

CLASSIFICATION OF PRECEDENTS

A Hybrid Approach to Indexing and Retrieving Design Cases in SEED (a Software Environment for the Early Phases of Building Design)

Z. AYGEM AND U. FLEMMING

School of Architecture, Carnegie Mellon University, Pittsburgh PA

Abstract. An efficient indexing of past solutions is crucial to case-based design (CBD) systems performing complex retrieval on large case-bases. This paper suggests a hybrid approach to the indexing and retrieval of design precedents. The suggested approach accounts for the issues of classification manifested in architectural discussions on type and CBD literature. The indexing scheme integrates description-logic based representation for classification and an object-based representation for precedents. The hybrid scheme constitutes a basis for the implementation of a generic case indexing and retrieval mechanism for SEED.

1. Introduction

Recent developments in computational design have extended the case-based design (CBD) approach, a design specific application of the AI paradigm of case-based reasoning (CBR), to the context of architectural design (Rosenman, et.al., 1992). This extension can be conceived as a continuation of a common practice: the use of *precedents* in design. The term *precedent*, introduced to the computational design literature by Oxman (1994), refers to a representation of the knowledge about a past design in a form that makes it applicable or “re-usable”, in new, but similar problem situations. The use of the term in the present paper, however, does not inherit Oxman’s knowledge organization scheme.

We argue in this paper that a CBD system for architectural design requires a modelling scheme to represent and classify architectural precedents in a CBD system. To develop such a scheme, we must study the relationships between classification concepts and classified design precedents within an information retrieval system. Some salient issues are identified and elaborated independently in the architectural discussion of type and the CBD literature. The typological discussion demonstrates the complexities involved in dealing

with the categorization of precedents and accounts for some of the issues that need to be addressed by appropriate indexing and retrieval mechanisms. However, the transfer of ideas between the literature in the two domains is difficult because they use different terminologies and lack a common reference framework. In order to resolve this issue, we introduce a memory model sketched after Smith and Medin's survey of the literature on the representation of concepts and categories (1981). We, then, argue with respect to architectural typology and CBD that:

- classifications in a case-base must allow for modifications if the CBD system is expected to incorporate new information on cases. Therefore, case retrieval mechanisms may have to cope with partial index descriptions and multiple classifications, which are abundant in architectural design.
- classifications may have to incorporate thematic features as well as features reflecting subjective judgments on cases. In many cases, these features cannot be derived from the symbolic representation of a design precedent.
- classifications speed up the retrieval of cases by allowing the system presort the case-base so that matches must be performed only on a subset of the case-base.

Consequently, our approach decouples case representation and classification in its data modeling and retrieval techniques.

The SEED project provides our first implementation environment and testing ground for the hybrid memory model we suggest. SEED requires the persistent storage of design precedents and their retrieval for re-use in a multi-functional, multi-user, and distributed design environment. This necessitates the use of a generic, flexible, and extensible classification scheme. In the following sections, we will outline the issues regarding the implementation of SEED's case indexing and retrieval engine.

2. Classification of Precedents

The majority of knowledge bases assisting CBR problem solvers follow Tulving's memory model, which distinguishes the semantic aspects of human memory from the episodic ones (Tulving, 1972). Episodic memory deals with personal experiences and their simple temporal relations, whereas semantic memory deals with language faculties that receive, retain, and transmit information about meaning and with the classification of concepts. Sowa makes a similar distinction, but bases it on the nature of what is stored, rather than the mechanisms of each memory (1984). The episodic memory stores detail facts about individual things and events as episodes, whereas the semantic memory contains more abstract and generic information, or the universal principles in Sowa's account.

Smith and Medin review the psychology literature on concept acquisition and categorization (1981) and suggest a "mixed representation" for concepts and categories. Their survey groups the research on the representation of

concepts and categories under the *classical*, *probabilistic* and *exemplar* views. In the *classical* view, all instances of a concept share common properties, and these properties are necessary and sufficient to define the concept. In this view, concept representations consist of a *summary description*, which applies to all possible test instances. The *probabilistic* view suggests that instances vary in the degree to which they represent a concept and consequently, in the degree to which they share certain properties. The assumption that characterizes the probabilistic view is that a concept is represented by a *summary description* that cannot be restricted to a set of necessary and sufficient conditions; rather, it is a measure of central tendency. Finally, the *exemplar* view proposes that there is no single representation of an entire class but only a set of specific representations of the class's *exemplars*. The basic premise of this approach is that people make extensive use of examples when they categorize.

