

base_x                    50  feet
base_y                    50  feet
building_rot_angle        0 degrees
structure_grid_x          (10  20  20  20  20  20
```

```
                          20 20 20 20 20 20
                          20 20 20 10)
structure_grid_y          (15 20 30 20 15)
arch_mod_x                5 feet
arch_mod_y                5 feet
ground_floor-height       18 feet
floor_height              12 feet
num_of_floors             (0 15)
core                      (70 130 35 65 0 15)
occupancy                 office
Structure_system          trussed frame
spaces                    (atrium, mechanical,
                          retail, office,
                          parking)
```

In the execution of ARCHPLAN, these default values will change based on program needs and user requirements. The building object expresses itself graphically through the interface and acts as a "read only" object. Changes may occur only through user action in the ARCHPLAN modules, ftrmanent output is produced through screen dumps and for the blackboard to be accessed by the other expert systems. Eventually, these expert systems will have a critique function and will have authority to change slots in the object.

5.2 Module One: Site, Cost, And Massing - SCM

At the very beginning of the architectural design process, decisions must be made concerning the building site, the building cost, and the basic footprint and massing of the building. While this is not the only approach towards designing a building, it is a valid initial assumption. The crucial parameters for the building site are the dimensions, the required setbacks from the site boundaries, the setback angle (city zoning laws normally require a builder to respect sun angles and daylight access to surrounding buildings), and the climate (important for energy budgets and for wind loads for high-rise buildings).

The second, and often most important, aspect is the building budget. We chose a simplified model to simulate the relations between the original, given budget, and parameters influencing the total budget. Given a certain budget, the program selects a range of possible building areas from the Means catalogue [Means87]. The total building cost is of course not only a function of the area, but also of the number of stories, the height of each story, the functions of the building, and the length and material of the perimeter. These relations are listed in the Means tables and are based on empirical data.

The user is able to set each one of these parameters manually (the allowable ranges are checked by the

program). As one option, the user may choose to optimize the building for first-cost only. As expected, the results are not very exciting, because the program will merely minimize all expensive parameters. As another option, the user can choose a cost optimization that takes into account more than one criterion. The emphasis in this option is on life-cycle-cost which is influenced by factors such as user satisfaction and maintenance costs (up to 92% of an office building's cost over its life time consists of the occupant's salaries; therefore absentee rates caused by user dissatisfaction have a substantial negative impact on the financial success of a building). Due to the difficulty of quantifying relations between user satisfaction and the building's physical appearance, this option is highly hypothetical, but acts as an interesting testing ground for the integration of qualitative and quantitative criteria.

Based on the initial parameters, constraints, relations between parameters, and the allowed actions (see below), the program then displays the preliminary massing of the building on the site, together with the parameters that influence the massing (see Figure 2). The parameters are shown as normalized bar graphs, varying from the lowest to the highest acceptable level.

In the ARCHPLAN implementation, the SCM decision module is an object that contains the following parameters:

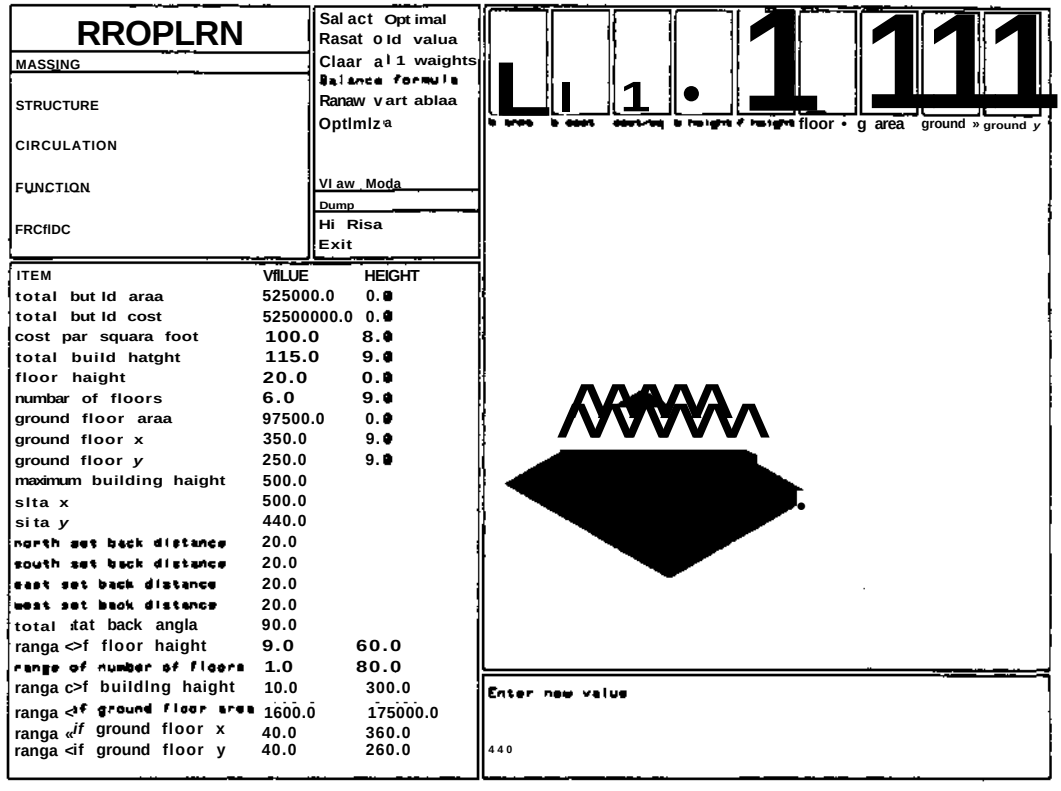
Variables:

```
Total Building Area (ranging from 5,400 to 1,000,000 square feet).
