A FEATURES FRAMEWORK FOR ARCHITECTURAL INFORMATION

Dynamic models for design

JOS VAN LEEUWEN
Eindhoven University of Technology, Building Information Technology,
PO Box 513, unit 20, NL-5600 MB, Eindhoven, The Netherlands.
email: j.p.v.leeuwen@bwk.tue.nl
http://www.calibre.bwk.tue.nl

AND

HARRY WAGTER
P2—Managers,
PO Box 70, NL-5240 AB, Rosmalen, The Netherlands.
email: h.wagter@p2managers.nl

Abstract. Supporting architectural design by means of computers requires the development of formal representations of design information. The paper reviews the way information is dealt with in design and describes this as the dynamic nature of design. This leads to the notion that an information model should evolve along with the development of a design. Keywords in this evolution are extensibility and flexibility of information models. The approach developed to fulfil these requirements is based on the techniques of Feature-Based Modelling of which the concepts and potentials are discussed with respect to their relevance for the area of architectural design. Subsequently, a framework is presented for the development of architectural Feature-based modelling systems. This framework includes a layered schema for information definition that provides basic classes of Feature types and the conditions for extension and flexible manipulation of both conceptual and actual information models. Relationships between the entities in these models are discussed in more detail and an approach is introduced for supporting the modelling of relationships that are not defined at the conceptual, or typological, level.

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

1. Information in Design.

Design, as a problem-solving process, involves activities of searching information, analysing, manipulating, and structuring information, generating new information, and evaluating and communicating information. These are not sequential activities, but take place in cycles (Markus, 1969; Maver, 1970). Lawson (1990) argues that designers tend to switch in an ad hoc manner between different activities, resulting in concurrency of activities with no predictable sequence. This *dynamic nature of design* is subject of the research presented in this paper.

Information is generated during design, concerning the definition and specification of selected or generated solutions. Design also involves combining information, finding relations between data and developing or discovering new structures in concepts and ideas that lead to design-solutions. Information is clearly not treated as static data, its content and structure is invariably subject to change. Assuming that designing and design-modelling are two tasks that are best performed concurrently, this means that an information model for support of design-tasks requires a flexibility that allows for (re-)definition and (re-)structuring of information: it requires evolution of the model during design.

1.1. EVOLUTION IN PRODUCT MODELS.

The notion of evolution and adaptation of models of design information is also a basis for the development of the Engineering Data Model by Eastman et al. (1991; 1995; 1997). In the EDM research it is argued that the schema for CAD data cannot be known beforehand, but is “defined incrementally as design proceeds” (Eastman 1991). The structure of design data describing components in design depends upon decisions regarding the technology and functions associated with the design components.

Ramscar (1994) believes that product modelling approaches will not achieve a complete model that is necessary for an integration or exchange standard, because they result in models with a fixed view, which are unable to deal with unforeseen design data and can not respond to the changing needs of their users.

The Building and Construction Core Model in ISO 10303, or STEP (ISO 1996), does not form a very fixed view in itself since it contains a rather loose collection of information entities. However, the purpose of this core model is to serve as a basis for Application Protocols (APs), dedicated to a particular range of applications. These APs will consist of rigid models, not being able to deal with unforeseen data-structures.

Paper at AID ’98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design ’98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

1.2. ILL-DEFINED PROBLEMS IN DESIGN.

As a problem solving process, design involves the structuring of information that often concerns ill-defined problems. Ill-defined here means that the problem is not known in full detail from the start of the design task, and consequently that its structure cannot be presumed. This is especially the case when many parts of the problem are interrelated while the relationships are not obvious. Design then involves recognising the structure of the design problem (Chermayeff and Alexander, 1963) or converting an ill-structured problem to well-structured sub-problems or to a more constraint problem (Simon, 1973).

Coyne et al. (1991) conclude that the formulation of the design problem is itself dynamic, its representation being revised continually in accordance with the changing situation. “The formulation of the design problem can be regarded as a problem-solving task in itself, one that is undergoing reformulation as the process continues.” (Coyne et al. 1991)

1.3. STYLISTIC EVOLUTION OF DESIGNERS.

Another aspect of the dynamic nature of design is that designers learn. They change their approach to solving design-problems, finding new techniques, new rules, new concepts. Mitchell (1990) refers to this as stylistic evolution, and stresses that CAD systems must provide for this essential component of creative design. A design system, according to Mitchell, is an open, flexible, constantly evolving knowledge-capture device rather than static collections of familiar tools and dispensers of established wisdom.

Conceptual information models that underlie design support systems must accommodate this stylistic evolution by adaptation of the conceptual model to the changing demands. A similar evolution can evidently be noticed in the Building & Construction industry as a whole, resulting in the constant development of new techniques, methods, products, and materials. These new agents in the field also require adaptation of information models for design support.

1.4. CREATIVITY REQUIREMENTS FOR INFORMATION MODELS.

The following conclusions can be drawn from the above considerations of the dynamic nature of design:

- Design is a process of problem-solving, often concerning problems that are initially not well-structured;
- Activities in design do not take place in a predictable order, the content and structure of information dealt with in design activities cannot be foreseen;

Paper at AID ’98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design ’98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

- Information related to design problems and solutions is dealt with in different ways, related to the approach of solving the design problem. Design involves creativity through combination of these approaches:
  - Selection of an existing solution involves matching information related to the problem and the existing solutions;
  - Creating a new solution involves generating new information;
  - Combining existing information in order to find new relations or structures in concepts that lead to design solutions;
  - Altering the design-problem in order to find a suitable solution.
- Individual designers, as well as the sector of the Building & Construction industry as a whole, are under constant development, with new knowledge, concepts, techniques, methods, products, materials, and styles emerging. Information models must evolve along with this development, in order to accurately represent the changing domain of design and B&C.

