Published as: van Leeuwen, Jos, Ann Hendricx, and Sverker Fridqvist. 2001. 'Towards Dynamic Information Modelling in Architectural Design.' In: Proceedings of CIB W78 2001, Mpumalanga, South Africa, 30 May – 1 June, 2001.

TOWARDS DYNAMIC INFORMATION MODELLING IN ARCHITECTURAL DESIGN

Jos van Leeuwen[1], Ann Hendricx[2], and Sverker Fridqvist[1]

[1] Eindhoven University of Technology
Faculty of Building and Architecture
Design Systems group
www.ds.arch.tue.nl
j.p.v.leeuwen@tue.nl
s.fridqvist@tue.nl

[2] K.U.Leuven University
Faculty of Applied Sciences
Department of Architecture, Urban Design and Planning
www.asro.kuleuven.ac.be
ann.hendricx@asro.kuleuven.ac.be

ABSTRACT: Product modelling has received a lot of attention in the last decade and is now growing into a successful means to support design and production processes, also in the area of building and construction. Collaboration through data exchange and model integration are coming within reach for all participants in the building process. However, it is difficult to apply the current approaches in product modelling to architectural design.
One of the major issues is the necessity for a design model to be conceptually adaptable as design proceeds and more information is becoming available or design decisions are reversed. It is also recognised that no assumptions should be made about design methods and that design information models must support various methods. Furthermore, concepts such as space and user activity play an important role in early design stages and must be supported by modelling tools. The success of computer support for architectural design therefore depends on how well it supports a dynamic handling of design information, as well as if it can handle information regarding non-technical issues.
The paper describes and discusses three long-term, independent research projects that are being carried out in three European universities. The three projects all address the issues mentioned above. While their initiatives were independent and the developments are not formally related, they show strong similarities in terms of objectives, conceptual approach, and methodology.

KEYWORDS: Design Support Systems, Product Modelling, Schema Evolution, CAAD, Object Orientation

1 RESEARCH IN THREE PARALLEL, INDEPENDENT PROJECTS

Recently, several product modelling approaches have been published that support dynamic schema definition and dynamic classification. This paper is a report of three parallel, independent projects, which all were initiated as reactions on observed shortcomings in both commercially available CAD software, and in attempts to develop building product modelling.

The three projects are:

- The IDEA+ project at K.U.Leuven University in Belgium.
- The BAS●CAAD project at Lund Institute of Technology in Sweden.
- The Feature-Based Modelling (FBM) project at Eindhoven University of Technology in the Netherlands.

These three research projects address the problem how information should be defined and structured for design processes, keeping in mind the specific requirements posed by the way designers wish to handle information.

2 PROBLEM AREA, COMPUTER INTEGRATED DESIGN PROCESS

Many efforts on information modelling relate to the support of production stages of business processes. The architectural discipline has adopted computer support in project planning and production processes. Production here means production of documents, e.g. drawings and bills of requirements, as well as manufacturing of building products. The focus in most forms of computer support for the architectural discipline is on the building as a well-specified object to make. Other views are not supported, such as the user view, i.e. the building as an environment to user activities.

Design requires a very different approach to computer support, since it involves creativity. Computer support must always be aimed at the enhancement of the primary tasks. Therefore, if this primary task is a creative process, computer support should stimulate creativity. At least it should not obstruct creativity. One major issue for support of early design tasks is handling the typical absence of detailed data, especially during evaluation and analysis activities.

Another flaw in current practice is the problem of inconsistent and redundant information. On top of the architect's descriptive sketches, preliminary plans and construction plans, other members of the design team tend to produce at least one extra (digital) model to perform their tasks on. An integrated approach is essential for the success of IT in construction. By this is understood that information concerning a building is continuously managed with computers throughout the life of a building, from its conception to its final demolition. In an integrated design environment, a gradually refined design representation would take a central position.

The IDEA+ project at K.U.Leuven University in Belgium
The IDEA+ project aims at developing an Integrated Design Environment for Architect designers, in which design tools and computational tests are gathered around and make use of one and the same core object model. three different aspects in which the IDEA+ core model differs from many other product modelling research initiatives: the systematic approach in the construction of the model, the respect for the evolutionary nature of architectural design, and the use of actual and complete design cases to test the model (Hendricx 2000).

