A *SORTAL* BUILDING MODEL SUPPORTING INTERDISCIPLINARY DESIGN COMMUNICATION

Rudi Stouffs[1], Ramesh Krishnamurti[2] and Albert ter Haar[3]

ABSTRACT

Finding common ground is increasingly more important in building design as building projects are becoming more complex. Finding common ground generally requires common views of the same building information. We propose the concept of a *sortal* building model as an extension to the Building Information Model (BIM), offering the user the means to build up design representations, in support of common or interdisciplinary views, and to use such representations for querying building information. The concept of a *sortal* building model is based on a framework for representational flexibility named *sorts*, which provides formal support for the construction of design representations, and for the comparison of alternative representations in order to support translation and identify where exact translation is possible. In this paper, we compare the process of constructing design views to a complex adaptive system, in which the design view is as much an outcome of as a means to the design communication process. This comparison sheds light on the characteristics that distinguish the BIM and the *sortal* building model and on the functionalities that are needed to support the creation and use of a *sortal* building model for exploring different views in support of (interdisciplinary) design communication. We also briefly describe those aspects of the *sorts* framework that assist in developing these functionalities.

KEY WORDS

BIM, design representations, design communication, interdisciplinary, complex adaptive system

INTRODUCTION

Failure costs in building construction, in the Netherlands, collectively amount to about 5 to 6 billion euros, about 8 to 10% of the total building costs (Brokelman and Vermande 2005). Part of these costs is due to design mistakes as a consequence of poor communication among different disciplines involved in the building design process. Software developments to

[1] Associate Professor, Technical Design & Informatics Chair, Dept. of Building Technology, Faculty of Architecture, Berlageweg 1, 2628 CR Delft, The Netherlands, Phone +31 15/278-1295, FAX +31 15/278-4178, r.m.f.stouffs@tudelft.nl

[2] Professor, Graduate Program in Computational Design, School of Architecture, College of Fine Arts, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213-3890, Phone +1 412/268-2360, FAX +1 412/268-7819, ramesh@cmu.edu

[3] PhD. Researcher, Technical Design & Informatics Chair, Dept. of Building Technology, Faculty of Architecture, Berlageweg 1, 2628 CR Delft, The Netherlands, Phone +31 15/278-4485, FAX +31 15/278-4178, a.terhaar@tudelft.nl

support the building process have mainly focused on mono-disciplinary design and analysis applications. Research into integrated design environments, starting in the 1970's, has so far failed to impact the design practice on any global scale. More recently, research into product models, information standardization and Building Information Models (BIMs) has focused on common representational models for multidisciplinary building information. While these models support all-encompassing (centralized or distributed) collections of building information and data exchange between applications within and between disciplines, they do not necessarily support design communication.

Design communication does not only concern the exchange of building information between different partners and disciplines, but also implies a large amount of human communication concerning knowledge transfer, decision making and finding common understanding. Finding common ground is increasingly more important in building design as building projects are becoming more complex as the result of advances in technological abilities, the will to break new grounds, the increasing power of diverse interest groups, issues of liability, etc. Finding common ground generally requires common or, at least, similar views of the same building information. Such views cannot be too general, nor pre-defined, as they need to target the specific subject of communication, and the specific backgrounds of the partners involved. Neither domain-specific design and analysis applications, nor BIMs, provide adequate support for creating such views and, thus, for finding common ground.

We propose the concept of a *sortal* building model as an extension to a BIM, offering the user the means to build up design representations, in support of common or interdisciplinary views, and to use such representations for querying building information. The concept of a *sortal* building model is based on a framework for representational flexibility named *sorts*, which provides formal support for the construction of design representations, the comparison of alternative representations in order to support translation and identify where exact translation is possible, and the integration of functional components into design representations in order to specify design queries (Stouffs and Krishnamurti 2004). Quintessential to the concept of a *sortal* building model is the need to construct design views, either from scratch or by adapting an existing view.

In this paper, we compare the process of constructing design views to a complex adaptive system, in which the design view is as much an outcome of as a means to the design communication process. This comparison sheds light on the characteristics that distinguish the BIM and the *sortal* building model and on the functionalities that are needed to support the creation and use of a *sortal* building model for exploring different views in support of (interdisciplinary) design communication. We also briefly describe those aspects of the *sorts* framework that assist in developing these functionalities.

