The Representation of Design Cases in Building Construction

WEIYUAN WANG, DONGMEI ZHANG and DONGZHI SUN*

ABSTRACT

The integrated utilisation of design cases with different types of generalised knowledge raises particular issues on the representation and memory organisation of design cases. The complexity of the design process in building construction and the lack of design intentions in building project documents make these issues even more important in creating integrated case-based design systems in this domain. What makes up a design case, how it is acquired and represented, how specific knowledge in design cases can be efficiently used and integrated with generalised knowledge, and how to organise case memory are addressed in this paper. Based on our previous experiences in developing case-based design systems and exploring heterogeneous knowledge memory, two schemes for design case representation and case memory organisation are presented. They are discussed in terms of content of cases, method of acquisition, framework of representation, and organisation of case memory.

Key Words

case-based reasoning; integrated design system; case representation; case acquisition; case memory organisation

INTRODUCTION

Case-based design appeals intuitively because much of the design knowledge comes through the experience of multiple, individual design situations. For many domains where design knowledge is difficult to acquire and may not be objectively applicable, the case-based paradigm provides a model for the acquisition, organisation and reuse of specific design knowledge. Therefore, there is a large and growing interest in the use of the CBR approach for building knowledge-based systems that aids in the process of design.

Recently many case-based design models have used the idea of integrated systems. Related work, including (Sycara and Navinchandra, 1988), (Goel and Chandraskaran, 1989) and (Hinrichs and Kolodner, 1991), is aimed at employing specific design experiences to solve certain classes of design problems by incorporating generalised or compiled design knowledge. These integrated systems of case-based designs are developed on the basis of the

Construction Informatics Digital Library http://itc.scix.net/
paper w78-1993-239.content

Design Computing Unit, Department of Architectural and Design Science, University of Sydney, NSW 2006 Australia.

following reasons. First, certain generalised domain knowledge compensates the incompleteness of specific design knowledge in design cases. Because of the incomplete design knowledge in design drawings, documents and interviews with designers, it may not be realistic to acquire design cases covering the wide range of knowledge such as design rational, design history, reasoning steps, or other procedural knowledge from real-world design projects. Second, the cognitive activity of designers requires the combined utilisation of generalised and specific design knowledge. Facing a new design problem, experienced designers do not design strictly by abstract design principles, nor do they exhaustively search a space of previous design cases. Third, in many domains, a certain amount of design knowledge in the form of general formalism is already available prior to the development of case-based designs. Case-based systems should make use of whatever knowledge is available.

Integrated case-based design systems raise the issue of integrating different types of knowledge and methods of reasoning. The types of available design knowledge and reasoning methods vary from context to context, thus leading to different ways to design case representation, knowledge memory organisation and reasoning process combination.

This paper focuses on the case representation and knowledge memory organisation in integrated case-based design systems. The issues that are addressed are the result of developing case-based design systems, and design cases are acquired from the domain of structural design of buildings. First, we discuss design case representation from two aspects: what is in a design case and what kind of representation is used. Two design representation schemes are presented, in which design case knowledge has been combined with generalised decomposition knowledge or design prototype knowledge. Then we present two approaches to organising memory containing design cases and generalised knowledge. For each of them, the relevant representation framework of design cases, indexing strategy, memory organisation are described. Finally, a summary is drawn.

DESIGN CASE REPRESENTATION

To build a case-based design, a repository of specific design situations is required. Two elementary tasks associated with design case representation are the content and the format of design cases; the former is to identify what is in a design case in order to reason about its applicability in a different design context, the latter is to determine an appropriate structure of encoding specific design knowledge in a design case. The content and format of a design case determine how a case can be represented so as to maximise the usefulness of the case's information to a design actor.

What Is In a Design Case

When storing a design situation as a case, the content of the case must be considered since the retrieval of design cases, the selection of the most relevant design case and the transformation of a design case all depend on the content of a design case.