The BAS●CAAD project at Lund Institute of Technology in Sweden.
The now finished BAS•CAAD research project aimed to develop principles for information systems for design, specifically architectural design at early stages. The project would expand the general concept of building product modelling to not only include buildings and building parts, but also users and user activities, and the relations between buildings and user activities(Ekholm & Fridqvist 1998, Fridqvist 2000). While the BAS•CAAD project is ended, other projects continue the line of research.

The Feature-Based Modelling (FBM) project at Eindhoven University of Technology in the Netherlands.
This project develops a framework for information modelling that provides extensibility of conceptual schemas and flexibility of instance models. Design objects are modelled in entities called Features, which in turn are defined by Feature types. The modelling framework provides classes of Features and corresponding classes of Feature types, in order to give the user access to the content of both the design instance model (containing Feature instances) and the conceptual schema (containing Feature types). The technique used in both schemas and models displays high levels of flexibility to support the dynamic way of dealing with information that characterises early design stages (van Leeuwen 1999).

Although the contexts and specific objectives of these three projects differ, there are a number of common conceptual aspects and beliefs concerning the way that design support should be developed. Probably the major of these aspects is the dynamic handling of design data during the process of design. The remainder of the paper concentrates on the features and approaches in the three projects, highlighting the commonalities in them, and drawing conclusions towards dynamic information modelling for architectural design.

3 DESIGN AS A PROCESS

Design is a complicated process. The objects of Design are not defined and fixed once and for all, but evolve during the design process. The initial partial descriptions of an object to be designed, are incrementally refined until they can be used for production. Thus, to be a support to the designer, information systems need to conform to this changing and open nature of design. Closely connected to this are the notions of design phases and different architectural representations. This results in an additional challenge: to provide the possibility of change. Additionally, design is a learning process, where the designer studies and tries to understand the conditions for the use of the designed artefact. During this process the designer makes assumptions of the nature of the artefact, its parts, and its environment; assumptions that must be possible to alter later.

It is crucial for a design support system to provide a design information model that (a) can be used to accurately represent the state of design during the design process and (b) can contain the semantics of the design paradigm and the meaning of design decisions.

3.1 Support for Abstract Design Concepts

Building design, especially in the early stages, deals not only with building parts such as walls, doors and windows, but also with abstract[1] objects, such as user activities, spaces, and functions. If a computer system is to provide design support for the early stages, it must include also these abstract objects. All three projects acknowledge the need to represent and manipulate abstract objects. This has led to the development of generic basic schemas in all projects, to allow all kinds of objects to be simultaneously modelled. The FBM and BAS●CAAD projects present this generic schema to the user, while the IDEA+ project presents pre-defined objects to the user that represent basic building concepts.

3.2 No Assumptions about the Design Method

In architectural design, geometry is not always the prime entry to start modelling information about the design. This is especially true in early stages of design; the design brief concentrates on activities and spatial functions rather than the physical layout of the design.

[1] The BAS●CAAD project holds the stance that also objects that are often described as abstract are actually concrete, just as building elements are. See (Ekholm & Fridqvist 2000).

A number of possible design entries can be distinguished. A designer can start his design either by:

- Organising physical objects;
- Organising abstract spaces with a possible materialisation afterwards (e.g. using a layout programme, in which user activities are assigned to abstract spaces);
- Planning user activities, connecting them to specific spaces afterwards (e.g. defining user requirements);
- Exploring shape without assigning a semantic meaning to it at the start.

This does not rule out the mutual connection between these types of objects, but boils down to not assigning a certain order of importance to them. These different views on the design process and the corresponding design entries are considered very important to prevent the design environment from hindering the creative design process. Activities in design do not take place in a predictable order, the information dealt with in design activities cannot be foreseen: the content and structure of required information or generated information cannot be presupposed.

All of the projects presented here strive to provide an open design environment that does not force any particular design method upon the user/designer. This is based on the shared notion of design as an intuitive, explorative activity, which will be most efficient if allowed to roam freely. Software constraints should not force the designer's mind away from the path it urges to follow. This goal is attained in each of the projects by letting the user determine the information content of modelling objects, rather than the system designer. This aspect of design support is also encouraged in (Ramscar 1994), (Junge 1995), (Eastman et al. 1993), and (Galle 1994).

