An experimental design system for the very early design stage

B. de Vries
A.J. Jessurun
Eindhoven University of Technology
Design Systems Group (www.calibre.bwk.tue.nl)
The Netherlands

ABSTRACT

The concepts of the experimental design system that are discussed are feature modeling and geometrical constraints. The main technique for creating the user environment is Virtual Reality. Feature modeling forms the basis for managing the design data. To start with, data storage is implemented in a Relational Data Base Management System. Along with this a (traditional) interface is developed for managing the data. Data management consists of feature type creation and feature type instancing. Features are used to define building elements, their relationships and additional constraints. Apart from the design data, geometrical data are stored. Possible design solutions can be limited using geometrical constraints. Specifying connection types between building elements result in a set of solutions for the position of the bounding boxes of the building elements in space.

1. INTRODUCTION

At the Eindhoven University of Technology a research program called VR-DIS (Vries 1997) is established to pursue a innovative multidisciplinary design system. We believe that VR as a User Interface will probably replace many of the existing techniques due to the (extra) possibilities which are available to communicate intuitively between man and machine. Especially in the architectural environment the challenge lies in developing a new work environment in which the design process can take place (Vries and Achten 1998).

A case study of a specific architectural design process has been done to ensure that the developed design system supports design actions at least for that case stage and probably also for other design tasks. First the case is described informally simply registering all design actions and design output. Secondly the case is described formally using a specification language that has especially been developed for these purposes. As a result a formal description of the design model becomes available that can be implemented using modern software engineering techniques.

In this paper conclusions will be drawn from the case description about the requirements of a design system for the very early design. Two theoretical concepts are introduced for managing design information and design support. An experimental design system will be constructed that supports the design process in a 3D environment using VR as a technology. The user interface will be described and the underlying design information structure. Concluding the first experiences with the design system will be discussed.

2. CASE ANALYSIS

The case describes 31 design phases. For an in depth explanation of the case see (Achten and Leeuwen 1998). The experimental design system described in this paper focuses on the first part of the case. In this part of the design process the requirements that have been elicited by the principal are 'translated' into a design containing the main building elements and spaces. Primary goals of this design phase are:

- check if all requested rooms and spaces fit on the building site
- check in communication with the principal if the requirements list is complete
- get an first impression about possible orderings of rooms and spaces
- get a first impression about the layout of the plan
- get a first impression of the facade including windows, doors and material outlook
- get a first impression about the possibilities of the roof shapes

To capture the design process that is taking place, the subsequent phases will be analyzed towards the followings aspects: buildings elements, relations between building elements and conditions that must be met by building elements, presentation of the design information and finally the design process moving from one phase to the next.
Excluded from the analysis is all information that is related to the administration about the project such as the name of the customer and the status of the project.

In general, requirements can be in conflict with each other (e.g. all rooms on the ground floor whilst there is not enough space on the building site). Therefore some requirements may need relaxation during the design process. Not all requirements are described explicitly in the brief (e.g. the living room should be on the ground floor). They are filled in by the designer and checked afterwards with the principle.

2.1 Building elements

The principal expresses his/her wishes using nouns to specify rooms (e.g. kitchen) and verbs to specify activities that will take place (e.g. living). Dimensions or floor areas are left to the designer to fill in. For some rooms the interior is described in more detail (e.g. the bathroom should contain a shower, a bath, a toilet and two wash basins). The principal has some explicit wishes about the material for particular building elements (e.g. brick walls and a wooden construction in the veranda).

2.2 Relations and Conditions

For some spaces and rooms the location is restricted in relation to other spaces or rooms (e.g. veranda is an extension of the living room). For some building elements additional conditions are specified (e.g. access of a room, material of the floor). The locations of the walls of the rooms are restricted to a grid that is chosen by the designer. For the main building volumes two grids are used, one of them slightly rotated.

2.3 Presentation

Only the first contact with the principle is documented containing the main list of requirements. From that stage on all design information is reflected by using drawing techniques. The first sketches are created to check the total floor area of the plan related to the area of the building site. At the same time some room ordering solutions are investigated to find a cluster of rooms that can be combined into one building volume. Of some building variants the outlook of the facade and the shape roof are presented.