The above conclusions lead to the statement of new requirements on information models that are to support the dynamic nature of design. These requirements, discussed in detail in the next section, are denoted by the term extensibility and flexibility, concerning the possibility for an information model to evolve along with the development of a design.

2. Models of Dynamic Information.

For information to be represented in an information model it is necessary that the definition of information be formally specified in a conceptual information model, detailing the type of the content of information and the structure of the content. For instance, the formal definition of the information concerning chairs would specify the type of properties that characterise chairs: number of legs, colour, arm-rests, headrest, etc.

2.1. EXTENSIBILITY.

In creative design, it is very often the case that typological information is not known in advance, that is, before information needs to be modelled for a particular case. In other words, concepts and notions in creative design cannot always be anticipated in generic conceptual information models; they are often defined during design, by designers. Continuing the example of chairs, the design of a chair with a tip-up seat may result in the definition of a sub-type of chairs with additional properties.

Some situations are listed below in which the content and structure of information concerning a particular concept cannot be known prior to the moment this concept comes into actual usage.

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

- Information representing a specific design-style is defined by a designer and is in a constant state of change and extension during the learning process and career of the designer;
- The definition of style-rules or conventions on, for instance, dimensions and methods of construction for a particular building project can only be formally determined during the appropriate stages of the design and construction process of this particular building;
- The development of new techniques and methods of construction requires that representations for these new concepts be added to conceptual information models;
- The development of new products or materials to be used in construction may require formal definition of new kinds of characteristics that will be used to describe them;
- A notable example of the above situation is the development of industrial construction systems. The production of the individual components of such a prefabricated system requires a dedicated conceptual information model. Subsequently, the application of such a system in actual building design and construction requires information to be available that is strongly specific, in content and structure, for the particular building system. This includes information on the characteristics of the components and the system as a whole, the requirements posed on the design and construction with the system, and, for instance, assembly-procedures.

In most of these situations, concepts remain subject to change after they have been formally defined. For conceptual information models to evolve with these changing concepts, flexibility of the model is required, which is discussed in detail in the next section.

Another issue that pleads for the extensibility of conceptual information models is that the responsibility of the definition of information concerning a particular concept should rest with the designer of the concept, not with, e.g., engineers or vendors of modelling software.

2.2. FLEXIBILITY.

One of the reasons why a conceptual information model should be flexible follows logically from the above discussed requirement of extensibility. Extension of a conceptual model requires flexibility: newly defined entities of information need to be embedded in the conceptual model. They will define relationships to existing entities, and reversely, existing entities will need to relate to new entities. For example the definition of a new component in an industrial building system inevitably needs to have relationships to the components already existing in this system, while the existing components probably will need to gain ‘awareness’ of the new component. This requires an

Paper at AID ’98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design ’98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

adequate level of flexibility in the definition of information entities, allowing properties or attributes defining relationships to be modified or added.

Obviously, flexibility is also required when a conceptual information model is modified in other manners than by extension with new entities of information. This kind of restructuring the conceptual model may follow a change of insight in the role and function of certain parts of the model: a change in the meaning of certain concepts. The stylistic evolution of designers, argued by Mitchell (1990) to be an essential part of creative design, is an example of when such a modification of concepts may occur.

Another situation involving restructuring an information model, does not necessarily take place on a conceptual level, i.e., on the level of typological definitions. The type of flexibility that is required by this kind of modification of a model's structure is related to the flexibility in the way designers use their concepts. The relationships between concepts in a design are rarely constant during the course of design; they are very likely to change from moment to moment, as a designer changes point of view, has a new inspiration, or tries to find other possibilities for solutions. This dynamic way of dealing with concepts and notions in design is regarded a crucial aspect of creativity in design and has been subject of research by for instance Coyne et al. (1991).

Adaptation of the conceptual model does not result in the desired behaviour of the models, since a similar concept may be used in different ways, even within a single model. This means that the desired dynamic behaviour should lie within the formal definition of the concept itself. The function of a concept may have to be defined as a dynamic one, as well as its relationships to other parts of the design.

A simple, though not trivial example will illustrate this in a design-situation. Spaces and boundaries of spaces are concepts that play an important role, especially in the earlier stages of design. The definition of a space boundary in an early stage is rarely fully known in the detail that it will have at more final stages. There is an evolution of the notion a particular designer has about the space boundary during the course of design. Some possible states of this notion of space boundary by the designer are shown schematically in figure 1.

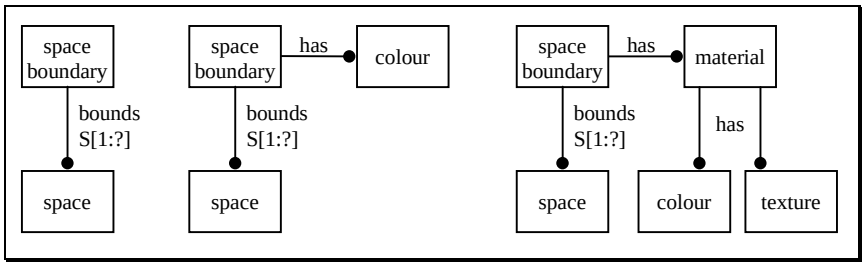


Figure 1 A space boundary in evolution during design. The definition of the entity-type space-boundary changes while design progresses. (Express-G notation)

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

2.3. IMPLICATIONS ON THE DEVELOPMENT OF DESIGN SYSTEMS.

The above considerations on how to support creativity in design by means of dynamic information models have the following immediate implications on the design and development of design systems.