Ground Floor Area (ranging from 8,100 to 160,000 feet).
Total Building Cost (ranging from $432,000 to $200,000,000).
Cost Per Sqft (ranging from $80 to $200 per square foot).
Total Building Height (ranging from 9 to 1,000 feet).
Number of Floors (ranging from 1 to 100 floors).
```

Constants:

```
Site X (the east-west length of the site, ranging from 90 to
400 feet)
Site Y (the north-south length of the site, ranging from 90
to 400 feet)
Site Area (ranging from 22,500 to 160,000 square feet)
North Setback (ranging from 0 to 100 feet)
East Setback (ranging from 0 to 100 feet)
South Setback (ranging from 0 to 100 feet)
West Setback (ranging from 0 to 100 feet)
Setback Angle (ranging from 45 to 135 degrees)
Maximum Building Height (ranging from 13 to 1,000 feet)
```

The differentiation between variables and constants is flexible, i.e., through the use of weighting factors from



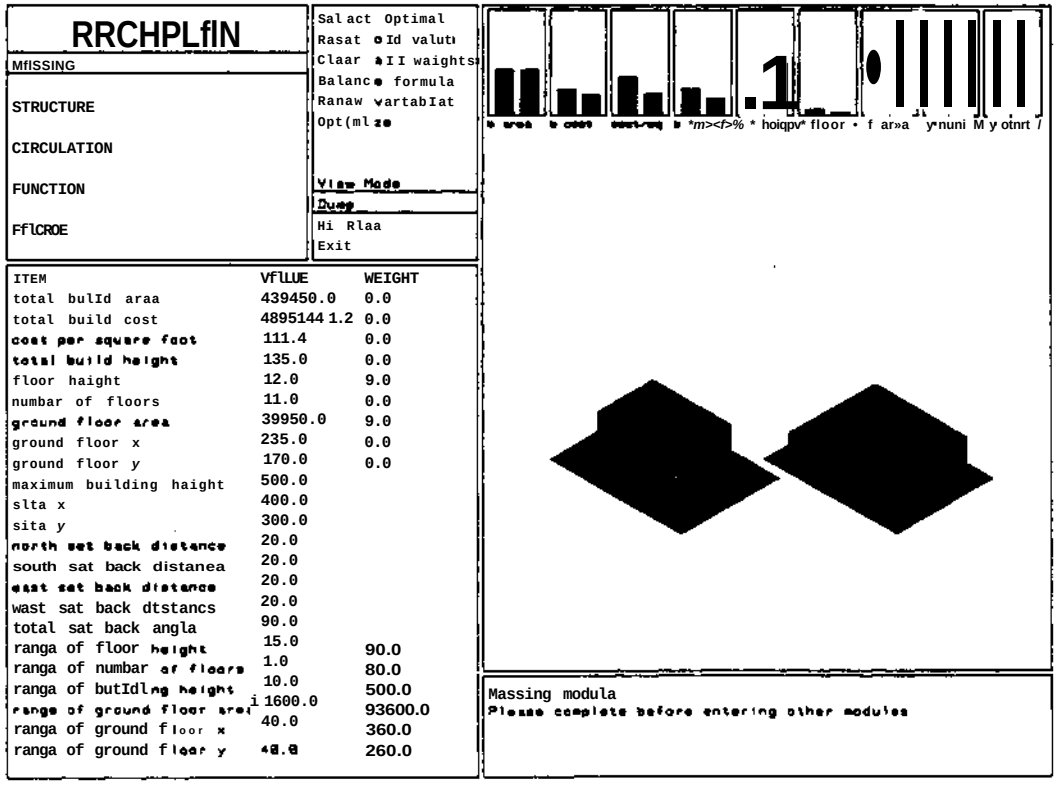
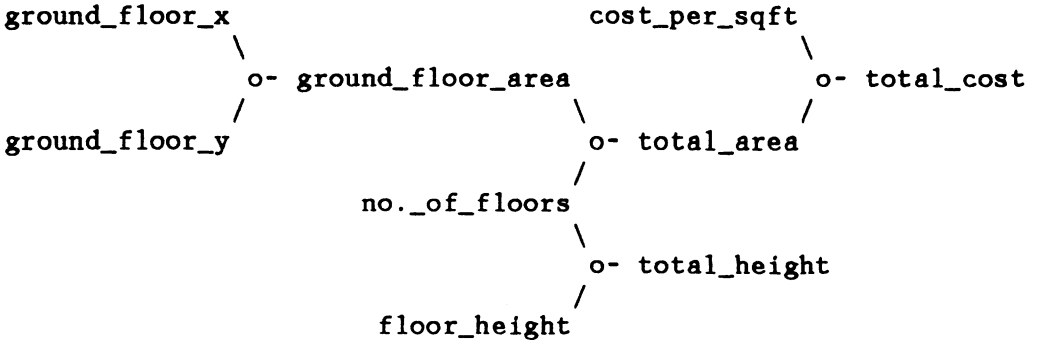


Figure 2: User view of the SCM module. Top: a 525,000 sqft building. The user supplied weighting factors for cost, building height, and the building footprint. Bottom: a smaller building. Right: optimized for minimum cost, Left: user defined.

1 to 10 (1 for least commitment, 10 for highest commitment), some variables are de facto transformed into constants. The constants listed are also represented in the database object and are constraints that are established at the very beginning of the process. They can only be changed if absolutely necessary. The SCM module allows the interactive editing of a set of default parameters. The most important building parameters are organized as objects in an activation network [Brownston85]. In an activation network, each node represents an object and each arc represents a relationship between two objects. If the arc is labeled, the label is a number indicating the strength of the relationship. When a node is processed, its activation level may change, and the effects of the change are propagated along arcs to related nodes, resulting in changes to their activation level. The SCM module can be expressed as an activation network of the following form:



The objects in the above activation network communicate with each other by sending and receiving messages. When an object receives a message, it consults its data base and the appropriate rules to decide what action to take. The rules may be stored directly with the object or in a different object. In ARCHPLAN, the result of any change is represented numerically in the related change of other variables, and graphically in the change of the normalized bar charts and the massing of the building.