2.4 Process

During the considered design process sketches and drawings are produced to get more insight about the composition of rooms and the shape of the main building elements related to the activities that will take place. There is no strict order in which the plans are produced, but rather is a rapidly shift from analyzing one aspect after another.

3. FUNCTIONAL SPECIFICATION

To describe a design system which supports the depicted design phase of the case, the system functions are specified looking at the following aspects: building element types, operation types, relation and condition types and presentation.

3.1 Building element types

The kind of building elements that are manipulated by the designer can by classified into the following categories:

1. Spaces
   Spaces are elements in which people stay or live and that are partly enclosed by physical boundaries or that have no physical boundary at all. (E.g. hall, dining place)
2. Structures
   Structures are all elements which are not spaces and that do have physical boundaries. (E.g. floor, column)
3. Infill
   Infill components are structures who are considered not to be part of the permanent structure of the building. (E.g. stair, chair)
4. Ground
   Elements belonging to the ground category are those elements that are part of the environment of the building. (E.g. garden, building site)

3.2 Operation types

Building elements can be created, moved, rotated and scaled. The shape of the rooms is rectangular as a starting point. The shape can be adjusted to fit with other shapes. This means ‘wrapping’ a shape around another shape or penetrating one shape into another.

Roofs of different shapes can be placed on top of the rooms in case of studying the facades or the building volumes. Floors are generated fitting the room sizes.

A control mechanism is required for locating one building element relative to another one. (e.g. one room next to the other). This functionality is especially required for those cases in which the principle expressed this kind of relationship. Implicit relationships between building elements supports the designer in positioning a new building element related to existing ones (E.g. surrounding a space by adding walls)

Building elements can be snapped to a (active) grid. The systems 'senses' the neighborhood of a grid line and will position the element according to the grid.

Elements can be grouped together to perform a single operation on a group of elements (e.g. changing the shape of all roofs)

Material properties can be attached to selected building elements.

3.3 Relation and Condition types

Geometrical relationship types that can be recognized between the building elements are:

1. space *adjacent* to ground (e.g. veranda adjacent to garden)
2. space *in* space (e.g. sitting place in living room)
3. space *adjacent* to space (e.g. kitchen adjacent to garage)
4. infill *in* space (e.g. toilet in bathroom)
5. space *on* top of space (e.g. roof on garage)
6. space *on* top of ground (e.g. first floor on building site)
7. structure *align* space (e.g. walls of the living room)

A special kind of geometrical relation type is the restriction of the location of walls on grid lines.

The access condition type is translated into geometrical relations during the design. (E.g. by locating two rooms next to each other and adding a doorway). Conditions that reflect only one building element are described as part of the specification of that building element. (e.g. outer walls are made of brick).

3.4 Presentation

The kind of presentations that should at least be supported are plan view, perspective view and facade view. Rooms can be identified by displaying a description. Secondly the use of rooms can be recognized by placing the appropriate components in it (e.g. a bed). In general the designer should be able to have a complete overview of the building as well as have some impression about the dimensions of rooms. Already in this stage the infill of the wall openings of the outer walls (door and windows) plays an important role, in particular their size and location. The material requirements as far as they are known should be visible.

The grid can be made visible and active using a toggle. To get a view on the layout of the floor plan, cross sections of all building elements are generated at a standerd height above a floor.

4. CONCEPTS

Two theoretical concepts have been selected to support the management of design data and the manipulation of geometries. Both concepts, Features and Constraints, will be discussed briefly. For an extensive explanation see (Leeuwen and Wagter 1998) and (Kelleners, Veltkamp and Blake 1997) respectively.

4.1 Features

Feature Based Modeling finds its roots in Mechanical Engineering. In this discipline Features are almost synonymous with Form Features. Form Feature descriptions are closely related to manufacturing of these forms. In an almost finished PhD study at the Eindhoven University in the Design Systems Group (Leeuwen and Wagter 1998) the original concept has been applied to architectural design. Main purpose was to investigate if the concept could support flexibility and extensibililty.