Concerning extensibility, a design system should allow designers to define and add the types of entities that will be used during the creation of a model. This ensures that the design model accurately represents the rationale of the design with entities that reflect the concepts and terminology used in taking design decisions. The newly defined entity types need to be embedded in the structure of existing entity types, which may require adaptation of existing types. This kind of modification of conceptual models forces strong requirements on the management of integrity of design models, an issue that has been addressed in Eastman (1996) and is also subject of developments in Bailey (ed.) (1997).

Concerning flexibility, design systems should support the flexibility that is required for the above mentioned extensibility and modification of conceptual models. Beyond this, design systems are required to allow ad hoc structures of information that are not typologically defined. This implies that relationships between entities of information can be modelled that are not laid out in the conceptual model, but that appear only in instantiated models.

3. Feature-based Modelling – concepts.

The approach chosen in the research presented here, Feature-Based Modelling (FBM), is a technique of modelling product information that originates mainly from areas of Mechanical Engineering. The background and history of these techniques have been discussed and summarised in early papers by Cunningham (1988) and Shah (1991a), and more recently by Shah (1994) and Bronsvoort (1993; 1996). In the light of this research, FBM has been reviewed for its relevance to architectural design in Van Leeuwen et al. (1996; 1997). The main conclusions from the latter reviews are that, although there is a strong focus on Form Features in the original area of application of FBM (see figure 2) and recently also in architectural design research (Gero and Park, 1997), the concepts of this modelling approach are very relevant for modelling architectural design information in a broader sense.

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

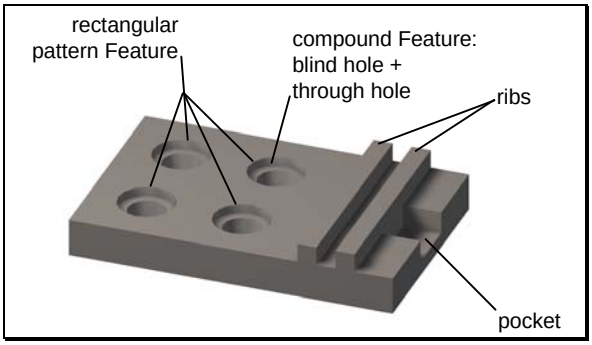


Figure 2 Examples of Form Features typically found in areas of mechanical engineering. (Van Leeuwen and Wagter, 1997).

Features in mechanical engineering describe characteristics of a product's part, which are relevant in reasoning about the part and in the process of manufacturing the part. Typical classifications of Features include categories of, e.g., Form Features, Precision Features, Material Features, Assembly Features, and Constraint Features. Efforts on standardisation in Feature modelling, e.g. ISO 10303 Part 224, have not yet resulted in international standards, but it is generally accepted that extensibility of collections of Feature definitions is an essential requirement (Van Emmerik 1990; Shah 1991b).

Two major approaches to generating Feature models are being followed. *Feature recognition*, historically the first approach, derives data from geometric models and builds up the Feature model from an analysis and interpretation of this data. This approach helps structuring an otherwise unstructured geometry into a data model that is useful for, e.g., manufacturing process plans. The modelling of the geometry itself is not addressed in this approach.

Design-by-Features, however, does assist in generating geometric models, since the model is built up using Feature as primitives, rather than using geometric primitives. The main advantage of the latter approach is that high-level information can be modelled from the beginning and does not need to result from analysis of semantically poor geometry. Combinations of the two approaches have been researched by De Martino et al. (1994).

3.1. CONCEPTS OF FBM RELEVANT FOR ARCHITECTURAL DESIGN.

In the context of this research, two major aspects of the Features technology are of interest for modelling architectural design information. The first concerns the formal definition of design information into Feature types, of which instances are created during the modelling of a particular design. The set of possible Feature types is not a limited set. Definitions of new Feature types can be created by designers when the need for a new type arises, for instance following

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

the development of a new concept, production-technique, or material. Conceptual models, formed by collections of Feature types, can therefore be extended with new definitions.

The second major aspect of interest concerns the way information is structured in the model. The Feature model that is built up during design forms a composition of Features and relationships between Features. The conceptual model provides a flexibility that allows the modelling of relationships that appear relevant only at times of designing. This flexibility is an inherent part of the definition of Features and results in an information structure that is not predefined. Both aspects are discussed in more detail in Van Leeuwen, Wagter, and Oxman (1996).

3.2. ISSUES BEYOND THE ANALOGY WITH CURRENT FBM.

The analogy between the domains of information modelling, mechanical engineering on one side and architecture on the other, does not hold when looking at the following issues.

In mechanical engineering a strictly hierarchical sub-division is used for the modelling of a product. Feature modelling is applied at the level of modelling the lowest level, the parts. Relations between parts are now subject of research as well (Van Holland and Bronsvoort, 1996), but also here a strict hierarchy is maintained. Architectural design does not generally follow such a disciplined, hierarchic structure for defining the resulting product. Architectural design information involves multiple levels of abstraction, some information concerning detailed elements of the building, other information concerning the building as a whole. This division into levels of abstraction is not standardised, nor is it strictly dealt with. Relationships between entities of information often cross from one level to another and, as the design proceeds, information may be shifted from more abstract levels to more detailed ones.