REQUIREMENTS FOR A DESIGN REPRESENTATION SYSTEM

The building domain, at all stages, is multi-disciplinary, involving participants, knowledge and information from various specializations. Problems in building design, therefore, require a multiplicity of viewpoints, each distinguished by particular interests and emphases. In the main, the architect is concerned with aesthetic and configurational aspects of a design, the structural engineer considers the structural members and their relationships, and the

performance engineer is interested in the thermal, lighting, or acoustical performance(s) of an eventual design. Each has views — derived from an understanding of current problem solution techniques in their respective domain — that require a different representation of the same (abstract) entity. Each view specifies a particular selection of information, presented in a particular way. As such, different views — or different representations — may derive from different design stages but may also support different persons or applications in the same design stage. Even within the same task, or by the same person, various representations may serve different purposes defined within the problem context and selected approach. This is especially true in architectural design, where the design process, by its exploratory and dynamic nature, invites a variety of approaches and representations (e.g., Kolarevic 2000).

As an activity in the design process, creativity relies on a restructuring of information that is not yet captured in a current information structure — that is, emergent information — for example, when the design provides new insights that lead to a new interpretation of constituent design entities. Creativity in design can be supported, to some extent, by descriptions of design entities with either definite or indefinite parts. When design descriptions have indefinite parts, new design entities can be recognized as alternative collections of design parts, and descriptions can be reinterpreted as composed of a different number of design entities. These descriptions can be augmented with properties that themselves have definite descriptions with definite or indefinite parts. Classic representation schemes also deal with definite descriptions but generally with definite parts and certainly properties are definite descriptions with definite parts. The classic BIM approach requires a specification of design entities as objects (with properties) that is maintained at all times, unless explicitly altered. Any reinterpretation of design entities, then, requires the specification of a computational change that not only fixes the source and destination object types beforehand, but also fixes their numbers and the mapping between properties. Continuity of such computational change requires anticipation of the particular structures that are to be changed (Krishnamurti and Stouffs 1997). Creativity, on the other hand, is outwith such anticipation.

It follows that, in the early phases of architectural design, the design representation is as much an outcome of as a means to the design process. Systems whose structure is "simultaneously both the means and the outcome of the social practices associated with elements of the system" (Kooistra 2002) present a special kind of systems. Kooistra refers to this mean/outcome mechanism as an ice canoe (Hough et al. 2001). We can consider the evolution of a design representation throughout the design process as such a system, in particular, a complex adaptive system. A complex adaptive system is the operational model of the complexity paradigm, which "uses systemic inquiry to build fuzzy, multivalent, multilevel and multidisciplinary representations of reality" (Dooley 1997). Dooley distinguishes two (or three) key principles to a complex adaptive system: "order is emergent as opposed to predetermined, and the state of the system is irreversible and often unpredictable". These principles also apply — or can be applied — to the evolution of a design representation in a dynamic design process. Considering that "order arises from complexity through self-organization" (Prigogine and Stengers 1984), the process of self-organization in the context of an evolving design representation takes on the form of human communication or correspondence leading to an agreement on the representation that prevails

in the system (see also Kooistra 2002). This communication can be considered among different users or between the user and the design application (correspondence between the user's mental model and the application's design representation). At the same time, the state or history of a design representation is in principle irreversible as changes to the representational structure can result in data loss. Since the design outcome is indeterminately related to the design requirements and the design process, and the design representation is an intricate part of this outcome, this representation is in principle also unpredictable.

Applied to a framework for building design representations in the context of a design process, these key principles can be translated into requirements of robustness and flexibility (see also Kooistra et al. 2003) with respect to the design representation system, and of the potential for data loss. Here, robustness means that the system offers the possibility for correspondence (or communication) leading to an agreement on the representation that prevails in the system. At the same time, the system must offer the possibility for representations to change and in such a way that, in principle, claims on this representation generate quality improvement. "A claim has the desired quality if in the construction process attention has been devoted to the construction progressing in such a way that using the claim serves to improve the quality" (Groen et al. 1980, see also Kooistra 2002). Laying claims can be considered as the engine of the design process, also with respect to (the design of) the design representation.