For design, what is stored in a case reflects the characteristics of design knowledge, as design case retrieval and transformation are based not just on surface features such as the description of a design solution, but also on the causal relations between function, behaviour, performance, etc. Design knowledge associated with a design case needs to be represented at several levels ranging from a topological description of the component object to linguistic specifications. This increases the complexity of the representation and organisation of cases.

In the development of an integrated case-based design system, the distribution of knowledge in design cases or a generalised formalism must be determined. In addition, certain correspondences need to be established between design cases and generalised design knowledge in order to complementarily employ both types of knowledge. Due to the various representation schemes of generalised knowledge, the ways to structure and represent design cases vary. In this paper, a design case represents a single design situation in the context of structural design of buildings. The knowledge of a design case contains declarative descriptions about a design problem and its design solution in terms of attribute-value pairs whereas the generalised design knowledge provides causal relationships to support the decision making in the form of decomposition hierarchies or design prototypes. In the following, we describe two approaches to representing design cases in the integrated systems of CBR with decomposition or design prototypes.

Two Structural Schemes to Represent Design Cases

We suggest that the representation of design cases in building construction requires a formalism that at least includes descriptions of structure, function and behaviour so that reasoning about the transformation from an old solution to a new design problem is possible. It may be explicit or implicit. In our systems, a design case simply consists of a set of attribute-value pairs describing the design problem specifications and design solution descriptions or function, behaviour and structure properties of a building. There are no explicit descriptions about the causal links between attributes in various categories. The reasons are that such information has been represented as generalised knowledge in decomposition hierarchies or design prototypes and that it is impractical to acquire such information for each, which requires great effort but just for a redundant representation of information.

In our projects, design cases are collected with the cooperation of Acer Wargon Chapman Associates, an engineering consulting firm in Sydney. We were given access to the drawings for the structural design of their building projects. For each building project, there is a set of drawings produced for documentation purposes, primarily for the purpose of communicating the information needed to construct the building. For example, the drawings show how much reinforcement each beam has, but none or very little design information such as lateral load resistance or system design information resides on the drawings. To acquire design cases, we augmented the information found on the drawings with interviews with the engineers involved the project. We chose not to put drawings in case memory but to capture the essential design information.

Case Represented as a Decomposition Hierarchy

In the case-based design model CADSYN (Maher and Zhang, 1991) combined with a general decomposition approach, a case hierarchy is a representation schema for a case. Each design case is stored declaratively in a hierarchical structure where each node in the hierarchy comprised a label and a set of attribute-value pairs. A design case is decomposed into a supercase part and a number of subcases, as illustrated in Figure 1. This case representation provides the process model with a means to use subcases independently of the entire case. A supercase provides a overall design episode context and general description. Each subcase describes the local context and the solution of a design subsystem. Subcases are directly indexed individually along with links that can be used to construct the whole case.

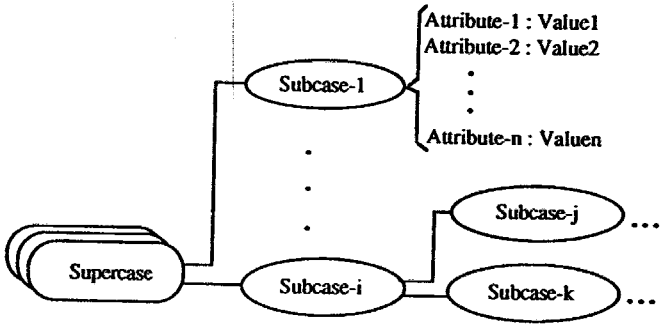

Figure 1. Structure of Cases

As both specific design cases and generalised decomposition knowledge are incorporated to derive a new design solution, a correspondence between

them is established in case memory. This correspondence can be expressed as follows: the design description of a particular design case is represented as a set of subsystems from the decomposition knowledge, where each subcase matches a generalised subsystem. This ensures that subsystems and their attributes can be recognised during case transformation for constraint checking and repair.