Our conclusion is that, if we want to benefit from the advantages of FBM also in earlier stages of architectural design, this requires that Features be applied at multiple levels of abstraction rather than just the most detailed level. A consequence is that the Features can be used to model abstract concepts as well as the more concrete concepts. The result of this approach is that the architectural design information model is built up entirely from Features which are used for the representation of any type of concept, any set of information that is regarded to be an entity in the design rationale.

4. A Dynamic Framework for Architectural Features.

On the basis of the original techniques of Feature-Based Modelling and with the considerations of additional requirements for architectural modelling purposes,

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

a framework has been defined for the development of information modelling systems for support of architectural design. This Feature modelling framework uses the following definition of the term Feature:

> *A Feature is a collection of high-level information, possibly emerging during design, defining a set of characteristics or concepts with a semantic meaning to a particular view in the life-cycle of a building.*

This definition reflects four important aspects of Feature modelling in the architectural context.

A. *High-level information with semantic meaning*.
   The semantic meaning of Features is directly derived from the domain for which Features are defined.
B. *Characteristics and concepts*.
   Features do not necessarily represent physical parts of a building, but may be used for the formal definition of any kind of concept that is to be regarded a unit in the design and modelling process, tangible or non-tangible. Examples are spaces and space boundaries, functions, routings, but also relationships such as dependencies or constraints.
C. *Emerging during design*.
   Definitions of Features are not always pre-defined. Features may come into existence during the process of design, when the need for a formal definition of a concept arises.
D. *Particular view in the life-cycle of a building*.
   Features are view-dependent, meaning that their existence is defined from within a particular context. However, the usage of the concept ‘Feature’ is open to all life-cycle stages of the B&C industry.

4.1. DEFINITION OF FEATURE TYPES.

The framework is based on a three-layered schema of information definition. The lowest layer in figure 3 contains actual building information in the form of Feature models. Entities in these models are Features, instances of the Feature types defined in the middle layer. Feature types contain domain knowledge which is either generic knowledge, to be defined by standardisation organisations, or specific knowledge which can be defined by the user of the design system. The format for the Feature type definitions is determined by the upper layer, the Meta layer, which describes what classes of Feature types may be defined in the system. The knowledge incorporated in Feature types includes static knowledge in the form of variables and semantics of their values, as well as dynamic knowledge. Dynamic knowledge involves behaviour of entities in the model, for instance as a reaction to modifications in the model. Both static

Paper at AID ’98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design ’98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

and dynamic knowledge need to be included in the definition of Feature types, and are therefore formally described in the various classes of Feature types in the Meta layer. The basics for these classes are discussed in section 4.3 of this paper.

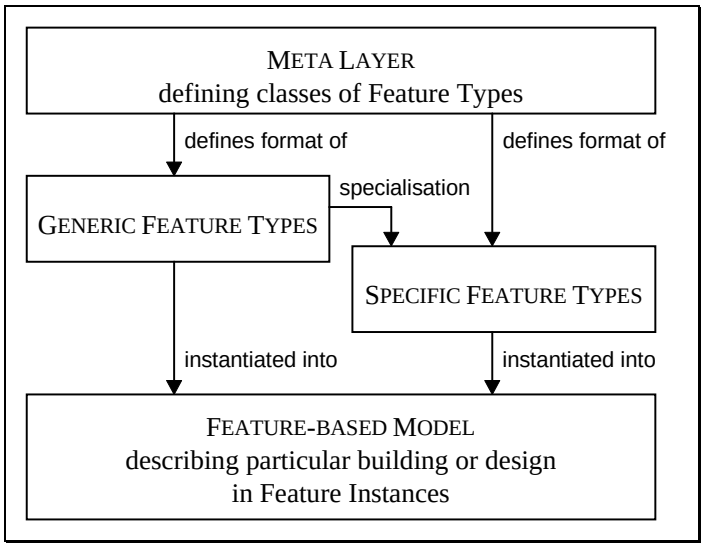


Figure 3 Infrastructure of Feature-based modelling with three layers of abstraction. (Van Leeuwen and Wagter, 1997).

Anticipating this, an example would be a Feature type, called UnitPrice, defined to represent information concerning costs. This type would be of the class Simple FeatureType, having a basetype 'real', and a unit 'Escudo per unit'. Instances of this type store the unit-price of other entities (i.e. Features) in the model to which these instance are attached.

4.2. FEATURE MODELLING ACTIVITIES.

Figure 4 shows the activities of Feature modelling in relation to the parts of the infrastructure displayed in figure 3. The activity of defining a Feature type involves formalisation of domain knowledge. This formalisation is controlled by the Meta layer providing the possible classes of Feature types that may be defined in the system.

The resulting new Feature type is then stored in a library of Feature types. Feature type libraries can be seen as the formal representation of domain knowledge, in which a classification, meaningful to the particular domain, is used. Classification of Feature types are discussed in more detail in Van Leeuwen, Wagter, and Oxman (1996). Specialisation of Feature types involves

the definition of a Feature sub-type based on a super-type selected from an existing library of Feature types.

Using the formalised domain knowledge available in Feature type libraries, the knowledge concerning a particular design case can now be modelled by the creation of instance of Feature types.