The *IDEA+ core object model* provides the possibility to approach the design process from different points of view. Both physical objects and the abstract notions of spaces and user activities are modelled. Moreover, the designer has the possibility to handle mere geometrical forms. Providing different design entries comes down to not assigning a certain order of importance to them: none of the basic elements (physical object - space - user activity - geometrical form) is existence dependent on another one.

The *BAS●CAAD framework* supports many aspects of modelling to co-exist in one model through its very generic modelling object, the *Thing Class*, and the supportive objects *Relation* and *Attribute*. A Thing Class represents a class of (tangible) things; possibly this class has only one member, in which case the Thing Class represents a single thing. Since a Thing Class is defined through reference to other Thing Classes, a (multiple) inheritance hierarchy is established, where generic concepts define specific concepts.

In the *FBM project*, modelling is done with objects called Features, which may represent any concept. The resulting Feature model is both flexible and extensible. Extensibility is achieved by allowing designers to define new types of Features. Flexibility is mainly manifested by the model's ability to contain ad-hoc design concepts that are not first defined as a new type of Feature.

4 OBJECT ORIENTATION

Object Orientation as a software development technique is considered the most appropriate for the kind of development in these projects. More important, however, is the fact that all three projects present design information to the user/designer in an object oriented fashion. The appropriateness of object orientation is also indicated by the fact that since the early 20th century, the building industries in various countries have developed and used building

classification systems to facilitate unambiguous communication among actors in building projects. Thus, the use of types or classes is already a common means to structure information about buildings.

4.1 Model Rigidity versus Versatility: Schema Evolution

Computer based modelling depends on one or another conceptual view of the context to be modelled. The more specific the conceptual view, the more limited the number of different views the modelling application can support. In this context, *schema evolution* is the capability of a design information model to adapt to changing insights during the design process. The conceptual information model, containing the classes of design objects, changes over time as design proceeds. A more rigid conceptual model predefines a larger set of design object classes and allows the designer less influence on these classes, whereas a more versatile conceptual model gives the designer more freedom to define and use the object classes, but offers less support for exchange and collaborative design.

The three research projects all regard it important to provide the user/designer a means to define his own object types. The main reasons are:

- To give the user/designer the freedom to follow his preferred path through design, without being governed by structures built into the software by its developers;
- To allow formation and use of new concepts that were not conceived by the system developers, such as new materials or building methods, or new uses of existing ones.

Among the three projects, two principally different standpoints have been assumed regarding rigidity versus versatility of a design model. The more idealistic approach of the FBM and BAS•CAAD projects favours maximum flexibility and defines only generic classes. Both FBM and BAS•CAAD allow the free creation of new types, that subsequently will be used just as pre-defined types.

In contrast, IDEA+ has chosen to avoid the possible anarchy of this approach. This project pragmatically uses a set of pre-defined building-related classes to avoid the problems that may arise from a completely open approach. However, it does supply the possibility to the end-user to define classes by joining the pre-defined concepts in new ways.

The idealistic stance (FBM and BAS•CAAD)

Concepts and notions in creative design cannot always be anticipated in generic conceptual information models; they are often defined during design, by designers. Co-ordinating different professional views and national practices is a great problem, that makes the definition of semantically rich classes difficult. The FBM and BAS•CAAD projects provide three mechanisms to support unanticipated schemas:

1. Attributes or properties are separated from classes (see also section 4.2). In the BAS•CAAD project, the ThingClass is separated from the Attributes. In the FBM project, making no distinction between classes and attributes, all concepts are modelled separately as Feature types with interrelationships;
2. The designer has full access to the schema classes and is completely free to create new attributes and classes, and does therefore not depend on the classes that a particular design system offers. In the FBM project, the designer is even free to add attributes to an instance, without prior modification of its class;
3. Libraries of classes can be issued by international, national, and professional organisations to provide the rigid foundation necessary for communication. The individual designer can choose to draw from these libraries while modelling a design, or build specialised classes as extension of these libraries.