The classical view has its limitations in terms of defining a unified description for its instances; however it offers a reliable inference mechanism to determine class membership because it deals with concept properties that are necessary and sufficient. The probabilistic and specially exemplar views, on the other hand, provide better models to represent artificial concepts. The mechanisms employed by these views do not yield an absolute true or false result to a class membership test; they provide a probabilistic inference or a degree of membership. Moreover, the degree of freedom provided by the disjunctive representation in the case of the exemplar view causes computational inefficiency in determining class membership. This comparison hints at the possibility of using the best of three worlds in a hybrid representation or a "mixed representation".

We suggest a hybrid memory model consisting of summary descriptions (that is, classification instances that apply to all the exemplars of a concept) and an open set of encountered exemplars. Consequently, the class membership test in this approach can employ an inference mechanism that operates on the summary descriptions in coordination with a matching mechanism operating on the exemplars.

3. Classification in Architectural Typology

The architectural discourse on type is one of the richest in design theory. It very often derives its effectiveness and power from a confused agreement or a cultural consensus on a vague definition of type (Bandini, 1984). It is important for our purpose, however, to reduce the ambiguities inherent in the definition of type. We therefore restrict our notion of type to a classificatory device, to a fundamental conceptual structure used in the categorization process.

Among the attempts to define type in the architectural context Moneo comes closest to the summary description or classification in the previously

introduced hybrid model (Moneo, 1978). Moneo conceives type as a concept that describes a group of objects characterized by the same “formal structure”. This formal structure should not be taken literally as a spatial diagram, but as a grouping of objects by certain inherent structural similarities. We will elaborate on these “inherent structural similarities” in connection with the multiplicity of groupings.

Two concepts in typological literature may help us to clarify what type represents with respect to a group of precedents: the model and the typological series. A model is a specific example or mechanical reproduction of an object (Argan, 1963); it can be conceived of as an exemplar. Argan refers to a series of “formal variants” or - more precisely - to a group of buildings exhibiting a formal and functional analogy as a typological series. Type is formed through a process of reducing a “complex of formal variants” or a series of instances to a common root form. In this definition, a formal variant or an instance exhibits the characteristics of the type governing the series simply because the type itself is deduced from these instances. In the hybrid model, these instances or formal variants are again exemplars, and the dependency is maintained through the summary description's encapsulation of a unitary description that applies to all these exemplars.

The formal structure that Moneo refers to, is sometimes defined by a deeper geometry (e.g., topology, centrality, linearity, grids). But Moneo argues that this reduces the idea of type to abstract geometry, whereas a formal structure in his sense reflects a vast hierarchy of concerns running from social interactions to building construction; ultimately, the group defining a type must be rooted in this reality as well as in an abstract geometry (1978). This position is supported by the fact that a fairly large number of typologies have been proposed to account for functional, institutional, formal, compositional, structural, and historic aspects of architectural objects. Some of these classifications can be merged, whereas some remain orthogonal. Moreover, in the design problem solving context, the same object may be grouped repetitively under different types depending on the design stage and the particular goals identified for the design problem at hand. In short, it is necessary to support multiplicity in the groupings or classifications for architectural precedents.

4. Classification in CBD

The appropriateness of the selections is one of the most important criteria in determining how useful the CBD approach can be. The following issues demonstrate the central role classification may play in case indexing and retrievals:

- **Computational efficiency in retrieval:** For some case-base actions, such as making a diagnosis or assessing a situation, indexing efficiency may be insignificant for retrieval

because case selection relies primarily on surface and contextual features (Waltz, 1988). In design, however, case retrieval is likely to involve lengthy comparisons of compositional and geometric properties of the cases. An efficient indexing scheme is needed to make the retrieval computationally tractable and to speed up the process by performing the matching only on a subset of cases filtered out from the entire case-base.

- **Use of thematic features in indexing:** An indexing vocabulary is defined to be a subset of the vocabulary used for symbolic representation of cases (Kolodner, 1994). In complex problem solving activities such as design, the retrieval may require the use of thematic features (e.g., goal, function, behavior). These features are obtained through an elaboration and interpretation of generalized models of the design domain (Maher et.al., 1995). As previously discussed, classifications in an architectural context (e.g., functional, institutional, formal, compositional, structural) often rely on features that reflect subjective judgments about the cases. If the classification vocabulary is assumed to be a subset of the case representation vocabulary, the features that cannot be computed from the case representation will be dismissed from the indexing vocabulary. Therefore, the case retrieval based on thematic features requires the indexing to provide a separate classification scheme that can incorporate these features.