5.3 Module Two: Function

The distribution of different functions in a building is of crucial importance to the appearance and performance of the structure. It could be argued that the functional, three-dimensional layout is the first design decision to be made. However, a close look at the design practice suggests that the functions are less form-determining in the conceptual design phase in the majority of modern high-rise buildings than the parameters dealt with in the SCM module. This observation also coincides with the global strategy of prototype refinement.

The program is capable of handling five different building functions:

- office
- retail
- atrium
- parking
- mechanical

Circulation, a building function in close relation to all of these, is treated in a separate module.

The Function module assists in the vertical and horizontal distribution of the different building functions within the basic massing volume (see Figure 3). Since this module relies heavily on built-in heuristics, user input is restricted. The decisions are made and reflected locally, unless the constants in the global building description object are violated. In this case, the program backtracks and control is passed back to the SCM module. In the SCM module, the user can choose either to automatically adjust the design description to the information received from the Function module, or make changes manually.

In a typical session, the user selects the Function module from the previous screen and makes the Function window current. The program then presents a chart with the five available functions and allowable percentages. Certain constraints apply:

- office space (ranging from 80% to 100% of net square footage)
- retail space (ranging from 0% to 20% of net square footage)
- atrium space (ranging from 0% to 10% of net square footage)

The sum of office, retail, and atrium space is always 100% of the net square footage. The mechanical floor is at least 5% of this area (typically, one mechanical floor every twenty stories, or at the top of the building for less than 20 floors). Parking is presently placed underneath the building, at the rate of one parking floor per seven building floors.