The pragmatic stance (IDEA+)
IDEA+ strives for an equilibrium between a) a rigid but manageable model in terms of consistency and data exchange and b) a very generic model allowing full flexibility and extensibility but leading to problems with consistency and data exchange. When the user can make up his own object classes without keeping some rules in mind, also the possibilities towards application programmes making use of this model are limited. The IDEA+ authors strongly believe that in case of a strictly generic model, the 'limiting' phase is only postponed and will occur during the development of applications based upon the model. A workable solution includes:

1. Presenting generic class definitions and relations, as well as a pre-defined set of classes representing common concepts within the building design/construction. The user decides for him/herself whether or not he or she will specialise the general design concepts into instances of the pre-defined set of classes;
2. Always presenting neutral 'user-defined' specialisations that make it possible to create instances outside the scope of the presented set, and that nevertheless follow the general syntax rules and object behaviour;
3. Locating different aspects and representations of a design concept as much as possible in different object classes that can be combined in a variety of ways (cf. the FBM and *BAS•CAAD* approach). See for instance the PHYSICAL ELEMENT structure in Figure 1.

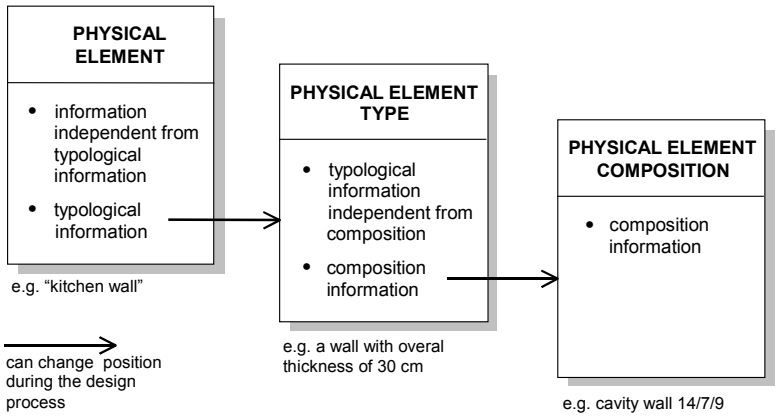

Figure 1. IDEA+: Unravelling the PHYSICAL ELEMENT to enhance its capability to change.

4. Differentiating between 'super user' and 'end user'; whereas the architect designer (called 'end user') will use the presented possibilities, an administrator or software developer (called 'super user' and having knowledge about the underlying core model) can create new class types within the generic structure.

4.2 Modelling Concepts

The FBM and IDEA+ projects both make a difference between the conceptual and the particular through a class/instance mechanism, where instances represent actual, or designed, objects, and classes represent conceptual types of objects.

The BAS●CAAD system differs here by the view that all objects in the computer-based model represent concepts, or classes of things, that may refer to one or several real-world things. Since a class may have only one member, this approach allows representing both generic classes and single things. Thus, it makes the system structurally simpler. However, this introduces the problem of discriminating model objects that represent individual parts of a specific design from generic model objects.

In object oriented programming, class definitions are fixed while instances are dynamically created and deleted. A fully dynamic information system that supports schema evolution therefore needs to represent the classes of such a schema by means of instances. Thus, such a system represents not only individual model objects by instances, but schema classes as well. We call this a *two tier approach* because the model objects are still instances of the schema classes from the perspective of the user of this system.

The FBM framework implements this functionality by defining two sets of object classes: one for the schema classes (called Feature Types); another for the model instances (called Features). Figure 2 shows how the FBM framework implements the two tier approach. The system instantiates objects for Feature Types and for Feature Instances from the respective object classes. For the designer, however, it appears as if the Feature Instances are instantiated from the Feature Types.

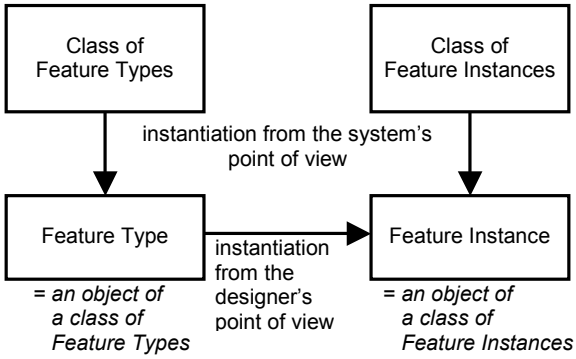

Figure 2. Two kinds of instantiation in FBM: from the system's point of view and from the designer's point of view.