- **Flexibility:** A CBD system performing tasks in an open world is likely to encounter incomplete knowledge, which may manifest itself as (1) incomplete knowledge of categories, (2) incomplete domain theories, or (3) under-specified problems (Hinrich, 1992). The indexing scheme and retrieval mechanism of a CBD system working in an open world are highly affected by the first and third type of incompleteness. Case-based design systems must, therefore, include open categories or unbounded sets, which are widely used in classifying design precedents: New classification concepts may be added to the system, and existing classifications may have to be modified. The need to incorporate partially described or multiple classifications in the case indexing scheme is further demonstrated through the multiplicity of groupings inherent in architectural typologies.

We suggest that a memory model in CBD must integrate specific design cases with an indexing scheme that allows for the partitioning of the case-base using the generic knowledge of a design model. Accordingly, we suggest a hybrid representation model that integrates a classification knowledge-base and a case-base. In this model, a case contains a classified design precedent (an exemplar). A classification is a description of necessary and sufficient properties that apply to all that it classifies. A classification can inherit descriptions from other classifications and these descriptions may incorporate thematic features. When a case is added to the case-base, multiple classifications are manually or automatically assigned to it, and the classification knowledge-base is updated to reflect these changes. The indexing engine should detect and report inconsistencies between the classification instances and the classified cases that may arise as a result of any update in the classification knowledge-base or the case-base.

5. Case-Based Design Support in SEED (SEED-CBD)

SEED is a software environment which aims at providing computational support for the early phases in building design. It intends to encourage an exploratory mode of design by making it easy for designers to generate and evaluate alternative design concepts and versions. SEED consists of modules that address specific tasks within the overall preliminary design. The tasks supported at the present time are architectural programming, schematic layout design and the generation of a fully 3-dimensional configuration of physical building components (Flemming & Woodbury, 1995). An object database allows designers to store and retrieve past designs that can be re-used and adapted in different contexts.

SEED-CBD provides access to a fairly large number of past solutions possibly generated by different modules, and hence eases the recalling process by supporting systematic indexing. The reuse of past solutions is generic in the sense that it extends across modules in SEED. Moreover, problems addressed by SEED modules are often decomposed into hierarchies of subproblems where standard solutions can be found at any level of the hierarchy. Therefore the reuse capability extends across the levels of problem decomposition hierarchy.

Modules in SEED adopt a case representation scheme where a case is typically composed of the *triplet*: $\langle \textit{problem}, \textit{solution}, \textit{evaluation} \rangle$. The definition of each triplet element varies from module to module. A triplet element designates an object configuration with attributes and constraints incorporating relational hierarchies that are orthogonal (e.g., constituent and inheritance hierarchies). The definition of a case index determines the way a case is recalled by the system. Modules have different sets of characteristics to index a case. Often, these characteristics cannot be confined to a single triplet element.

The *generality* and *separation of the classification from the matching inference*, respectively, are the major criteria for the design of SEED-CBD's support for indexing and retrieval. The former criterion manifests itself in terms of a simple and common interface for case-base operations. Each module provides the case index, problem, solution, and outcome, which are merely generic containers. Accordingly, a module's account of how the retrieval is performed is captured in the content of a generic target. The common interface also decouples the indexing and retrieval system from its clients so that the system does not have to go through a major change when a new sub-system is introduced to SEED. The latter criterion reflects the differences between the two inference engines that are part of the case retrieval mechanism. The classification inference yields a TRUE or FALSE to a *is-a?* query, whereas the matching yields a degree of similarity. Classifications are represented by

relatively simple data structures which allows the engine to make complex inferences. The system can infer subsumption relations from classifications instead of relying on the direct assertions of these relations. The simplicity of representations also allows for a safer use of multiple inheritance. The matching inference, on the other hand, deals with fairly complex object structures. To assure polymorphism, SEED modules use single inheritance in their object-based representation. The matching inference, therefore, deals with single inheritance hierarchies. The mechanisms for classification and matching inference can be modeled separately; however, they need to be coordinated during retrieval.

The hybrid modeling approach that complies with the latter criterion is manifested in the definition of *case descriptor* and *target descriptor*. An account of how various inference mechanisms work in connection with these definitions is provided at the end of this section.

A *case descriptor* is the means of specifying the basis of retrieving a case from the case-base. It contains pointers to the objects to be matched along with their *classifications* and *match indices*. A *classification* is an aggregation of *classifiers (descriptions)* describing an object in the data model. The object, in turn, is a class instance. A *classifier* may be restricted to a particular class or may classify objects of any class. A class inherits the attributes and behavior of its superclass. The restriction to single object inheritance reduces the number of cases on which matching is performed during the retrieval. Only the case descriptors with an object that belongs to the class or to a subclass of the target object are considered.