The program starts by checking the slots in the central database and assigning the percentages for each function by built-in knowledge. The user can also change the default percentages by graphically moving the

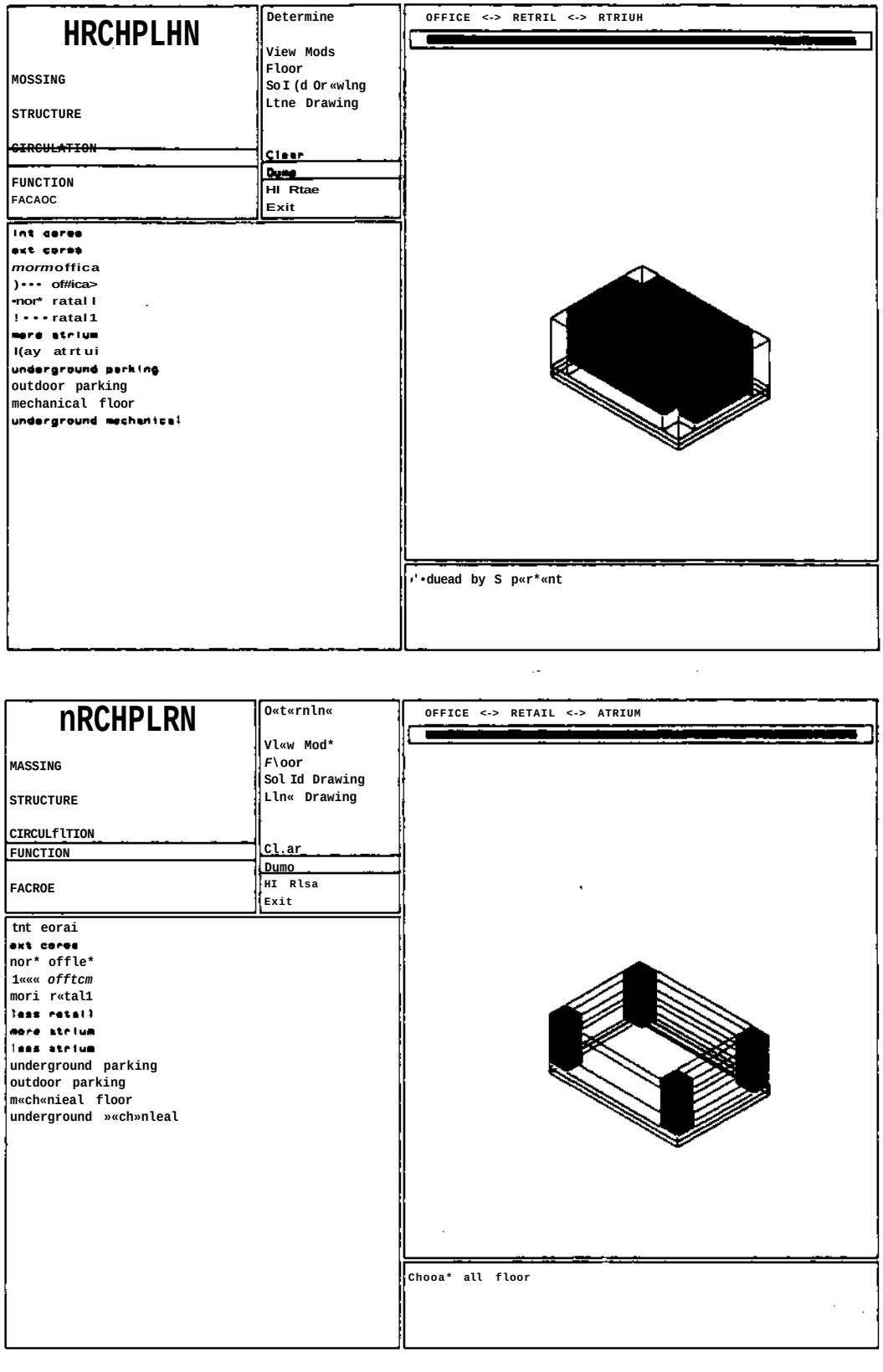


Figure 3: User view of the Function module. Top: wireframe representation of the building, office area displayed as a solid. Bottom: elevator and service shafts.

bars that represent them. Built-in knowledge is used because in the SCM module no functional decisions are made. Examples of this knowledge, in the form of design advice, are:

- Start by dividing the total volume into 70%office space, 20%retail space, and 10%atrium space.
- Start by placing retail at the ground floor and office above.
- If the building is high, place the atrium on the lower level
- If the building is low, develop it from the top level down
- Do not run a service shaft through the atrium if the atrium is at the top of the building.
- Explore several options of combining office and retail three-dimensionally: ground floor only office, ground floor only retail, ground floor office and retail.

The rules are contained in "advice-objects", which give advice to control objects that modify the Function module object. The knowledge in the advice objects is quite limited at the present; the intention is to develop an interactive advice object that learns through induction from direct user input and from the frequency of user choices of particular functional arrangements. The advice objects send messages to the Function module object which expresses itself graphically, the Function module object also checks with the building object for conflicts in the two databases. If they are discovered, and are substantial, the user is prompted to resolve the problem on the Function module level. If the inconsistency produced by user choice or action in the Function module is substantial and the user refuses to resolve it on this level, the program returns to the SCM module and corrects the problem there, giving the user feedback how the previous decision influenced height, cost, massing, and the other parameters.