Flexibility is found to be a key characteristic of the architectural design process. The focus in architectural design shifts between different levels (e.g. urban plan, building, room) and different aspects (e.g. physical, structural). Design decisions are taken in a cyclic process focusing on a part of the complete design task (e.g. the ground floor). Thus, changing a design and its underlying representation is in fact part of the creative process. Moreover design decisions are often dependent on each other. Therefore the design representation must support the linkage of characteristics of design parts (e.g. the color). In that case changing a parameter of one building element is propagated to another (e.g. all internal doors share the same color). In Feature-Based Modeling any Feature (characteristic) of a building element can be shared by any other building element. For this purpose three relationship types are introduced: Specification, Decomposition and Association. Specification specifies a certain characteristic of a Feature (e.g. color). Decomposition declares a building element being a part of another building element (e.g. hinge is part of a door). Association is any relation that cannot be expressed by the former two (e.g. a door in wall relation).

The second key characteristic is the fact that a design evolves in time. Descriptions of building elements are often rather abstract and incomplete in the early design stages. Gradually design parts are refined and design alternatives are accepted or rejected. Sometimes the design process is reversed when reconsidering previously found solutions. All the time new design decisions must be checked for consistency with other design decisions. The representation must support registration of design solutions of the same building element in different stages. So design solutions should never be destroyed unless the designer explicitly wants to do so. In Feature-Based Modeling, Features can be related with each other using the Inheritance relationship. Inheritance specifies a subtype having all the characteristics of the supertype and if necessary one or more

extra. Moreover each Feature is time stamped and carries the name of the creator (an identification of the designer).

A third key characteristic of architectural design is the fact that most designs are unique in themselves. Therefore describing a generic model of a building appears to be an impossible task. Instead of that in the proposed Feature-Base-Modeling theory, Features can be created on two levels: Feature Type level and Feature level. On Feature level Feature Types are instanced into Features. The designer is offered libraries of predefined Feature Types, but is also allowed to create his/her own Feature Types or Feature Sub Types. In this way standardization can be pursued by the creation of libraries without limiting the designer and refraining him/her of new innovative design solutions.

4.2 Constraints

Reasoning about geometrical relationships between building elements in general requires a common formal geometry description. In VR systems geometry representations for the surfaces of shapes usually consist of lists of triangles (strips). Computer hardware is optimized for displaying these triangles. Ideally constraints also would use the same triangles as primitives. As a start though, we found that Bounding Boxes containing building elements serve well as far as we deal with spatial relationships. Nevertheless there are some limitations it helps us to focus on the development of algorithms for expressing spatial relationships instead of solving problems that have to do with the complexity of geometry descriptions. The theoretical approach we adopted for managing geometrical constraints is based on Allen's temporal interval algebra (Allen 1983). He recognizes five basic relationships that can exist between a point and an interval: ahead, front-touch, in, back-touch and behind. Because in our design system each of the local axes of the Bounding Boxes representing the building elements are always parallel with one of the global axes, it is appropriate to use interval-interval relationships. This lead to 13 relationship types. (For a more in-depth explanation about 'Qualitive Spatial Relationships', see (Gorti and Sriram 1995)). Constraints on a Bounding Box can be rewritten to relations between intervals in three sets, one for each axis.

In the constraint solver (Kelleners, Veltkamp and Blake 1997) five constraint types are recognized:

1. Connection constraints: Touch, AlignSides
   These are constraints that show Bounding Boxes are connedted with each other.
2. Distance constraints: OppDistance, LatDistance
   These constraints specify the distance between two sides of Bounding Boxes.
3. Contain constraint: Contains
   This constraint specifies that one Bounding Box should contain another.
4. Non-intersection constraints: NonIntersect
   This constraint specifies that two Bounding Box should not intersect.

5. Unary constraints: FixPosition, FixSide, FixDimension, FixOrientation, MinDim, MaxDim
   These are constraints that put restraints on the position, the dimension and the orientation of the Bounding Box. These constraints are no interval constraints but necessary for the solution process.