In IDEA+, the conceptual object model uses object classes, while a specific project is a collection of instances of these object classes. Nevertheless, a specific project holds both the generic and the detailed object instances and relationships. As such, IDEA+ also implements a variety of the two-tier system, in that individuals are represented through instances of the generic class *Element* (see Figure 3). These are semantically determined through references to instances of one of the subclasses of *CAAD Entity*. The graphical representation of an Element is separately defined through an instance of *Graphical Entity.* Thus, a symbol in a drawing or other graphical representation of the design can have its meaning changed as the design process develops.

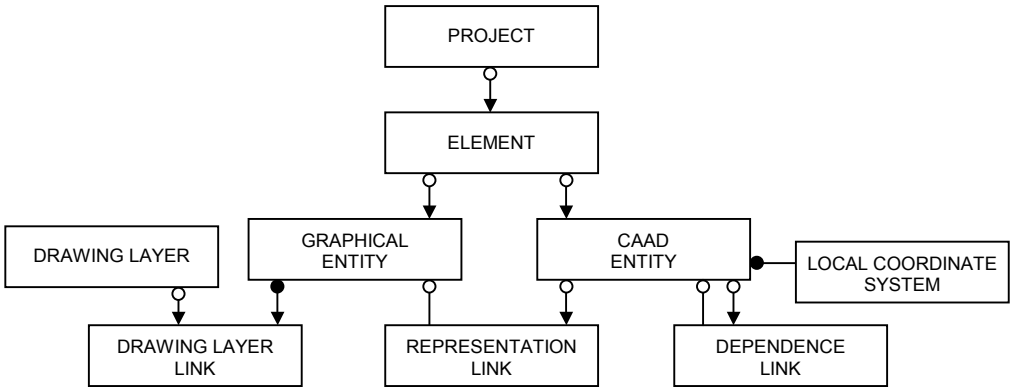

Figure 3. The generic schema of the IDEA+ core object model.

Using separate classes, objects and properties are semantically separated in the model. Separating properties from objects is a common theme in these three projects. Also in the

FBM and BAS•CAAD projects, this is achieved by representing the existence of an individual object by instances of a separate class, and the properties of things by other classes.

The FBM project achieves separation of objects from properties by following a strict referencing approach. All the relationships between Feature Types and those between Features are made by reference. Rather than modelling objects that have properties, this approach models properties that specify objects. Four kinds of relationships can be modelled: specialisations, decompositions, associations, and specifications. Apart from this distinction, relationships can be modelled at the conceptual level (as part of a Feature Type) or at the instance level (added to a particular Feature instance).

Separation of objects from properties is achieved in BAS•CAAD by defining Thing Classes through references to Attributes and to other Thing Classes. A single Thing Class can be determined by others in two ways: 1) by referring to them as more generic super classes; and 2) through Relation objects, which model inter-object relations. While most concepts will be represented by Thing Classes and Relations, the Attribute class is provided for representing simple properties that can be described by a number, a string or any other kind of simple data.

5 BACKGROUND AND CONTEXT OF THE PROJECTS

The developments discussed in this paper should be seen in the contexts of each of the three projects, which are outlined briefly in this section.

5.1 IDEA+: Core Model for an Integrated Environment

The development of a core object model is only one topic of the global IDEA+ research project. As a whole (Hendricx and Neuckermans 2001), the project aims at developing an Integrated Design Environment for Architecture, allowing the modelling and testing of a design with access for all professionals involved. In this environment, the gradually refined design representation takes a central position. At the appropriate time, additional software tools that are in tune with the precision of the design representation at that moment can be plugged in and make use of the design representation. As to the core object model, the conceptual version of it has been developed to a large extent and specific topics have been implemented. A full prototype version, coupled to a daylight visualisation tool (Geebelen and Neuckermans 2000), is due for July 2001. On top of the respect for the evolutionary nature of architectural design, other specific IDEA+ features are:

1. Following a systematic approach: The core object model is developed with the object-oriented analysis method MERODE (Snoeck et al. 1999). Following a model-driven approach, MERODE groups object classes into layers according to their change rate. This is achieved by separating objects of the problem domain (what are the objects you are working with?) from objects that embody functionality (what are the things you want to do with them?). The result is an analysis model that consists of two sub-models (enterprise model and functionality model) and that can be easily embedded in a global environment.
2. Testing with actual design cases: The core object model should be apt to describe architecture in a full-fledged way. For that reason, the model has been put to test with actual design cases, ranging from single-housing projects until extensive urban renewal projects.