The Function module produces three-dimensional output and interactively highlights functions to better understand their distribution in three-dimensional space (see Figure 3). The Function module also produces output for CORE and SPACE, the planned generative expert system for the design of core and space layouts [Flemming86b]. It is planned that CORF and SPACE will accept the two-dimensional plan information from ARCHPLAN and begin the individual layout of the functional spaces which ARCHPLAN only produces as big building blocks.

5.4 Module Three: Circulation

Circulation in high-rise buildings addresses the problem of moving occupants and equipment from floor to floor and within floors, and to guarantee the safe evacuation of the occupants in emergencies. Circulation is not only a transportation and evacuation problem, but has a major impact on the internal functioning and on the architectural expression of a high-rise building. The two extreme cases for the placement of vertical circulation are the completely internal (service and elevator core in the center of the building) or the completely external solution (service and elevator cores attached to the outside of the buildings). Most

high-rises have vertical circulation systems that lie in between those two extremes and. therefore ARCHPLAN concentrates on creating vertical circulation proposals based on variations of these two prototypes.

The Circulation module is accessible as soon as the SCM module has established a "base case" building. If Circulation is started as the second module, then the Circulation object inherits the existing data of the building object. Supplied with this knowledge, the program starts to present the list of parameters which influence the location and size of the circulation cores. If Circulation is accessed as the third or fourth module, then it inherits the additional decisions that were made in the previous modules. The choice of a location for the circulation core is important, as it affects decisions about structural system and function distribution. Several issues play a role in the determination of the location, size, and number of the vertical circulation. ARCHPLAN considers the following factors:

```
available exterior window space --> east
available exterior window space --> west
available exterior window space --> south
available exterior window space --> north
minimize total surface area
intention to add new adjacent building --> east
intention to add new adjacent building --> west
intention to add new adjacent building --> south
intention to add new adjacent building --> north
require external stair case (fire escape security)
visual interest image --> east
visual interest image --> west
visual interest image --> south
visual interest image --> north
equal distance access to cores
increased elevator accessibility
flexible tenant distribution
deep office space
fixed multitenant occupancy
structural simplicity
circulation takes lateral forces
```