FORMALLY RELATING REPRESENTATIONS

Typically, a representation is a complex structure of properties (or attributes) and constructors, and a representation may be a construction of another (Stouffs et al. 1996). Van Leeuwen et al. (2001) describe a property-oriented data modeling approach, in which design concepts are represented as flexible networks of objects and properties. In contrast to the classic BIM approach, an object has no predefined set of properties and the composition of properties defining an object can be changed at any time. Concept modeling (van Leeuwen and Fridqvist 2003; see also van Leeuwen 1999) provides an elaboration of this approach. It distinguishes concepts and individuals, both defined in terms of properties, where concepts represent modeling schemata and individuals represent particular designs. A concept defines an individual's initial structure of properties, however the relationship between an individual and a concept is a loose one: properties can be added to an individual irrespective of whether its concept predefines these same properties; individuals can also relate to multiple concepts or relate to more specific concepts over time. As such, concept modeling allows for the extensibility of conceptual schemata and for flexibility in modeling information structures that differ from the conceptual schemata these derive from. Under the property-modeling approach, correspondence can be achieved through the evolution of the network of objects and properties and by agreement on the naming of objects. These names can themselves be understood as laying claim to these objects with the purpose of improving quality. Thus, modeling design representations through incremental changes can play an important role in achieving agreement and thus in containing the "chaos" to which the construction of design representations within a flexible framework can lead.

Such an incremental approach, however, can greatly benefit from a formal framework that allows for alternative representations of a same entity to be compared and related,

formally, in order to support translation and identify where exact translation is possible. For example, Stouffs et al. (1996) were able to show, using a subsumption relation defined on well known solid models — boundary solid representations (Baumgart 1975, Mäntylä 1988, Paoluzzi et al. 1989) and the maximal element representation (Krishnamurti 1992) — that information loss between some of these solid models is inevitable. Subsumption is a powerful mechanism for comparing alternative representations of the same entity. When a representation is subsumed by another, the entities represented by the former can also be represented by the latter representation, without any data loss. There are many representational formalisms that consider the subsumption relationship in order to achieve partially ordered type structures; most are based on first-order logic. Applied to building design, a good example is Woodbury et al. (1999), who adopt typed feature structures as the model for design space exploration. Like many other formalisms, typed feature structures consider a record-like data structure for representing data types. Record-like data structures facilitate the encapsulation of property information in (a variation of) attribute/value pairs (Aït-Kaci 1984). Furthermore, the properties may themselves be typed by type structures, i.e., expressed in terms of record-like data structures, containing (sub)properties. Then, the subsumption relationship defines a partial ordering on type structures. Furthermore, the algebraic operations of intersection and union (or others similar) may be defined on type structures so that the intersection of two type structures is subsumed by either type structure, and the union of two type structures subsumes either type structure.

Key to typed feature structures is the notion of partial information structures and the existence of a unification procedure that determines if two partial information structures are consistent and, if so, combines them into a single, new (partial) information structure. Typed feature structures further consider a type hierarchy and a description language, where each type defines a corresponding description. The subsumption relation between feature structures extends the subsumption ordering on types inherent to the type hierarchy. Woodbury et al. (1999) also specify a generating procedure that relates feature structures with a description (or type) that they satisfy, and that incrementally generates more complete design structures. This fact — that the generating procedure monotonically generates more complete information structures — could be interpreted as excluding the possibility for information loss and thus making design states reversible. However, the inclusion of an information removal operator is possible providing more flexibility at the cost of limiting search strategies (Woodbury et al. 1999).

A SEMI-CONSTRUCTIVE ALGEBRAIC FORMULATION

Of course, there is more than one approach to formally model design descriptions. We consider a semi-constructive algebraic formulation, termed *sorts*. The premise here is that whenever individuals are confronted with information, they naturally classify it according to their own needs and understanding — sort it out, so to speak. *Sortal* descriptions of design entities can have definite or indefinite parts, and can be augmented with properties that are, themselves, *sortal* descriptions and, therefore, can have definite or indefinite parts. *Sortal* descriptions can also combine under a subsumption relationship. *Sortal* descriptions may be shared and they have implications for information transfer, information coverage and information loss.

A BEHAVIORAL SPECIFICATION FOR *SORTS*

"In functional semantics, if two functions exhibit the same behavior they are the same. In type feature structures, if two paths exhibit analogous properties, they represent analogous design (or reuse) cases" (Krishnamurti 2006). However, difficulties arise when dealing with information of different types in a uniform way. For instance, at the representational level, operations that may otherwise seem trivial, such as adding or removing data elements, become resolutely non-trivial — for instance, the addition of two numbers when these represent cardinal values (e.g., a number of columns that is increased) and when these represent ordinal values (e.g., for a given space, determining the minimum distance to a fire exit or the (maximum) amount of ventilation required given a variety of activities), and similarly, additive versus subtractive colors, depending on whether these refer to the mixing of surface paints or colors of light, respectively.