5.2 BAS•CAAD: Property Modelling rather than Object-Class Modelling

The BAS•CAAD project was part of the National Swedish “IT Bygg” program, (IT Build), which aims to increase the use of IT in the Swedish construction industry. The BAS•CAAD

project sought to define a general structure for information systems, that would allow information from all phases of construction to co-exist in one model. Specifically, the project aimed towards the early design stages, and to support modelling of both buildings and user activities. The entry point of the research was system theory, which now forms the conceptual foundation for the BAS●CAAD modelling framework that is the result of the project.

The BAS●CAAD Thing Class represents a class of (tangible) things. Multiple inheritance is a fundamental aspect of the BAS●CAAD system since Thing Classes generally are defined by other Thing Classes, thus creating a multiple inheritance structure. The user of the BAS●CAAD system is required to create new Thing Classes to represent the design and its parts, properties, and aspectual views. This process of breaking down the high-level initial concepts into more specific ones is ended by using pre-defined library objects, which will provide the necessary firm basis needed for communication.

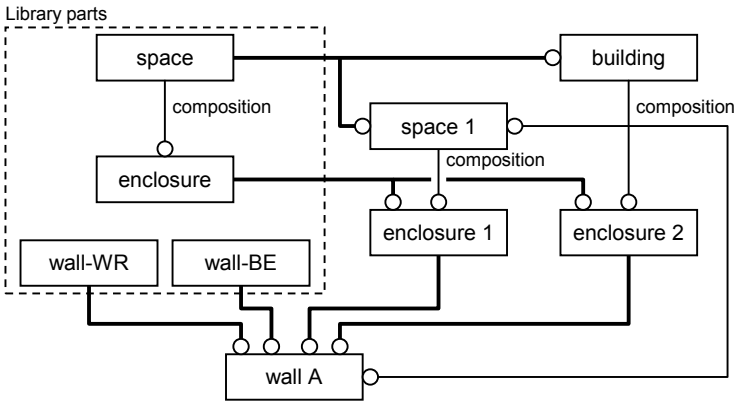

Figure 4. The BAS●CAAD system in use.

Figure 4 illustrates an example of how the BAS●CAAD system would be used. It shows how a building is modelled as being composed of spaces and walls. It also shows how different aspects of one object can be separated. The object "wall A" serves two functions, it both encloses the room "space 1", and the building as a whole through inheriting "enclosure 1" and "enclosure 2", respectively. Further, "wall A" is defined as both a work result and a building entity by inheriting "wall-WR" and "wall-BE".

5.3 VR-DIS: Design in Virtual Reality

The FBM project is part of the so-called VR-DIS research programme at Eindhoven University of Technology. The objective of the VR-DIS programme (for Virtual Reality – Design Information System) is to achieve a design environment that supports collaboration of multiple participants. This involves various modules for support of design tasks and evaluation of the design, for instance by the users of the future building. The programme concentrates on Virtual Reality as the major technique for the interface to the design environment. The programme consists of a number of projects, mostly by PhD students but also including post-doc researchers. These researchers cover a wide range of backgrounds in the AEC discipline, e.g. architectural design, building physics, construction management, urban planning, but also in the area of computer graphics.

While the technique for design information modelling in the VR-DIS programme uses the Feature-Based Modelling approach, a wide range of other aspects are part of the research. Prototypes, currently in various stages of development, testing and evaluation, include:

- Intuitive 3D sketching environment;
- Constraint-based spatial modelling tool;
- Measuring user satisfaction through virtual environments;

- Visualisation and manipulation of dynamic data structures in virtual reality;
- Multi-agent system for network decision analysis;
- Multi-agent system for urban planning support.

More theoretical research is continued in the following areas:

- Feature type recognition and case-based reasoning;
- Management and communication of design knowledge through the Internet;
- Design representations and managing complexity in architectural design;
- Associative reasoning in the early phase of architectural design.

More detailed information on the VR-DIS programme can be found in (Achten et al. 2001) and the Design Systems group website (http://www.ds.arch.tue.nl/research).