For this module, it is particularly important to make the inference process the program uses as transparent as possible and consequently graphically present the above parameters that influence the decision of the circulation location (see Figure 4). As in the SCM module ARCIIPI AN uses weighting factors to represent the relative importance of one parameter. In this case, however, the parameters have no absolute values. A comparison of the two modules explains the reason. An example from the Circulation module:

- A deep, uninterrupted office space is very important (weighting factor 10 is assigned by sliding the bar graphically to the right).

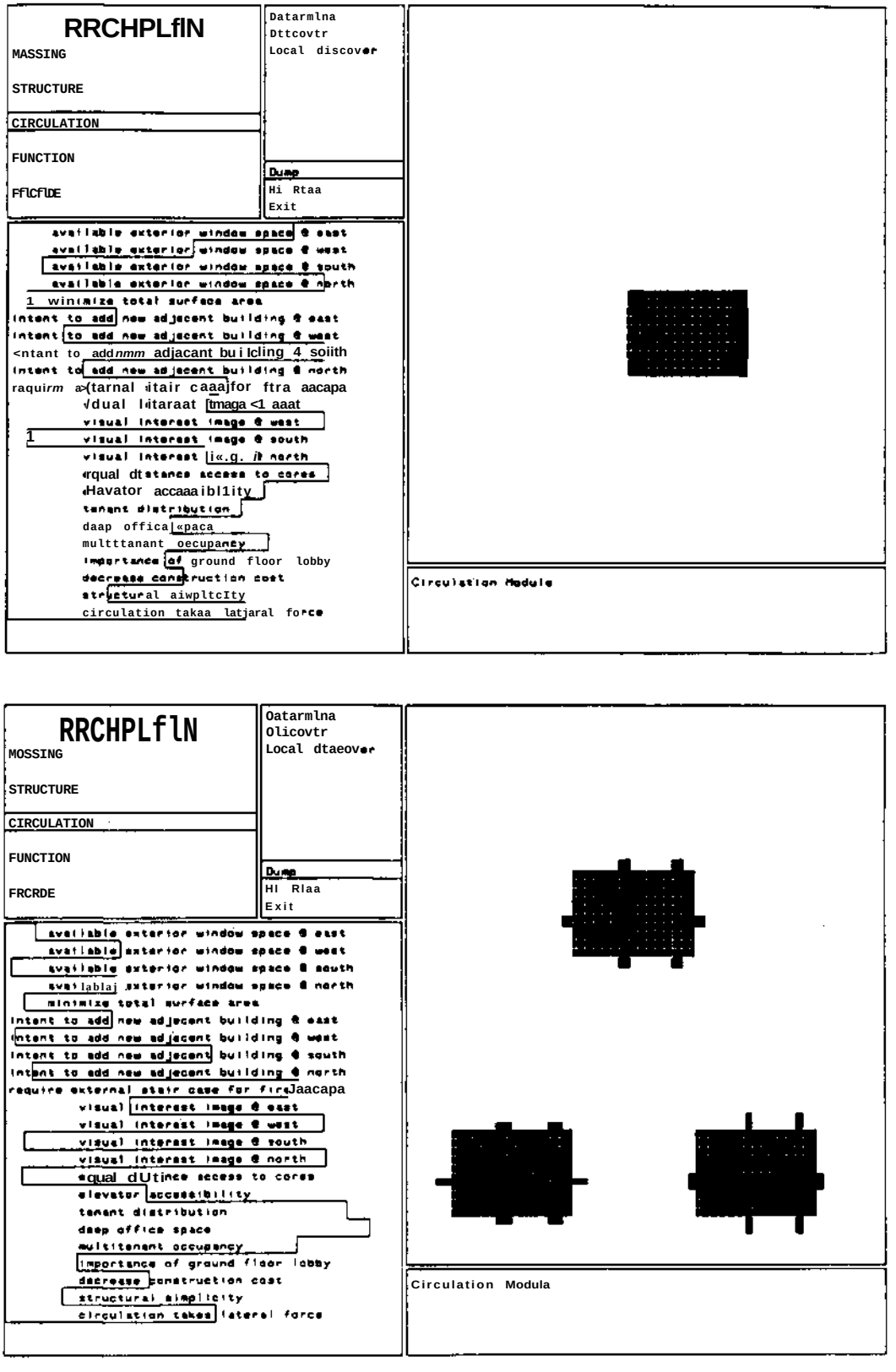


Figure 4: User view of the Circulation module. Different two-dimensional layouts are shown as a result of user input through sliding bars. Sliding the bar for a parameter from left to right increases the relative importance of this parameter.

- A deep, uninterrupted office space is not necessary (weighting factor 0 is assigned by sliding the bar graphically to the left).

An example from the SCM module:

- The total building budget is $25,000,000, and it must not be exceeded (the user enters 25,000,000 and a weighting factor of ten numerically by typing over the default numbers)

- The total building budget is $25,000,000, but other factors may be more important (the user enters 25,000,000 and a weighting factor from 0 to 5 numerically by typing over the default numbers)

Besides exploring the behavioral difference of parameters with absolute values and weighting factors and factors with relative importance only, we were also interested in the user reaction to the two different input modes. First results show that offering graphical interaction with sliding bars leads to about three times more experimentation than the strictly numerical interface.

In a typical session, the user starts by first examining the above parameters which are all set to default values. Two options are available to see the program's proposal for the location and configuration of the circulation: discover (the equivalent to forward chaining) and determine (the equivalent to backward chaining). The options normally produce distinct solutions for size, configuration, and location of the vertical circulation, represented in two-dimensional floor plans. The user can also start by changing the value of the parameters immediately and so produce a large set of possible circulation layouts.

In case of conflict with the building database (the Function module may have assigned the elevator in the center, the Circulation module on the outside), the program will try to first solve the discrepancy on the level of the conflicting module, in this case the circulation module. If the conflict cannot be solved, the program backtracks to the SCM module where the central building description can be adjusted manually or automatically. Changes from this adjustment are propagated to the other modules.

5.5 Module Four: Structures

All design decisions in the previously described modules have an impact on the type and performance of the building's structural system. An architect interacting with ARCHPLAN will probably not start with the structure module, whereas an engineer might want to see the impact of the building's form on the structural system and vice versa. Both approaches are possible, as the Structure module is directly accessible after the SCM module.