For example, a Contains-constraint which specifies that Bounding Box B2 should hold Bounding Box B1 can be rewritten to the following interval constraints:

$I_b^{B1} < I_e^{B1}$ and $I_b^{B2} < I_e^{B2}$ and $I_b^{B1} < I_b^{B2}$ and $I_e^{B2} < I_e^{B1}$ and $I_b^{B1} < I_e^{B2}$ and $I_b^{B2} < I_e^{B1}$.

Here, I_b^{B} and I_e^{B} define an interval belonging to Bounding Box B. The subscripted $_b$ en $_e$ mean the beginning and the end of an interval. For the Contains-constraint the above six constraints are created for all three axes.

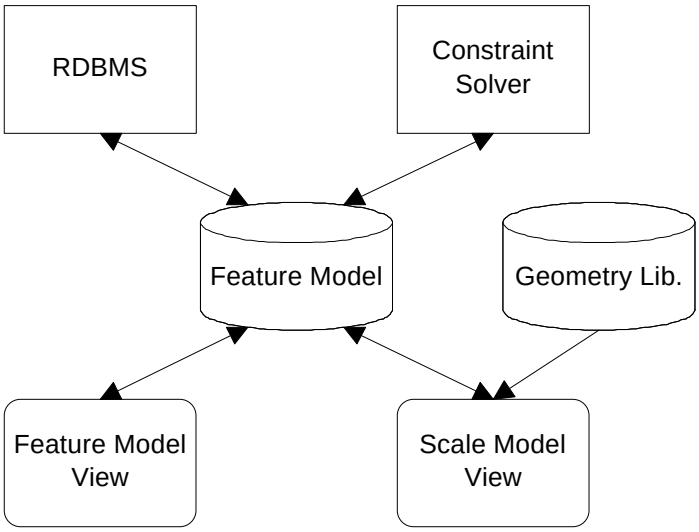
Since the purpose of the design system is that constraints can be created between building elements rather intuitively there is no control mechanism to check whether the geometrical model is under-constrained or over-constrained. In an under-constrained situation the solving process will produce a large number of solutions that is hardly manageable by the designer. In a over-constrained situation the solving process will find no solution at all. To deal with this problem a solution method is chosen that will always calculate a solution that is ‘as close as possible’ to the former situation. This method also better relates to design support where an architect implicitly defines constraints between building elements by placing them as he does during the actual use of the system.

5. IMPLEMENTATION

5.1 System architecture

The Feature Model View uses a Relational Data Base Management System for managing the Feature Model data. The Scale Model View directly reads the geometry data from the Feature Model store and the Geometry Store. The constraint solving process is triggered from the Scale Model View.

Figure 1: **System Architecture**



5.2 Geometry

The geometry of spaces and structures will be generated from the data in the Feature Model. Infill components reference to a geometry file.

5.3 Constraint solver

The geometrical relations between building elements will be implemented using constraints. To manage the constraint solver, spaces, structures and infill components are represented by bounding boxes. Five different high level design constraints can be recognized:

1. Building_Element A *adjacent* to Building_Element B
   This relation will be implemented with the Touch-constraint. The axis along which the building elements touch must be specified (X or Z)). At first in the building element creation phase, this value will usually be unknown and so the axis will obtain a default value. The constraint will take care of having the building elements touch at one of the sides with a shared plane area > 0.

2. Building_Element A *in* Building_Element B
   This relation will be implemented using the Contains-constraint. The constraint will take care of one building element being inside another. It does not express at which location.

3. Building_Element A *on* Building_Element
   This relation will be implemented using the Touch-constraint. The axis along which the building elements will touch is always the Y axis.

4. Structure A *aligns* Space_Side B
   This relation will be implemented using the AlignSides-constraint and the LatDistance-constraint. The sides (left, right, front, back, top, bottom) of the space

and the structure must be specified respectively. The constraint supports the positioning and scaling of walls and floors as space boundaries.

Moreover, to support the design process the following constraints will imposed:

- All Infill components will get a FixDimension constraint for all dimensions. We presume that the components dimensions will not change during this part of the design process.
- All spaces get a MinDim and a MaxDim constraint. These constraints are deduced from the Length, Width and Height of a space by calculating 10 % respectively 200% of each.