In structural design, the content of a design case is constructed in three layers: problem specifications as a global context; a grid representation for each function of building as the geometric context; and structural systems as a design solution for each grid level. The problem specifications of a design case includes general architectural specifications and loading information, such as the number of stories, the intended use of the building, etc. The grid representation contains bay numbers and sizes in the two principle orthogonal directions, and other functional and geometric information. A design case of a particular building is illustrated in Figure 2. This office building provides four functional spaces: parking, retail, office, and service-core. GEN-CASE shows a overall problem description of the building. The local design context for the office space is shown in the GRID3, and the structural design description associated with GRID3 is illustrated in terms of lateral-systems, gravity-system, transfer, etc. The attributes in the structural design description are categorised as requirements (req) and design decisions (des).

Case Represented with Design Prototype Framework

With this approach, all attribute-value pairs of a design case are categorised into classes of function, behaviour and structure. The purpose of this representation is to explore the integrated application of design cases with design prototypes (Rosenman *et al*, 1991; Sun and Gero, 1991; Wang and Gero, 1991). A design prototype is generally a conceptual schema for representing generalised or compiled knowledge for a class of similar designs (Gero, 1990). It groups the relevant attributes of a generic design into the categories of function, behaviour and structure, as well as context, defines each attribute in terms of unit, type, value range and default value, and represents the dependencies between attributes as relational, qualitative and quantitative knowledge. For example, if the type of window is dependent on the use of room, the relation will be represented as window-type <- room-use; if the cost per unit floor area increases as the height of building increases, the relation will be represented as cost-per-unit-floor-area <- building-height^{+}. The role of design prototypes in knowledge-based design systems was demonstrated with an example of building envelope design in (Tham, 1991).

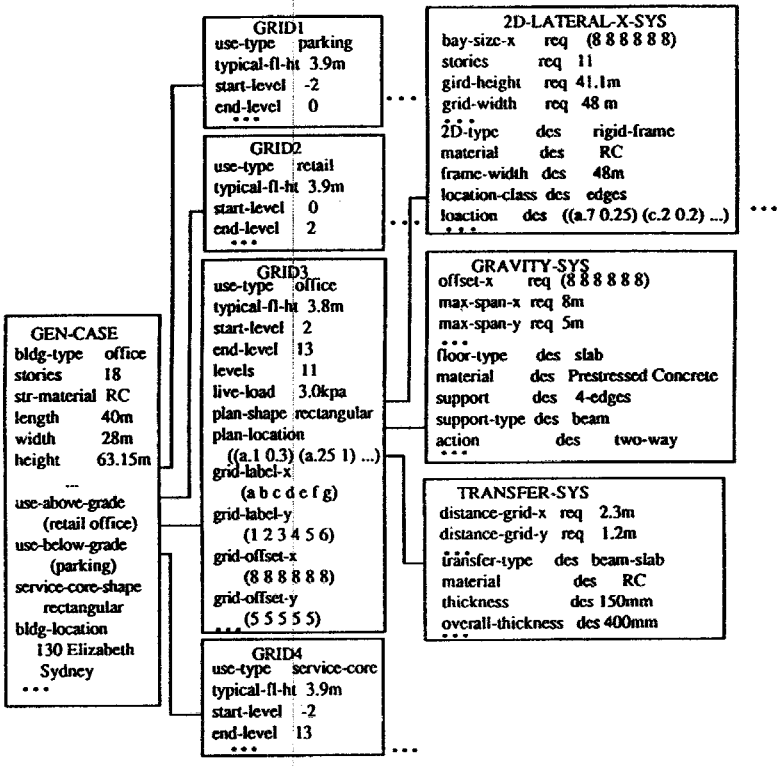

Figure 2. A Partial Description of a Design Case for an Office Building

A design case has been regarded as an instantiation of a design prototype with this approach. Thus, case representation matches the representation framework of design prototypes as showed in Figure 3. This representation has many advantages. Design prototypes are efficient in generating and evaluating design alternatives because of its deeper knowledge in the dependency network represented by various relations, while design cases are powerful in providing a good set of initial values for a new instantiation of a design prototype. Therefore, a past design case may provide a good starting point for a new design while a design prototype may guide the process of design case adaptation or transformation with its rich design knowledge. Without design cases, a design initialisation may need to be done by a human designer or by a long chain of reasoning steps. This approach has represented specific knowledge in design cases and compiled knowledge in design prototypes and integrated them in a single design system to complement the strengths of the individuals.