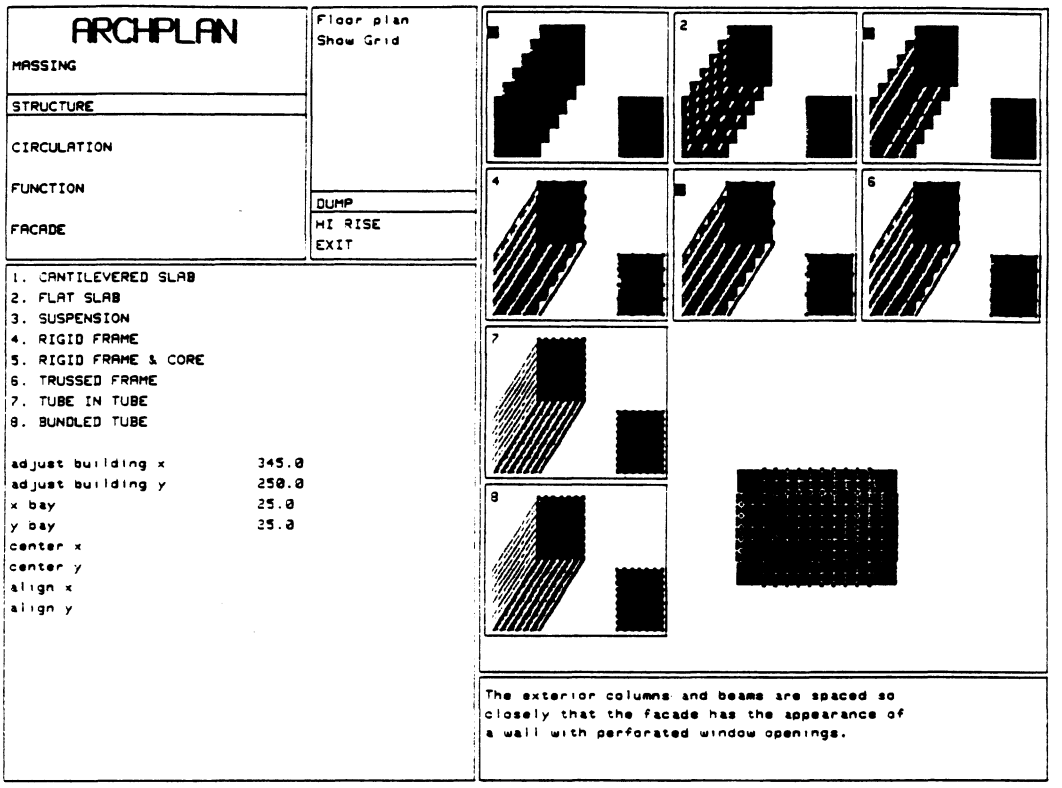


Figure 5: User view of the structure module. A maximum of eight different structural types is offered, if the selection has not been restricted by previous constraints. The options which are blotted out, in this case 1, 3, and 5, should not be chosen.

This module is intended to give the designer an overview over possible structural types appropriate for the building design (see Figure 5). The synthesis of a structural system for a design developed with ARCHPLAN is reserved for the HIRISE structural design expert system [Maher84]. The Structural module considers at the moment the following structural systems:

- Cantilevered slab
- Flat slab
- Suspension
- Rigid frame
- Core & rigid frame
- Trussed frame
- Tube in tube

- Bundled tube

If the building object has been defined through the previous design decisions, the options are limited. If the Structure module is executed early in the design process, the set of selectable structural types is larger. After the user has accepted the proposed structural type for the given building, or has made an independent choice, the structure is displayed three-dimensionally for the current building object. The program solves conflicts that may arise out of the user's choice in the same manner as in the other modules.

6. Critique and Future Developments

ARCHPLAN is incomplete at this point and serves as a testing ground for different design methodologies and their computational representation. We expect not one final method, but a combination of methods for different design applications and design stages to emerge as the optimum. ARCHPLAN uses a spatial representation closely related to that of HIRISE which restricts it at the moment to rectangular structures. The implementation of ARCHPLAN in Common Lisp and its object-oriented extensions is advantageous in terms of programming and experimentation. The production of a transparent and friendly user interface is a separate project of importance for the practical application of ARCHPLAN. Based on these and other critical remarks, the following developments are planned:

- Improvements in the flexibility of the module structure.
- Addition of optimization routines where possible (existing presently only in the SCM module).
- Addition of explanation modules ("Why" and "How" options).
- Addition of a decision history option for future induction purposes.
- Exploration of design creativity in the framework of ARCHPLAN.

Some of these problems, such as explanation and decision history, can be solved without further investment of research work, as ARCHPLAN is now being translated in a commercial expert system shell (ESE) which allows access to external functions and offers extensive interactive user interface support.

7. Conclusion

ARCHPLAN has proven to be a valuable framework for the testing of design ideas and their representation in a workstation environment. Simplified representations of existing high-rise buildings, such as the Lloyds of London building in London, England, the Bank of Hongkong offices in Hongkong, and the Fifth Avenue office building in Pittsburgh, Pennsylvania, can be generated with ARCHPLAN as test cases. The test cases provided an invaluable tool to develop and test the knowledge base. Knowledge is represented in two forms: as algebraic relations and as as rules, both embedded in the object-oriented programming environment.