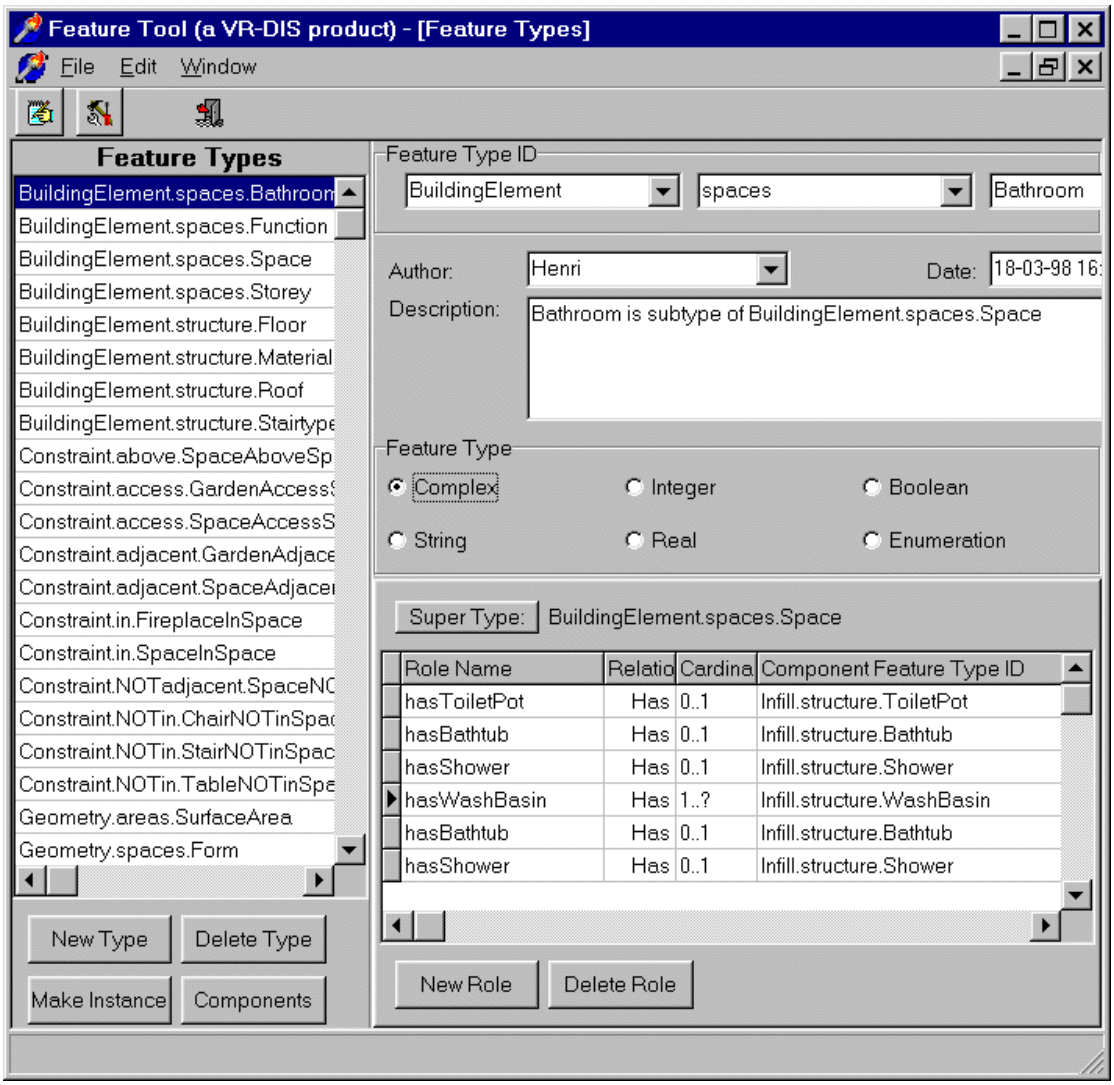
5.4 Presentation and Operation

At he moment the presentation of design information consists of two interfaces. One interface for manipulating the Feature Model of the design and one interface for viewing the shapes of the design (the so called Scale Model) and limited manipulation of the shapes.