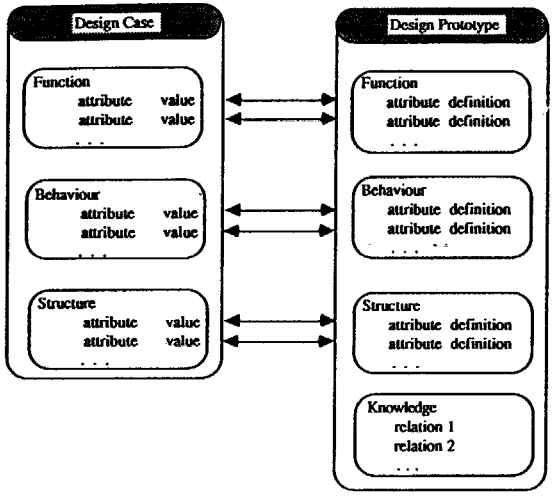

Figure 3. Correspondence Between Representations of Design Cases and Design Prototypes

MEMORY ORGANISATION OF DESIGN CASES

Except for very small case bases, previous design cases must be organised so that the search of past design cases can be more selective and retrieval efficiency can be achieved. The organisation of case memory should be able to support a wide range of design tasks and allow a retrieval algorithm to retrieve design cases with different types of probes and return the most relevant design cases by searching only the related part of memory. To achieve these goals, we usually need to consider the following problems: for a given domain what type of features in design cases may be selected as indices, how to search the case memory during retrieval, how to update the memory when a design case is incorporated into memory. In designing integrated systems, we may also need to consider the links and the consistency between case memory and generalised knowledge so that the maximum benefit of the integration will be achieved.

Design cases may be indexed by different types of features such as function, behaviour, structure, context, design problem specifications, or even design knowledge. A case memory may be searched with a fixed order of priorities for indexing features or by a method of concept refinement (Reich and Fenves, 1991). The case memory may be updated incrementally or non-incrementally by a program or a knowledge engineer.

Two approaches of memory organisation for design cases presented in this paper are based on the Schank's idea of MOP, memory organisation packet (Schank, 1982). In both structures, case memory is dynamically organised as a concept hierarchy with more general concepts at higher levels and more

specific concepts at lower levels. There are differences in these two approaches. One difference is in the method of index selection, another is in the amount of information generalised in the nodes of concept hierarchy. We will discuss these separately in the rest of this section.

CADSYN's Approach

The case memory in CADSYN is a representation of previous design situations and indexing information. Being a manageable structure, case memory is organised into two components: case hierarchies and case indexing representation. In this schema, the problem specifications in the cases are fixed, that is, the design problem description of each case is specified by a set of similar attributes and their values.

The case indexing representation contains a problem abstraction hierarchy and a solution list. The problem hierarchy is a set of generalisation nodes based on previous design problem requirements and is developed using a conceptual classification approach. In the hierarchy, each level has its pointer to record super-nodes of nodes in this level. In the problem hierarchy, design problem requirements are used as indices of cases to retrieve relevant design solution descriptions. The problem hierarchy is illustrated in Figure 4. The pointer at level 2 inherits all cases of the nodes above level 2. For example, if node-22 has relevant specification, then cases 4, 7, 9, and 2 are retrieved.