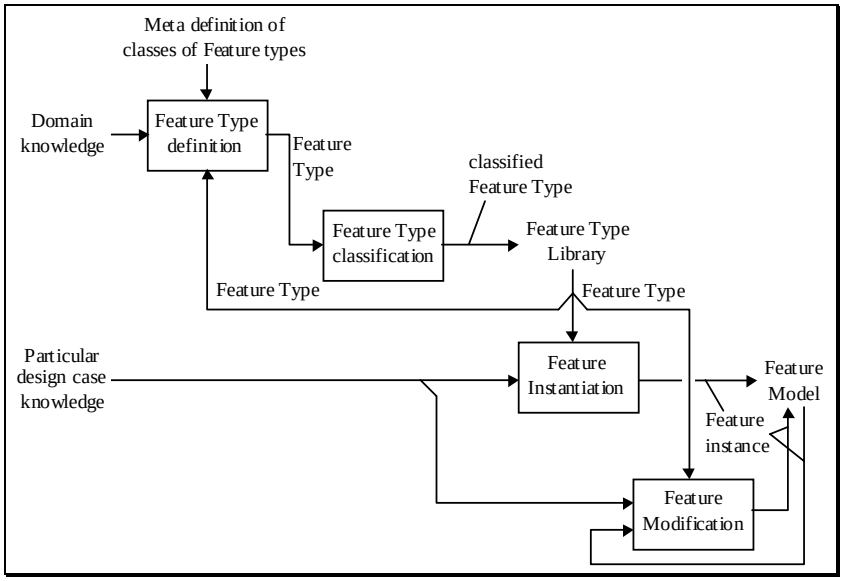


Figure 4 Activities of Feature modelling. Formalisation of domain knowledge into Feature types, which are classified into libraries of Feature types. Instantiation of Feature types into Feature instances comprising Feature models.

4.3. BASIC CLASSES OF FEATURE TYPES.

Eight basic classes of Feature types are defined in the framework of FBM for architecture (see figure 5). These classes are used for types of information that have equivalents in most programming languages and for instance the data definition language EXPRESS (ISO 1994).

A FeatureType is the abstract base-class for all FeatureType classes. It defines that all Feature types have a name and description and identify their author and date of definition. Feature types also may include so-called event-handlers which specify the behaviour of this type of Features as a reaction to a particular event in the environment of the Feature. Event-handlers may for instance be used for requesting special user-input for the contents of a Feature on the event of Feature instantiation. Other events, such as deleting a Feature, may invoke reactions of related Features.

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

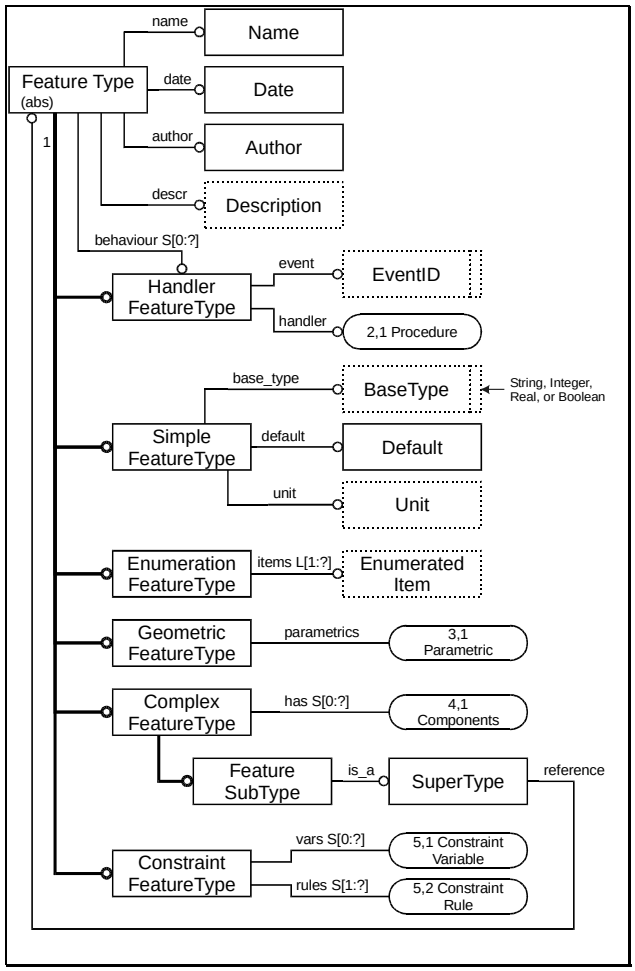


Figure 5 Basic classes of Feature types defined in the Meta layer of the FBM infrastructure. (Express-G notation)

A Simple FeatureType is the basic type for simple data such as strings, integers, reals, and booleans. These types may define a default and a unit for their value.

An Enumeration FeatureType allows the definer of the type to enumerate names, which can be used in modelling to select a particular semantic meaning. An example is a Enumeration FeatureType called ‘Wall-type’ which enumerates the following types of walls: [single wall, cavity wall, composed wall].

A Geometric FeatureType defines geometry in the same way as Form Features do in the mechanical engineering area of FBM. This is parametric geometry, in which the parameters can have relationships to other kinds of Feature types, such as a Simple FeatureType defining the height of a building element.

Paper at AID ’98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design ’98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

A Complex FeatureType forms the basis for compositions of Feature types. The composition of a complex Feature type indicates which other types are included in this type, what role they have in this type, and what is their cardinality. Any Feature type can be part of this decomposition, although, for instance, avoiding circular definitions must be one of the tasks of a FBM system.

A Feature SubType is a Complex FeatureType that inherits properties from another Feature type. In addition, a Feature SubType specifies properties that specialise this type from its super-type.

A Constraint FeatureType is can be used to model constraints, which are a special kinds of relationships between Features that can be automatically maintained and evaluated by a so-called Constraint Solver. Related research is done by Kelleners (1997) (see also section 5 of this paper).