5.4.1 Feature Model View

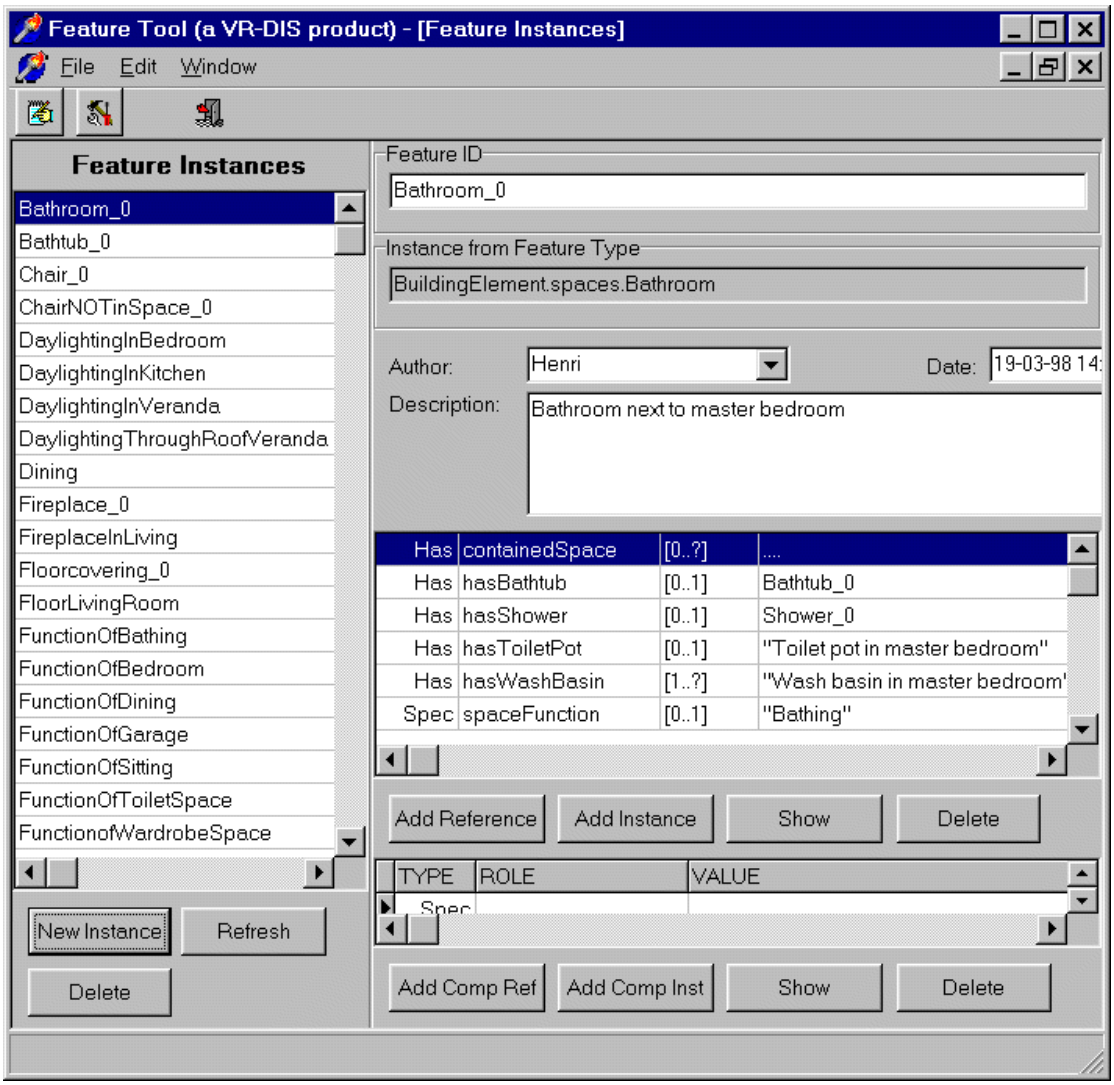
The Feature Model View provides full access to all design information modeled as features. The Feature Model View consists of two main parts, one for editing feature types and one for feature instances.