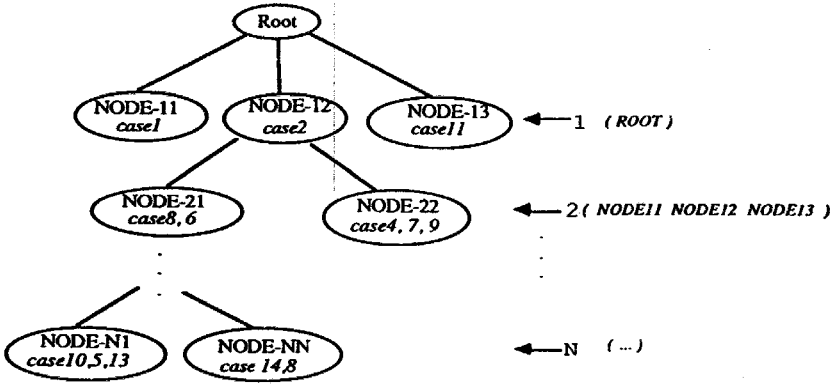

Figure 4. The Structure of the Problem Hierarchy

The solution list consists of all systems and all subsystems in the decomposition knowledge, and serves to index subsystems of cases. Each system or subsystem indexing indicates relevant cases which contain a specific

corresponding subcase, as illustrated in Figure 5. The solution list is used to find relevant cases which have a given subsystem solution whenever the case-based reasoner is invoked at any point of decomposition.

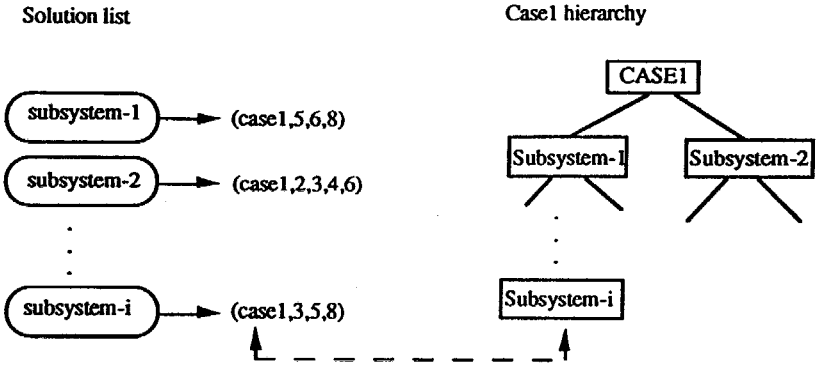

Figure 5. The Solution List

Design Prototype's Approach

With design prototype's approach, a case memory consists of several conceptual schema hierarchies, each of which is created from different levels of abstractions with dynamic memory organisation techniques. Three basic schema hierarchies are function schema hierarchy, behaviour schema hierarchy, and structure schema hierarchy. The function schema hierarchy is created and indexed according to the functional properties of a design and the intermediate nodes in the hierarchy are defined as F-schemas. Part of the representation of a F-schema is showed in Figure 6. Although the creation of the schema is dependent on the amount of similarities in functional features among cases, the model also generalises information on behavioural and structural attributes shared by cases under each schema. This is an critical distinction between this model and other memory organisations or concept formations (Lebowitz, 1987). The behaviour schema is defined as B-schema and the structure schema is defined as S-schema. The B-schema hierarchy and the S-schema hierarchy are created with the same method but with different categories of attributes as indexing features.

The primary advantage of this organisation is that the case memory can be searched with functional, behavioural or structural descriptions about a given design as well as any combination of them. This may allow a design agent to reason at various levels of design abstractions. Since the value range and the default value of an attribute in a conceptual schema is dynamically

generalised from a set of similar design cases, usually the value range is more narrow than that in the related design prototype and the default value has less deviation. We may consider the range of attribute provided in a design prototype as a feasible value range which can be used in a wide scope, and that in a generalised schema as a recommended value range for particular design situations. The latter is more useful in certain design situations. During a case adaptation or transformation, a more reasonable alternative value may be selected from the range of an attribute in a schema and the selection will be guided by the relational knowledge in design prototypes.

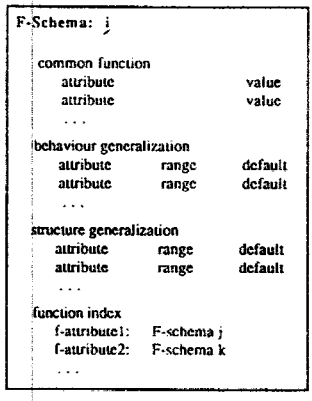

Figure 6. Part of a F-schema Representation

Conceptual hierarchies created with other categories of features such as design context or various relational knowledge may be also useful. The method of creating generic cases and indexing design cases with one type of features and generalising information with all types of features may be proven more useful in case memory organisation or design concept formation in future.