The project demonstrated the importance of real time graphical feedback for knowledge based architectural design systems. The object-oriented programming approach applied to design and graphics problems is powerful and on a level of abstraction that is closer to the human designer than traditional programming approaches. ARCHPLAN showed that hybrid programs - being part knowledge based systems, part traditional algorithmic programs - can be realistic architectural design tools. The most valuable effect was to gain new insights into the design process through the necessary formalization of design knowledge and decision mechanisms in each of the ARCHPLAN modules. This experience also suggests that future design programs will have extensive idiosyncratic characteristics.

At the moment, ARCHPLAN is a design assistant to produce meaningful high-rise building design descriptions that are used by engineering expert systems and to compare manually designed buildings to those designed with ARCHPLAN. Future program development has two main emphases: one is increasing design automation and optimization on a global level in producing feasible high-rise design solutions. The other is refining local decision making in particular design aspects such as building circulation and functional distribution. Along with this development in which the system is now "learning" from existing design test cases, cost tables, and personal design experience, its future role will be that of a design tutor which could teach and explain design to novice users.

Acknowledgements

The author would like to thank his research assistants and programmers Chia Ming Chen, Chen Cheng Chen, Shen Guan Shih, Richard Cobti, and Jeffrey Kobcrnick. Special thanks to Professor Steven Fenves and Michael Rychener for their advice.

References

[Akin87] **Akin, Omer, Hemming, Ulrich, Schmitt, Gerhard, and Woodbury, Robert.**
Envelopment of Computer Systems for Use in Architectural Education.
Architecture Research Series, Department of Architecture, Carnegie Mellon University, March 1987.

[Brownston851 **Brownston, Lee, Farrel, Robert, Kant, Elaine, and Martin, Nancy.**
Programming Expert Systems in OPSS.
Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1985.

[Cox86j **Cox, Brad J.**
Object Oriented Programming.
Addison-Wesley Publishing Company, Reading, Massachusetts, 1986.

[Eastman75] **Eastman, Charles.**
The Use of Computers instead of Drawings in Building Design.
Journal of the American Institute of Architects 3:46-50, 1975.

[Eastman77] **Eastman, Charles, and Henrion, M.**
GLIDE: A Language for Design Information Systems.
In *Proceedings of the 1977 SIGGRAPH Conference,* pages 24-33. **SIGGRAPH,** 1977.

[Flemming86bJ **Hemming, Ulrich, Rychener, Michael D., Coyne, Robert F., and Gl avin , Timothy J.**
A Generative Expert System for the Design of Building Layouts.
Technical Report, Engineering Design Research Center, Carnegie Mellon Univers ity, 1986b.

[Gero87cJ **Gero, John S., Maher, Mary Lou.**
A Future Role of Knowledge Based Systems in the Design Process.
In Wagter, Harry (editor), *CAAD Futures.* ECAADE, Eindhoven University of Technology, The Netherlands, May, 1987.

[Hewlett-Packard86|
HP 9000 Series 300 Computers LISP Application Notes
Hewlett-Packard Company, Fort Collins, Colorado, 1986.

[Maher841 **Maher, Mary Lou.**
HI-RISE. A knowledge-based expert system for preliminary structural design of high-rise buildings.
PhD thesis, Carnegie Mellon University, 1984.

[Mclntosh82) **Mclntosh, Patricia G.**
The Geometric Set of Operations in Computer-Aided Building Design.
PhD thesis, University of Michigan, 1982.

[Mclntosh841 **Mclntosh, John F.**
The Application of the Relational Data Model to Computer-Aided Building Design.
PhD thesis, University of Michigan, 1984.

[Means87] **Horsley, William F.**
Means Systems Costs 1987.
Robert Sturgis Godfrey, 1987.

[Minsky75| Minsky, M.
A framework for representing knowledge.
In Winston, P. (editor), *The Psychology of Computer Vision*. McGraw-Hill, New York, 1975.

[Radford86] **Radford,** Antony D., and Gero, John S.
Designbyoptimizationinarchitectureandbuilding.
Van Nostrand Reinhold Company, New York, 1986.

[Rosenman85] Rosenman, Michael A., Gero, John S.
Design codes as expert systems.
CAD Computer Aided Design 17(9): 399-409, November, 1985.

[Schmitt86bj Schmitt, Gerhard.
Expert Systems in Design Abstraction and Evaluation.
In Kalay, Yehuda (editor), *The Computability of Design*. John Wiley & Sons, New York, 1987.

[Starbase85J Hewlett-Packard Company.
StarbaseReference
2 edition, Hewlett-Packard Company, Fort Collins, Colorado, 1985.