An important ingredient of *sorts* is behavioral specification. Behavioral specification is a prerequisite for the effective exchange of data between various representations. Fortunately, it is reasonably limited to the common arithmetic operations of addition, subtraction, and product. It turns out that the more common CAD operations of creation and deletion, and selection and deselection, can all be expressed as some combination of addition and subtraction from one design space (*sort*) to another. The complex operations of grouping and layering can be treated likewise.

The simplest specification of a part relationship corresponds to the subset relationship on mathematical sets. This part relationship particularly applies to points and labels, e.g., a point is part of another point only if the two are identical, and a label is a part of a collection of labels only if it is identical to one of the labels in the collection. Then, operations of addition (combining elements), subtraction, and product (intersecting elements) correspond to set union, difference, and intersection, respectively. Explicit designer action is required in order to alter any data element. Only when two elements are identical can these combine as one.

Another kind of behavior arises when we consider the part relationship on line segments. A line segment is an interval on an infinite line carrier; in general, one-dimensional quantities such as time may be considered as intervals. An interval is a part of another interval if it is embedded in this interval; intervals on the same carrier that are adjacent or overlap combine into a single interval. Specifically, interval behavior can be expressed in terms of the behavior of the boundaries of intervals (Krishnamurti and Stouffs 2004). This behavior also applies to infinite intervals, provided there is an appropriate representation of both (infinite) ends of its carrier.

Behaviors also apply to composite *sorts*, that is, a part relationship can be defined for its component data elements belonging to a composite *sort* defined under a conjunction (attribute or property operator) or disjunction. The composite inherits its behavior from its components in a manner that depends on the compositional relationship.

The disjunctive operator distinguishes all operand *sorts* such that each data element belongs explicitly to one of these *sorts* — the disjunctive *sort* subsumes each operand *sort*. For example, a *sort* of points and lines distinguishes each data element as either a point or a line. Consequently, a data element is part of a disjunctive data collection if it is a part of the partial data collection of elements from the same component *sort*. In other words, data

collections from different component *sorts*, under the disjunctive operator, never interact; the resulting data collection is the set of collections from all component *sorts*. When the operation of addition, subtraction or product is applied to two data collections of the same disjunctive *sort*, the operation instead applies to the respective component collections.

Under the attribute operator a data element is part of a data collection if it is a part of the data elements of the first component *sort*, and if it has an attribute collection that is a part of the respective attribute collection(s) of the data element(s) of the first component *sort* it is a part of. When data collections of the same composite *sort* (under the attribute operator) are pairwise summed (differenced or intersected), identical data elements merge, and their attribute collections combine, under this operation. Elements with empty attributes are removed and the composite behavior is that, in the first instance, of the first component *sort*.

When reorganizing the composition of components *sorts* under the attribute operator, the corresponding behavior may be altered in such a way as to trigger data loss. Consider a behavior for weights (Stiny 1992) (e.g., line thickness or surface tones) as becomes apparent from drawings on paper — a single line drawn multiple times, each time with a different thickness, appears as if it were drawn once with the largest thickness, even though it assumes the same line with other thickness. When using numeric values to represent weights, the part relation on weights corresponds to the less-than-or-equal relation on numeric values. Thus, weights can combine into a single weight, which has as its value the least upper bound of all the respective weight values, i.e., their maximum value. Similarly, the common value (intersection) of a collection of weights is the greatest lower bound of all the individual weights, i.e., their minimum value. The result of subtracting one weight from another is either a weight that equals the numeric difference of their values or zero (i.e., no weight), and this depends on their relative values.

Now consider a *sort* of weighted entities, say points, i.e., a *sort* of points with attribute weights, and a *sort* of pointed weights, i.e., a *sort* of weights with attribute points. A collection of weighted points defines a set of non-identical points, each having a single weight assigned (possibly the maximum value of various weights assigned to the same point). These weights may be different for different points. The behavior of the collection is, at first instance, the behavior for points. On the other hand, a collection of pointed weights, which is defined as a single weight (which is the maximum of all weights considered) with an attribute collection of points, adheres, at first instance, to the behavior for weights. In both cases, points are associated with weights. However, in the first case, different points may be associated with different weights, whereas, in the second case, all points are associated with the same weight. In a conversion from the first to the second *sort*, data loss is inevitable. An understanding of when and where exact translation of data between different *sorts*, or representations, is or is not possible, becomes important for assessing data integrity and controlling data flow (Stouffs et al. 1996).