SUMMARY

We have discussed the acquisition of design cases from real-world projects and presented two schemes to case representation and case memory organisation which resulted from our previous experiences in developing integrated case-based design systems. CADSYN's approach combines design cases with decomposition hierarchy, represents a design case in design problem descriptions and design solution descriptions, and organises case memory into the problem hierarchy and the solution list. For more details about the application of design cases with decomposition, one may refer to (Zhang and Maher, 1991). The design prototype approach integrates design cases with conceptual schema design prototype, represents design cases in

terms of function, behaviour and structure, and organises case memory into multiple conceptual schema hierarchies so that cases can be searched from different dimensions. More details about connection between design cases and design prototypes and the process of memory organisation of heterogeneous knowledge are discussed in (Wang and Gero, 1993).

Acknowledgments

The work described in this article is based on PhD research carried out under the supervision of Prof. Mary Lou Maher and Prof. John S Gero. We would like to thank Acer Wargon Chapman Partners for providing access to their structural design reports and drawings and for participating in discussion.

References

Gero, J S (1990), Design prototypes: a knowledge representation schema for design. AI Magazine, Vol 11, No 4, pp 27 - 36.

Goel, A and Chandrasekaran, B (1989), Use of device models in adaptation of design cases. In Proceedings of the DRAPA Workshop on Case-Based Reasoning. Pensacola Beach, Florida. pp 100-109.

Hinrichs, T R and Kolodner, J L (1991), The roles of adaptation in case-based design. In Proceedings of the DARPA Workshop on Case-Based Reasoning. Washington, DC pp 121-132.

Lebowitz, M (1987), Experiments with incremental concept formation: UNIMEM, Machine Learning, Vol 2, pp 103-138.

Maher, M L and Zhang, D M (1991), CADSYN: using case and decomposition knowledge for design synthesis. In Artificial Intelligence in Design (ed JS Gero). Butterworth-Heinemann, Oxford, pp 137-150.

Reich, Y and Fenves, S J (1991), The formation and use of abstract concepts in design. In Computational Approaches to Concept Formation (ed DH Fisher, MJ Passani and P Langley). Morgan Kaufmann, San Mateo, CA.

Rosenman, M A, Gero, J S and Oxman, R E (1991), What's in a case: the use of case bases, knowledge bases and databases in design. In CAAD Futures'91 (ed GN Schmitt). ETH, Zurich, pp 263-277.

Schank, R C (1982), Dynamic Memory: A Theory of Learning in Computers and People. Cambridge University, Cambridge.

Sycara, K P and Navinchandra, D (1989), Influences: a thematic abstractions for creative use of multiple cases. In Proceedings of the DARPA Workshop on Case-Based Reasoning Workshop. Morgan-Kaufman, San Mateo, CA.

Sun, D and Gero, J S (1991), Representing design experience through design cases: the use of case-based reasoning in design. In The Technology of Design (ed G Woodbury). ANZAScA, University of Adelaide, Adelaide, pp 235 -244.

Tham, K W (1991), A model of routine design using design prototypes, PhD Thesis, Department of Architectural and Design Science, University of Sydney.

Wang, W and Gero, J S (1991), A model for the organisation of design prototype and design case memories. In Artificial Intelligence in Design, Preprints of the Workshop of IJCAI-1991, Sydney (ed JS Gero and F Sudweeks).

Wang, W and Gero, J S (1993), A memory organization for heterogeneous design knowledge. IABSE Colloquium : Knowledge-Based Systems in Civil Engineering, Beijing, 12-14 May 1993 (accepted).

Zhang, D M and Maher, M L (1991), The transformation process in case-based reasoning in design. In Artificial Intelligence in Design, Preprints of the Workshop of IJCAI-1991, Sydney (ed JS Gero and F Sudweeks).

c. 1993, Management of Information Technology for Construction, K. Mathur et al (Eds), World Scientific Publishing Co., Singapore.