6 CONCLUSIONS DRAWN FROM THE INDIVIDUAL RESEARCH PROJECTS

6.1 Conclusions from IDEA+

The development of the IDEA+ conceptual model has been based on the principles of an object-oriented analysis method. This has lead to a systematic and well-documented model. If only because of this, the model does not hamper later extensions or modifications, and can be the keystone for further research. Especially the strong distinction between objects of the problem domain and objects that embody functionality, makes it easy to embed it in a global environment and link it to other software modules.

Concerning the aspects of flexibility and extensibility, IDEA+ strongly differentiates between end user and super user. For the *end user* (e.g. the architect designer), he or she can express his or her individual design creativity by combining the presented objects - be it physical objects, graphical entities or abstract concepts as space and user activity - in a number of ways. Moreover, the designer can include user-defined objects that nevertheless follow the overall structure. The model does not force any level of detail or design method upon the designer. Separating, firstly, the semantic and the graphical features of a design concept and, secondly, a design concept and its detailed features, enables the model to reflect changes that occur in the course of the design process. Considering these aspects, actual and complete design cases have put the IDEA+ core model to the test.

For the *super user* (e.g. an administrator, researcher or software developer), he or she can create new object classes and relationships as long as he or she respects the generic principles of the object model. The super user can fully exploit the advantages of the followed object-oriented approach.

6.2 Conclusions from BAS•CAAD

The BAS•CAAD project displays how modelling based on a generic set of modelling objects allows many different modelling aspects to co-exist in one model. It also shows that the necessary rigid foundation needed for communication can be achieved through commonly approved libraries. Finally, the BAS•CAAD project successfully implemented automatic model object classification. This technique is important for an open system with complete freedom for the end-user to define classes, since it supports referring the user-defined classes to the common libraries. Additionally, this mechanism may serve as the basis for several design support functions of a CAD system.

6.3 Conclusions from FBM

The FBM project has resulted in a detailed design of the two tier approach in the form of a framework that provides the classes of Features and classes of Feature types as shown in

Figure 2. The framework is implemented in an API that forms the basis for the development of various modules of the design system in the VR-DIS programme. Case studies using realistic design cases and evaluation experiments have been and are still conducted to confirm the functionality of both the modelling approach and the VR interface to the design system.

The implementation of the framework was originally based on object oriented database technology, but now uses XML-Schema as its vehicle for data storage (van Leeuwen & Jessurun 2001, see Figure 5). With this new technology, the design system allows the designer to share and use shared design knowledge through Internet. Current projects that develop this aspect of collaborative design concentrate on Feature type recognition and case-based reasoning.

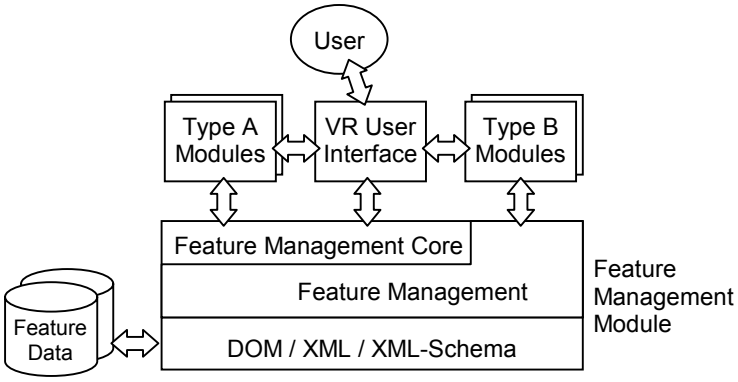

Figure 5. VR-DIS system architecture using XML to implement Feature modelling.

7 COMMON CONCLUSIONS DRAWN TOWARDS THE DEVELOPMENT OF NEXT GENERATION ARCHITECTURAL DESIGN SUPPORT SYSTEMS

The projects discussed demonstrate the feasibility of information modelling approaches that support the dynamic nature of the design process, i.e. to support change in design models; support abstract design concepts; support multiple design approaches.

The gradual process of design calls for a solution that allows gradually adding properties as well as parts to the design model. Moreover, it requires manipulating properties and rearranging properties. In traditional product modelling approaches, an object has a predefined set of properties, laid out in the object's class definition. The *property oriented* approach, as exemplified in the projects discussed, separates the objects from its properties. Networks of objects and properties represent design concepts. The loose structure of these networks provide the flexibility necessary for manipulating and rearranging properties. In this manner, the design representation can continuously reflect the state of the design process.