Behavioral specification is a prerequisite for a uniform handling of different and a priori unknown data structures. The behavior of such data can be expressed through a number of operations chosen to match the expected behavior. When an application receives data along with its behavioral specification, the application can then correctly interpret, manipulate, and represent this information without unexpected data loss. The part relationship that underlies the behavioral specification for a *sort* enables matching to be implemented for this *sort*; since

composite *sorts* inherit their behavior and part relationship from their component *sorts*, any technical difficulties in implementing matching apply just once, for each primitive *sort*.

DATA RECOGNITION USING *SORTS*

Logic-based models essentially represent knowledge; *sorts*, on the other hand, represent data — any reasoning is based purely on present or emergent information.

The typed feature structures formalism, like most logic-based formalisms, links subsumption directly to information specificity, that is, a structure is subsumed by another, if this structure contains strictly more information than the other. One consequence of (logical) subsumption is that the absence of information in a design representation does not necessarily imply the absence of this information in the design, that is, representations are automatically considered to be incomplete. As a result, when searching for a design (representation) that satisfies certain information, less specific representations cannot automatically be excluded (e.g., Baader et al. 2003). For example, consider polygon objects that may have a color assigned. When looking for a yellow square, a square without any color specified is considered a potential solution — unless, it has another color explicitly specified, or it is otherwise known not to have the yellow color. The fact that a color is not specified does not exclude an object from potentially being yellow. Nothing can be excluded unless it is explicitly excluded — that is, logic-based representations consider an open world assumption. *Sorts*, on the other hand, hold to a closed world assumption. A polygon only has a color if one is explicitly assigned: when looking for a yellow square, any square will not do, unless it has the yellow color assigned.

Recognizing information, especially of the emergent kind, is invaluable, rather, necessary, should the information be, subsequently, the subject of action. From a creativity standpoint, design relies on an ability to restructure emergent information. Data recognition and subsequent manipulation can be considered part of a single computation:

$s - f(a) + f(b)$.

Here s is a data collection, a is a representation of the data pattern, f is a transformation under which a is a part of s, and $f(b)$ is the data replacing $f(a)$ in s. $s - f(a) + f(b)$ is an expression of computational change and can be written as a design rule: $a \rightarrow b$. Rule application consists of replacing the emergent data corresponding to a, under some allowable transformation, by b, under the same transformation.

Formally, rules may be grouped as a *grammar* — a device for specifying the set of all designs generated by the rules collectively. Each generation of a design in the language starts from an initial design, and uses the rules to create a design that contains elements from a given terminal vocabulary. Rules and grammars specified as such, lead naturally to the generation and exploration of possible designs. According to Mitchell (1993) and Stiny (1993), in the case of creative spatial design, spatial elements that emerge under a part relation are highly enticing to design search. The closed world assumption is commonly used in design grammars to constrain emergence. More specifically, labeled points commonly serve to constrain the applicability of shape rules. *Sorts* provide a component-based approach to developing grammar systems, utilizing a uniform characterization of grammars (Stouffs and Krishnamurti 2001).

CONCLUSIONS

There will always be a need for different representations of the same entity, whether it be a building in its entirety that is under consideration, or a part of a building, albeit a shape, or some other complex collection of properties. Considering a representation as a complex structure of properties and constructors, then, comparing alternative representations requires a comparison of their respective properties and their mutual relationships, and the overall construction. At the same time, the expressive power of a representational framework is defined by its vocabulary of properties and constructors. A formal definition of this representational framework and its vocabulary can give designers the freedom and flexibility to develop or adopt representations that serve their intentions and needs, while at the same time these representations can be compared with respect to scope and coverage in order to support information exchange and communication. Such a comparison will not only yield a possible mapping, but also uncover potential data loss when moving data from less-restrictive to more-restrictive representations.

ACKNOWLEDGMENTS

The second author is funded by a grant from the National Science Foundation, CMS #0121549, support for which is gratefully acknowledged. Any opinions, findings, conclusions or recommendations presented in this paper are those of authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

Aït-Kaci, H. (1984). *A lattice theoretic approach to computation based on a calculus of partially ordered type structures (property inheritance, semantic nets, graph unification)*. Ph.D. Diss., University of Pennsylvania, Philadelphia, PA, 187 pp.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P. (2003). *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, Cambridge, 574 pp.

Baumgart, B.C. (1975). "A Polyhedron Representation for Computer Vision." *National Computer Conference 1975*, AFIPS Press, Montvale, NJ, 589-596.