Finally, a Handler FeatureType introduces event handling in the FBM system. Event handling allows designers to incorporate dynamic knowledge in a Feature model. A handler is a procedure, formally defined using a procedural language, that is evaluated on the occasion of a specified event.

4.4. RELATIONSHIPS IN FEATURE MODELS.

Of the basic classes of Feature types, as shown in figure 5, the complex Feature types and Feature sub-types are the types that form relationships with other types. A survey of relationships between information entities in architectural product models has resulted in the distinction of the following kinds of relationships.

- discriminating relationship
- containment relationship
- structural dependencies
- adjacency
- tolerance
- dimensional relationship
- positional relationship
- existential dependency
- algorithmic relationship

These relationships are categorised into four types, three of which are known from Object Oriented approaches.

- Specialisation
  Specialisation indicates that a Feature type is a sub-type of another type. The sub-type inherits all characteristics of the super-type and distinguishes itself from the super-type by adding characteristics: the result is a specialised type.

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

Often this kind of relationship is denoted as an **is_a** relationship. Discriminating relationships are of this category.

- Decomposition
  Decomposition indicates another kind of hierarchy, in which a Feature type is divided into parts, or components. There is no inheritance, the 'component-types' just contain a part of the total of characteristics. Decompositions are often called **has_a** relationships. Containment relationships are of this category.
- Association
  Associations are not hierarchic relationship, there is no inheritance, nor division of characteristics. This relationship indicates any association of two Feature types that does not fall in either categories of specialisation and decomposition. However, a particular type of association is distinguished separately and called specification.
- Specification
  The above distinguished types of relationships appear in OO approaches to information analysis. A fourth type is distinguished in the FBM framework. A specification relationship is a kind of association of two Feature types indicating that one type specifies information about the other type. This is not to be confused with decomposition, since the specifying Feature type does not need to be a part of the specified Feature type. An example of a specification is a type called 'Door' for which the manufacturer-details are specified by a type called 'Manufacturer'. Note that the specification relationship is directed from the 'Door' to the 'Manufacturer'.

Relationships in the FBM approach can be distinguished on two levels. On the typological level, relationships are defined within Feature types. A relationship may be optional, but in principle a relationship at typological level implies that all instances of the particular type will instantiate the relationship. In object oriented approaches, this level of relationship is normally defined using the first three categories shown above. In the FBM framework, Specialisations are defined using the class of Feature sub-types, while decomposition and association are defined in the composition of complex Feature types.

If a relationship is desired between Feature instances that is not defined at the typological level, a problem arises because the relationship cannot be instantiated. The following example will be used to discuss the options for solving this problem: A designer wants to model the relationship between doors and a escape routes through the building. This implies that certain doors require adequate fire resistance. In the conceptual model used by this particular designer, it happens that fire resistance is not defined as part of the Feature type 'Door'.

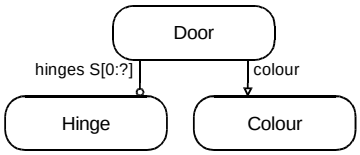
Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

Three different actions that can be taken to solve this problem are discussed below:

1. modifying the type definition
2. creating a new type or sub-type
3. modelling the relationship for the particular instance only

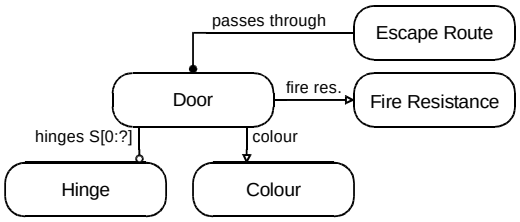
All three of these options need to be provided for by design information systems that are to fulfil the requirements of flexibility as described in section 2.2.

Table 1 Three options for adding relationships to a Feature instance.

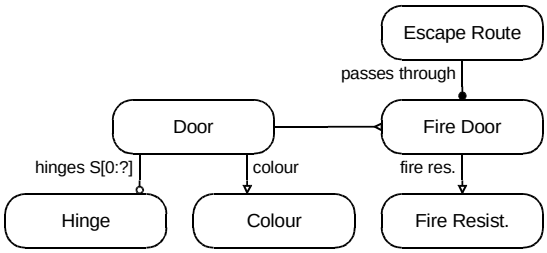


The definition of the Feature type 'Door'. Feature types are shown in white rounded boxes. Relationships are shown with the following symbols:

- inheritance
- decomposition
- association
- specification

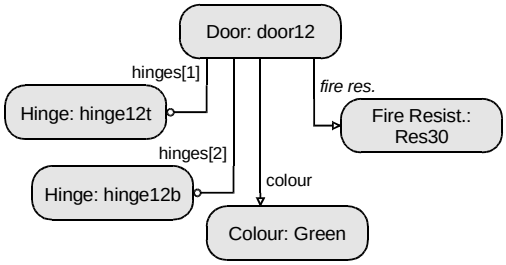


The first option is to change the definition at the typological level, adding the desired relationship to the Feature type. This solution is acceptable if all instances of the particular type require this relationship. Making the relationship optional increases the chances that this is a satisfactory situation, however, in many cases the desired relationship bears no relevance to other instances of the same type: for many doors fire resistance may not be significant. Adding to the definition of a Feature type, its size and complexity continues to increase and eventually it will loose its relevance, which is being the common denominator of its instances.