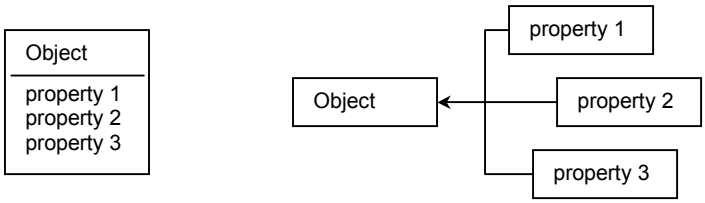

Figure 6. Traditional approach (left) versus Property oriented approach (right)

Schema evolution is the capability of the a design information model to adapt to changes in design concepts. A versatile conceptual design model provides the freedom to add and change concepts and relationships. However, it is acknowledged that for purposes of

exchange and communication, levels of standardised schemas are required. These projects implement different ways to provide for this.

Schema evolution is implemented in the three projects in two distinct manners. The difference between the approaches follows the choice whether schema evolution should be available for the end user or for authorised users only. The more open the system is, the more stringent will be the requirements on the system regarding information management.

Offering the possibility of schema evolution to the designer, will have either a consequence for the designer or for the system. Either the formalisation of design concepts is now a new task for the designer, or the system must be able to detect and automatically formalise new design concepts.

REFERENCES

B. de Vries et al., (2001). "The VR-DIS Research Programme, Design Systems group". In: *Proceedings of the Computer Aided Architectural Design Futures Conference 2001*, 8-11 July 2001, Eindhoven University of Technology, The Netherlands.

Ekholm A. and Fridqvist, S. (1998). "A dynamic information system for design applied to the construction context". In: *Proceedings of the CIB W78 workshop The life-cycle of Construction IT*. June 3-5, 1998 in Stockholm, Sweden.

Ekholm, A. and Fridqvist, S. (2000). "A concept of space for building classification, product modelling, and design." *Automation in Construction* 3 (2000) pp. 315-328.

Eastman, C. M., Chase, S. C., Assal, H. H. (1993). "System architecture for computer integration of design and construction knowledge." *Automation in Construction* 2 (1993), pp 95-107

Fridqvist, S. (2000). *Property-Oriented Information Systems for Design, prototypes for the BAS·CAAD System*, PhD thesis, Lund University, Sweden.

Galle, P. (1994). "Specifying objects as functions of attributes: Towards a data model for design". In: Lasker G E (Ed.), *Advances in Database and Expert Systems*, pp 69-73. Windsor, Canada: The International Institute for Advanced Studies in Systems Research and Cybernetics.

Geebelen, B., Neuckermans, H. (2000). "IDEA-l, an Early-Stage Architectural Design Tool for Natural Lighting." In: *International Building Physics Conference, Tools for design and engineering of buildings*, Eindhoven, Sep. 18-21, 2000, pp.275-282.

Hendricx, A. (2000). *A Core Object Model for Architectural Design*, PhD thesis, K.U.Leuven University, Belgium.

Hendricx, A., Neuckermans, H. (2001). "A model driven approach to the development of an architectural object model." Accepted for publication in the *International Journal of Artificial Intelligence in Engineering*, Special issue on Product Modelling.

Junge, R. (1995). "Aspects of new CAAD environments." *CIB proceedings, publication 180, CIB workshop on computers and information in construction*, Stanford 1995.

van Leeuwen, J.P. (1999). *Modelling Architectural Design Information by Features, an approach to dynamic product modelling for application in architectural design*, PhD thesis, Eindhoven University of Technology, The Netherlands.

van Leeuwen, J.P. and A.J. Jessurun (2001). "XML for flexibility and extensibility of design information models". In: *Proceedings of CAADRIA 2001*, Sydney, April 19-21, 2001.

Ramscar, M. (1994). "Static models and dynamic designs - an empirical impasse vs. an inductive solution." *Proceedings of 1st European conference on product and process modelling 1994*, Dresden.

Snoeck, M., Dedene, G., Verhelst, M., Depuydt, A. (1999). *Object-oriented Enterprise Modelling with MERODE*. K.U. Leuven University Press.