Brokelman, L. and Vermande, H. (2005) *Faalkosten, de (bouw)wereld uit!* SBR, Rotterdam, 46 pp.

Dooley, K.J. (1997). "A complex adaptive systems model of organization change." *Nonlinear Dynamics, Psychology, and Life Sciences*, 1 (1) 69-97.

Groen, P., Kersten, A., and de Zeeuw, G. (1980). *Beter sociaal veranderen*. Coutinho, Muiderberg, The Netherlands, 176 pp.

Hough, R.R., Kooistra, J., and Hough, T. (2001). "Growing intelligent agents for the delivery of knowledge, a Collaborative Form." *Systemica*, 13, 343-348.

Kolarevic, B. (2000). "Digital Morphogenesis and Computational Architectures." *In* Ripper Kos, J., Pessoa Borde, A. and Rodriguez Barros, D. (editors), *Construindo n(o) espaço digital*, PROURB, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 98-103.

Kooistra, J. (2002). "Flowing." *Systems Research and Behavioral Science*, 19 (2) 123-127.

Kooistra, J., Stouffs, R., and Tunçer, B. (2003). Metadata as a means for correspondence in design analysis. *In* Tunçer, B., Ozsariyildiz, S.S., and Sariyildiz, S. (editors), *E-Activities in Design and Design Education*, Europia, Paris, 19-28.

Krishnamurti, R. (1992). "The Maximal Representation of a Shape." *Environment and Planning B: Planning and Design*, 19 (3) 267-288.

Krishnamurti, R. (2006). "Explicit design space?" *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 20 (2) 95-103.

Krishnamurti, R. and Stouffs, R. (1997). "Spatial change: continuity, reversibility and emergent shapes," *Environment and Planning B: Planning and Design*, 24 (3) 359-384.

Mäntylä, M. (1988). *An Introduction to Solid Modeling*. Computer Science Press, Rockville, MD, 401 pp.

Paoluzzi, A., Ramella, M., and Santarelli, A. (1989). "Boolean Algebra over Linear Polyhedra." *Computer Aided Design*, 21 (8) 474-484.

Prigogine, I. and Stengers, I. (1984). *Order out of Chaos*. Bantam Books, New York, 349 pp.

Stiny, G. (1992), "Weights." *Environment and Planning B: Planning and Design*, 19 (4) 413-430.

Stouffs, R. and Krishnamurti, R. (1996). "On a query language for weighted geometries." *In* Moselhi, O., Bedard, C., and Alkass, S. (editors), *Third Canadian Conference on Computing in Civil and Building Engineering*, Canadian Society for Civil Engineering, Montreal, 783-793.

Stouffs, R. and Krishnamurti, R. (2001). "Sortal grammars as a framework for exploring grammar formalisms." *In* Burry, M., Datta, S., Dawson, A., and Rollo, J. (editors), *Mathematics and Design 2001*, The School of Architecture & Building, Deakin University, Geelong, Australia, 261-269.

Stouffs, R. and Krishnamurti, R. (2004). "Data views, data recognition, design queries and design rules." *In* Gero, J. (editor), *Design Computing and Cognition '04*, Kluwer Academic, Dordrecht, The Netherlands, 219-238.

Stouffs, R., Krishnamurti, R., and Eastman, C.M. (1996). "A Formal Structure for Nonequivalent Solid Representations." *In* Finger, S., Mäntylä, M., and Tomiyama, T. (editors), *Proc. IFIP WG 5.2 Workshop on Knowledge Intensive CAD II*, International Federation for Information Processing, Working Group 5.2, Pittsburgh, PA, 269-289.

van Leeuwen, J.P. (1999). *Modelling Architectural Design Information by Features*. PhD Diss., Eindhoven University of Technology, Eindhoven, The Netherlands, 276 pp.

van Leeuwen, J.P. and Fridqvist, S. (2003). "Object version control for collaborative design." *In* Tunçer, B., Ozsariyildiz, S.S., and Sariyildiz, S. (editors), *E-Activities in Building Design and Construction*, Europia, Paris, 129-139.

van Leeuwen, J.P., Hendrickx A., and Fridqvist, S. (2001). "Towards dynamic information modelling in architectural design." *Proc. CIB-W78 International Conference IT in Construction in Africa*, CSIR, Pretoria, 19.1-19.14.

Woodbury, R., Burrow, A., Datta, S., and Chang, T. (1999). "Typed feature structures and design space exploration." *Artificial Intelligence in Design, Engineering and Manufacturing*, 13 (4) 287-302.