A second option is to define a new Feature type, either as an amended copy of the original Feature type, or as a Feature sub-type that inherits the super-type's characteristics while adding the desired relationship: a 'Fire Door'-type is defined. This approach avoids the problem of growing Feature types. However, the definition of a new Feature type for each divergent instance may result in an unmanageable number of Feature types (security doors, automatic doors, etc). Furthermore, an instance can only be derived from one type: these door-types are exclusive so that fire doors cannot be attached to the security system. Apart from this, it may not be the designer's notion of the situation that 'adding a relationship to a particular entity' is seen as creating a new concept.

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.



The actual situation in the third option may well be described as a designer adding a non-typical relationship to a Feature instance (the fire resistance specifies only *this* door), without regarding this as a change in concept of the original Feature type. This situation should be modelled exactly in this manner, for the information model to represent most accurately the rationale of this design decision.

What is required thus is a kind of relationship that can be modelled between Feature instances without the necessity of including the relationship in the definition of the Feature type. For this purpose we distinguish a *type-relationship* from an *instance-relationship*. Type-relationships are defined at typological level and have possible relevance to *all* instances of a type, whereas instance-relationships are added at instance level without being defined at typological level, thus without relevance to other instances of the same type.

In the schema, Feature instances are shown in greyed, rounded boxes. Instantiated relationships defined at typological level are shown in normal font, while instance level relationships are shown in italics.

Three of the four categories of relationships described earlier in this section, are relevant also for the kind of relationships that can be expected at instance level. The following matrix lists the seven types of relationships that are part of the FBM infrastructure:

Table 2 Relationships in the FBM infrastructure.

	type-relationship	*instance-relationship*
specialisation	**is_a**	**-**
decomposition	**has_a**	**inst_has_a**
association	**association**	**inst_association**
specification	**specification**	**inst_specification**

The three types of instance-relationships are semantically equivalent to their counterparts at the typological level.

One of the most important implications of instance-relationships for information modelling systems concerns the way information is searched for and addressed in the information model. Instead of using knowledge from the structure of the conceptual model (the typological level), the system must now search for relationships at instance-level as well in order to find the requested information in a model. For instance, information concerning costs may be available in the model by means of relationships defined for the different types of elements in a building, but in addition, certain costs may be added, as a specification, to particular instances of building elements that bring additional

costs to the construction process: ‘all steel columns of the type HE230A cost $x.xx per meter, but the one labelled D23 costs an additional $z.zz because it is harder to position.’

5. Conclusions and current work.

This paper demonstrates that extensibility and flexibility are essential requirements for information models that are to support architectural design and its dynamic character. The technologies of Feature-Based Modelling (FBM) in mechanical engineering address this extensibility and flexibility and are therefore very relevant for architectural design as well. The possibilities and benefits of applying these technologies in an architectural design support system may be a new stimulus in the discussion on product models in design, which seems to have stalled on the issue of rigidity of standardised core models.

Two important differences between the fields of application, mechanical engineering versus architecture, oppose the drawn analogy: (a) Features in architecture will have to cover information concerning both concrete and abstract concepts in architectural design; and (b) Features are relevant to multiple levels of abstraction with crossing relationships between these levels. The notion of different levels of abstraction requires an integration of the modelling approach discussed here with results of design research. This work will be carried out in a joint research with Achten (1997; 1998) who investigated the generic representation of typological design knowledge.

The paper discusses a framework for the application of architectural Features in the development of information modelling systems for support of architectural design. This framework distinguishes three levels of information definition: a meta layer defining classes of Feature types, a level of Generic Feature Types, to be standardised internationally, and Specific Feature Types for extension of the latter, and a level of actual Feature models containing instances of Feature types. Classification and standardisation of the level of Generic Feature Types has the potential of developing into a new sort of core models. The relation with the standardisation efforts in ISO 10303–STEP– and ISO 13584–Parts Library– are evident, though outside the scope of this paper. The authors are eager to open the discussion on this subject and to collaborate with experts in this field.

Currently, the approach of FBM for architecture is being developed further within the context of the newly defined research on the development of a design information system in virtual reality. This project, called VR-DIS[1], aims to develop a design system that can be used for interactive design and evaluation,

[1] VR-DIS is Virtual Reality – Design Information System / Distributed Interactive Simulations.

Paper at AID ’98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design ’98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

using a VR interface. The system will be based on a combination of the results from various, recent researches, including: design research on typological design knowledge (Achten, 1998); developments in multiple disciplines on design and evaluation, e.g. Mallory and Rutten (1997); development of Virtual Reality-based user interfaces (Coomans and Timmermans, 1997); research on the registration and evaluation of user-behaviour in a virtual building (Dijkstra and Timmermans, 1997); the techniques on constraint solving as developed in Kelleners (1997); and the Feature modelling approach as presented in this paper.

References

Achten, H.H. and Oxman, R.M.: 1998, Typological knowledge acquisition through a schema of generic representations, *in* Gero et al. (eds), *Artificial Intelligence in Design '98*.

Achten, H.H.: 1997, *Generic Representations – an approach for modelling procedural and declarative knowledge of building types in architectural design*, Ph.D. thesis, Eindhoven University of Technology, Eindhoven.

Bailey, I., (ed): 1997, *Requirements Specification for EXPRESS Mapping Language*, ISO 10303 TC184/SC4/WG11 N013, ISO, Geneva.

Bronsvoort, W.F., and Jansen, F.W.: 1993, Feature modelling and conversion, Key concepts to concurrent engineering, *Computers in Industry*, **21**(1), 61-86.

Bronsvoort, W.F., Dohmen, M., Bidarra, R., Van Holland, W., and De Kraker, K.J.: 1996, Feature modelling for concurrent engineering, *in* Horváth and Váradi (eds), *Proceedings of the International Symposium on Tools and Methods for Concurrent Engineering '96*, Technical University of Budapest, Budapest, pp.46-55.