In SEED, the classification may involve any number of inheritance hierarchies that are orthogonal (e.g., a classification **Army Headquarter Firestation** combines three concepts of orthogonal classification hierarchies i.e. ownership, organization and function related classifications). Classifiers can be attached to objects thereby allowing them to belong to multiple classifications. This greatly reduces the problems that may arise from SEED's insistence on single object inheritance. Our definition of classifiers uses a description-logic based knowledge representation scheme. Classifiers are not meant to work with complex data models incorporating geometry, tuples, and series; they are simple descriptions represented by relatively shallow data structures. The classifier inference mechanism works with a taxonomy of classification concepts based on the subsumption relationship; it efficiently answers questions regarding subsumption.

In addition to a classification, a case descriptor may incorporate *match indices*. A *match index* is a special representation of a case descriptor object used by a specific matching algorithm. For a matching algorithm operating on constituent hierarchies, a match index can be the transitive closure of an object which contains pointers to its direct and indirect constituents. In this manner the

match index shortcuts the traversal of the constituents by reducing the number of recursive calls. Unlike classification indices match indices are computed once when a case is stored into a case base and can be checked out only during retrieval.

A *target descriptor* structures case retrieval queries and consists of an aggregation of matching criteria. A *matching criterion*, in turn, consists of the following:

- A *target matchable* is a class description based on which an object is instantiated for matching during retrieval. A target matchable may contain attribute descriptions that specify values for relational and simple attributes. Each class description in a target matchable can be assigned a classification to trigger the subsumption inference prior to matching.
- A *match operator* is used to specify the matching algorithm(s) to be used on a particular target matchable. Match operators can be disjointed or conjoined to form an operator allowing multiple matching algorithms to be applied to one matchable.

The retrieval mechanism for SEED-CBD reflects the bipartite structure of the suggested indexing scheme. There are two major inference engines working with the suggested indexing mechanisms: classification inference and the matching for the objects specified in the case descriptor. A classification (C_i) for a target matchable may be a new classifier or an aggregation of existing classifier instances merged into a temporary description. In the first phase, a query is sent to a corresponding knowledge-base to find classifier instances that are subsumed by C_i . For each subsumed instance, the classified objects are retrieved, and associated cases are aggregated to form a set (S_i). In the second phase, the class name provided in the target descriptor (class_{*i*}) is used to limit the objects to be matched in S_i to the ones that are of the same class as class_{*i*} or to its sub-classes. The matching process involves the use of various match operators for a comparison of case descriptor components with target matchables. Matching criteria incorporate various kinds of objects and the corresponding match operators. Our definition of the target descriptor allows the SEED modules to select an existing match algorithm or to specify their own during retrieval. SEED-CBD provides the default matching algorithms (i.e., *base-match* and *deep-match*). *Base-match* traverses the objects attribute-by-attribute to compare the associated values. The traversal does not proceed into the attribute hierarchies. *Deep-match*, on the other hand, performs matching on constituent hierarchies (Flemming et.al., 1996).

The classification inference and matching can work in coordination or independently depending on the target specification. If the target includes a classification but no object to be matched, only the classification inference is activated. If it includes an object to be matched and no classification, the case retrieval engine activates the match operator specified in the target descriptor to perform a match between the objects specified. In the latter scenario, all cases in the case-base together with the match indices that can be used by the match

operator are checked out for matching. Finally, if a target provides a classified matchable (as in *Figure 1*) the case retriever activates the classification inference in advance to reduce the number of matching candidates. For the matching of two objects, the measure of similarity is a weighted sum of matched attributes with a range of 1 to 100; for classification inference, it is a binary value corresponding to whether the classification of the target descriptor component subsumes the classification of a case descriptor component. Upon completion of retrieval, recalled cases are ranked based on the value representing the weighted sum over the matched objects.