Figure 2: **Feature View: Types**



The left part of the screen shows a list of already created Feature Types. In the right part a new Feature Type can be created by choosing one of the basic types (complex, string, integer, real, boolean or enumeration) and entering a description. Moreover the relations are specified by entering a role name, a relation type (has, spec or assoc), the carnality and the referenced Feature Component.

Figure 3: **Feature View: Instances**

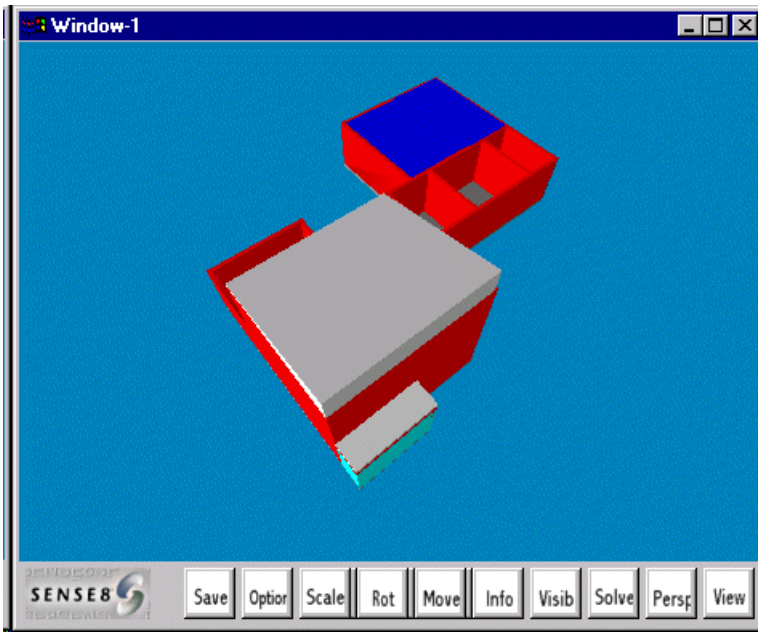


In the left part of the screen a list of already created Feature Instances is shown. A new instance is created by first selecting a Feature Type from a list. In the right part of the screen a name (Feature ID) and description is entered. At the same time the list is shown of all possible relations as defined for that Feature Type. Now these relations can be filled in by adding references to other (already existing) Feature instances.

5.4.2 Scale Model View

In the Scale Model View the following buttons are available: Perspective/isometric view, View direction (Top, Bottom, Front, Back, Left, Right), Solve constraints, Save model, Toggle object visibility, Object information, Move object, Scale object, Options (Toggle fix infill component dimensions, Toggle limit space dimensions)

Figure 4: **Scale Model View**



Spaces are represented by their space boundary, being a box. Structures and infill components are represented by their specific shapes. Spaces, structures and infill components can be moved and scaled. Infill components can also be rotated.

Each design phase is stored separately in a database file.

6. DISCUSSION

6.1 First impression

6.1.1 Feature Model characteristics

Explaining the characteristics of the Feature Model is most easily done by comparing the modeling technique with Product Modeling as defined in the STEP standards. The Feature Model appears more flexible because of the possibility to create new Feature Types during the design session. Using STEP Application Protocols the product model is restricted to a certain application area and fixed for a period of time. Secondly, Features can be shared by other Features. Though the Express language allows for construction of dependencies, in the Feature model sharing attributes is maintained in a more direct manner, closely related to the way a design expresses these kind of conditions. Thirdly, relations between features can be added at instance level. So, the user is not limited to the definitions of the existing feature types but he/she can extend Features with any kind of relationship with other Features. In this way project-specific information can be handled.