Chermayeff, S. and Alexander, C.: 1965, *Community and privacy, toward a new architecture of humanism*. Anchor books.

Coomans, M.K.D. and Timmermans, H.J.P.: 1997, Towards a Taxonomy of Virtual Reality User Interfaces, *in Proceedings of the International Conference On Information Visualisation-IV*, August 27-29, 1997, London.

Coyne, R.D., Rosenman, M.A., Radford, A.D., Balachandran, M., and Gero, J.S.: 1991, *Knowledge-based design systems*, Addison-Wesley, Reading, Massachusetts.

Cunningham, J.J., and J.R. Dixon: 1988, Designing with features: the origin of features, *in Proceedings ASME Computers in Engineering Conference*, San Francisco.

Dijkstra, J. and Timmermans, H.J.P.: 1997, Exploring the Possibilities of Conjoint Measurement as a Decision-Making Tool for Virtual Way Finding Environments, *in* Liu, Yu-Tung (eds) *CAADRIA '97. Proceedings of The Second Conference on Computer Aided Architectural Design Research in Asia*, 17-19 April 1997, Hsinchu, Taiwan, Hu's Publishers, Taipeh.

DeMartino, T., Falcidieno, B., Giannini, F., Hassinger, S., and Ovtcharova, J: 1994, Feature-based modelling by integrating design and recognition approaches, *Computer-Aided Design*, **26**(8).

Eastman, C.M., and Jeng, T.S: 1997, *A database supporting evolutionary product model development for design*, working paper, Georgia Institute of Technology, Atlanta.

Eastman, C.M: 1996, Managing integrity in design information flows, *Computer Aided Design*, **28**(6/7), 551-565.

Eastman, C.M., Assal, H.H., and Jeng, T.S: 1995, Structure of a product database supporting model evolution, *in Modeling of buildings through their life-cycle* (proceedings workshop CIB W78 '95), CIB W78, Stanford University, pp.327-338.

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.

Eastman, C.M., Bond, A.H., and Chase, S.C.: 1991, A data model for design databases, *in* J.S. Gero (ed), *Artificial intelligence in design '91*, Butterworth Heinemann, Sydney, pp.339-365.

Gero, J.S., and Park, S.-H: 1997, Computable feature-based qualitative modeling of shape, *in* Junge, R. (ed) *Proceedings CAAD Futures '97*, TU München, München: pp.821-830.

ISO: 1994, Description methods: the EXPRESS language reference manual, ISO TC184/SC4 10303 part 11, International Organization for Standardization, Geneva.

ISO: 1996, *Building Construction Core Model, BCCM*, ISO TC184/SC4/WG3 N496, International Organization for Standardization, Geneva.

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

Lawson, Bryan: 1990, *How designers think, the design process demystified*, 2nd edition, Butterworth Architecture, London.

Mallory, S.M.H. and Rutten, P.G.S.: 1997, *Reducing risk through building performance evaluation with intelligent computer assistants*, working paper, Eindhoven University of Technology, Eindhoven.

Markus, Thomas A.: 1969, The role of building performance measurement and appraisal in design method, *in* Broadbent and Ward (eds), *Design methods in architecture*, Lund Humphries, London.

Maver, Thomas W.: 1970, Appraisal in the building design process, *in* Moore (ed), *Emerging methods in environmental design and planning*, MIT Press, Cambridge, Massachusetts.

Mitchell, W.J.: 1990, A new agenda for computer-aided design, *in* McCullough, Mitchell, and Purcell (eds), *The electronic design studio*, The MIT Press, Cambridge, Massachusetts.

Ramscar, M.: 1994, Static models and dynamic designs, an empirical impasse vs an inductive solution, *in* Scherer(ed), *Product and process modelling in the building industry*, Balkema, Rotterdam, pp. 69-76.

Shah, J.J., Mäntylä, M., and Nau, D.S. (eds): 1994, *Advances in Feature based manufacturing*, Elsevier Science, Amsterdam.

Shah, J.J: 1991a, Assessment of features technology, *Computer-Aided Design*, **23**(5), 331-343.

Shah, J.J: 1991b, Conceptual development of form features and feature modelers, *Research in Engineering Design*, **1991**(2), 93-108.

Simon, H.A.: 1973, The structure of ill-structured problems, *Artificial Intelligence*, **4**, 191-201.

Van Emmerik, M.J.G.M: 1990, *Interactive design of parameterized 3D models by direct manipulation*, Ph.D. thesis, Delft University Press, Delft.

Van Holland, W., and Bronsvoort, W.F: 1996, An object-oriented product structure for assembly modelling, *in* Tichem, Storm, Andreasen, and MacCallum (eds), *Proceedings of the 2nd WDK Workshop on Product Structuring*, Delft University of Technology, Delft.

Van Leeuwen, J.P. and Wagter, H.: 1997, Architectural design-by-Features, *in Proceedings CAAD Futures '97*, TU München, München, pp. 97-115.

Van Leeuwen, J.P., Wagter, H., and Oxman, R.M: 1996, Information modelling for design support - a Feature-based approach, *in Proceedings of the 3rd Conference on Design and Decision Support Systems in Architecture and Urban Planning*, August 18-21, 1996, Spa, pp. 304-325.

Paper at AID '98, Lisbon, 20-23 July 1998.
Publication in: Artificial Intelligence in Design '98, Eds. J.S. Gero and F. Sudweeks, Kluwer, Dordrecht, 1998.