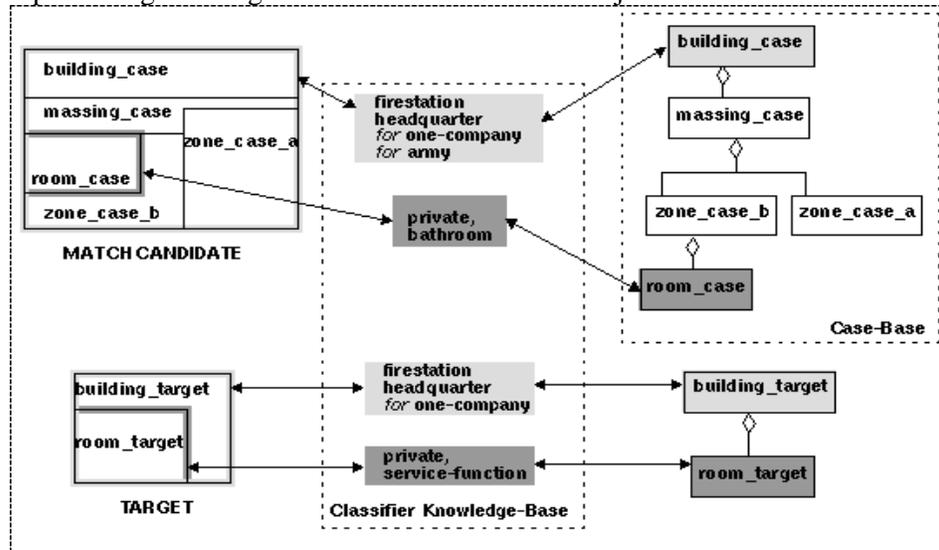


Figure 1. A retrieval example

6. Conclusion

In this study we identified the role and characteristics of a classification scheme as part of case indexing and retrieval in a CBD system. Accordingly, we outlined a recall mechanism that has a classification inference and the matching engine as the major components. The combination of the following features characterizes our approach to case indexing and retrieval:

- **Generality:** For the indexing mechanism, a case index is merely a container of objects with classifications. Whether a case is retrieved based on its problem specification, outcome, or solution is left to the individual modules. Similarly, the retrieval mechanism allows the modules to specify their own matching operations in case they need to employ domain specific reasoning. The suggested functionality will be accessed through a common interface. Consequently, the indexing and retrieval mechanisms will not be affected from the addition or removal of sub-systems as clients.
- **Hybrid approach to model a case memory:** The classification is separated from the matching inference. This enables the modules to modify their classification knowledge-base without having to modify their domain knowledge for cases.

- **Extensibility of the classification scheme:** As a follow up to the second feature, a common interface to support the functionality to add, remove or modify the classifications can be provided. This way, notions that are new to the case-base's knowledge domain can be introduced to the system.

Acknowledgment

This research is jointly sponsored by the U.S. Army Corps of Engineers Construction Research Laboratory (USA-CERL); the National Institute for Standards and Technology (NIST); and the University of Adelaide.

References

- Argan, G. C. (1963) On the typology of architecture *Architectural Design*. Dec.
- Bandini, M. (1984) Typology as a form of convention. *AA Files* **6**:73-81.
- Flemming, U. and Woodbury, R. (1995) Software environment to support early phases in building design: Overview. *Journal of Architectural Engineering* **4**:1:147-152.
- Flemming, U., Aygen, Z., Coyne, R., Snyder, J. (1996) Case-based design in a software environment that supports the early phases in building design. *Issues and Applications of Case-Based Reasoning to Design*, Maher, M. L. and Pu, P. (eds.) Lawrence Erlbaum Assoc., NJ.
- Hinrich, T. R. (1992) *Problem Solving In Open Worlds*, Lawrence Erlbaum Assoc., NJ.
- Kolodner, J. L. (1994) *Case-Based Reasoning*. Morgan Kauffman Publishers Inc., CA.
- Maher, M. L., Balachandran, M. B. and Zhang, D. M. (1995) *Case-Based Reasoning in Design*. Lawrence Erlbaum Assoc., NJ.
- Moneo, R. (1978) On typology. *Oppositions*. MIT Press. **13**:1: 22-45.
- Oxman, R. (1994) A computational model for the organization of case knowledge of a design precedent. *Design Studies* **15**:2.
- Rosenman, M. A., Gero, J. S. and Oxman, R. A. (1992) What's in a case: The use of case bases, knowledge bases and databases in design. *Proc. CAAD Futures 91*, Schmitt, G. (ed), Wiesbaden. 285-299.
- Smith, E. E. and Medin, D. L. (1981) *Categories and Concepts*. Harvard University Press, Cambridge, Mass.
- Sowa, J. F. (1984) *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, Reading, Mass.
- Tulving, E. (1972) Episodic and semantic memory. Organization of Memory, Tulving, E. and W. Donaldson (eds.) Academic Press, NY.
- Waltz, D. L., (1988) Is indexing used for retrieval? *Proc. DARPA Case-Based Reasoning Workshop*, Clear Water Beach, FL. 41-44.