6.1.2 Constraint Solver characteristics

The constraint solver acts as a support tool for locating building element geometries at the desired location and keeping topological relationships intact. Lacking facilities like snap in this primitive experimental design system it is the only alternative next to typing

coordinates. The constraint solver appears to be a good help, but more important is that the way of expressing building element positions by constraints is very close to the designer's way of thinking. Constraints in the described case are used for:

- generating a first floor layout

  Generating plans from requirements usually causes offensive reactions, because designers don't like their primary activity being automated. In this experimental design system on the contrary, the generated layout of the floor plan is not more than a starting point and only maintains those spatial relationship which are explicitly defined by the principle or implicitly by the designer. In practice such relationships are rather limited in number and therefore it is rather easy to understand the system's behavior.

- placing building elements

  Support for placing building elements can be found in more advanced CAD packages. Usually the shape, location and size of a geometrical object in such applications are defined by specifying parameters. Parameter specification is executed in a designerly way using techniques like object snap. In our experimental design system though, any object can be restricted in its dimensions and location. The kind of restriction that the designer wishes to impose depends on many factors. In the experimental design system these factors are roughly classified into the four building element categories which proves to be adequate for the envisaged design goal.

- checking if imposed geometric relationships are validated

  Instead of solving a design problem expressed by constraints, constraint validation will do in many occasions. The system should indicate which objects are part of a constraint that is invalid. Then the user can decide on which constraint to relax. This facility is not (yet) present is our design system.

6.2 Shortcomings

Comparing the initial system requirements with the experimental design system implementation one can deduce the following shortcomings:

- There is no support for the use of grids.
- Material properties cannot be added to building objects.
- Building element geometry cannot be transformed otherwise than by means of scaling.

6.3 Concluding remark

Evaluating the experimental design system one should bear in mind that we deliberately focused on the development of the core of a design system that allows us to manipulate all information during the design process. In the near future we will extend the system with 'real' design tools and design knowledge (Mallory-Hill 1998). The Windows

interface of the Feature View will be replaced by a VR Feature manipulation tool (Coomans 1998).

7. REFERENCES

Achten, H.H. and J.P. van Leeuwen (1998) A Feature-Based Description Technique for Design Processes: A Case Study, *Proceedings of the 4th Conference on Design and Decison Support Systems in Architecture and Urban Planning*, Maastricht, The Netherlands, July 26-29, 1998 (forthcoming).

Allen, J.F. (1983) Maintaining knowle4dge about temporal intervals, *Commun. ACM* Vol 26 No 11, 1993, pp 832-843.

Coomans, M.K.D. (1998) A VR User Interface for Design by Features, *Proceedings of the 4th Conference on Design and Decison Support Systems in Archietcture and Urban Planning*, Maastricht, The Netherlands, July 26-29, 1998 (forthcoming).

Gorti, S.R. and R.D. Sriram (1997) From symbol to form: a framework for conceptual design, *Computer-Aided Design* 28, p. 853-870.

Leeuwen, van, J.P., and H. Wagter (1998) A Features Framework for Architectural Information, *Artificial Intelligence in Design '98*, ed. Gero and Sudweeks, Dordrecht, Kluwer, 1998 (forthcoming).

Kelleners, R.H.M.C., Veltkamp, R.C., and E.H. Blake (1997) Constraints on Objects: a Conceptual Model and an Implementation, *Proceedings of the 6-th Eurographics Workshop on Programming Paradigms in Graphics*, Budapest, 1997, pp.67-78.

Mallory-Hill, S.M. (1998) Building a Case-Based Design Assistant for Workplace Environment Design, *Proceedings of the 4th International Conference on Design and Decision Support Systems in Architecture and Urban Planning*, Maastricht, The Netherlands, July 26-29, 1998 (forthcoming).

Vries, B. (1997) VR-DIS Research program, *Internal report Faculty of Architecture and Building*, Eindhoven University of Technology, Eindhoven, 1997.

Vries, B. and H.A. Achten (1998) What offers Virtual Reality to the Designer, Third biennial world conference on Intergrated Design & Process Technology, What offers Virtual Reality to the Designer, Berlin, Germany, July 6-9, 1998 (forthcoming)

Vries, B., and A.J. Jesserun (1998) An experimental design system, *Virtual Environments and Systems, 18th ASME International Computers in Engineering Conference*, Atlanta, Georgia, September 13-16, 1998 (forthcoming).