Looking Back to the Future

An Updated Case Base of Case-Based Design Tools for Architecture

Katharina Richter[1], Ann Heylighen[2], Dirk Donath[3]
[1,3]Bauhaus University Weimar, Germany, [2]Katholieke Universiteit Leuven, Belgium
[1,3] http://infar.architektu.uni-weimar.de [2] http://www2.asro.kuleuven.be/home/heylighen
[1] katharina.richter@archit.uni-weimar.de, [2] ann.heylighen@asro.kuleuven.be, [3] donath@archit.uni-weimar.de

In the early and mid 1990s the idea to apply CBR to the task of designing — in short Case-Based Design (CBD) — led to a considerable number of research initiatives across the world. Several promising CBD tools and prototypes were developed and enthusiastically celebrated within the research community, seemingly announcing a promising future for CAAD. However, because the predicted breakthrough failed to appear, an in-depth evaluation of six CBD tools was conducted in 2001 in search of reasons for this limited success. At first sight the situation has not changed much since then, yet a closer look reveals CBD research still to be quite active, be it sometimes disguised. This observation, combined with our belief in CBD's potential for aiding professional and student architects, motivated an expanded issue of the 2001 study. This issue determines the position of current CBD research within the CAAD domain and uncovers focal points set by CBD researchers and the tools they created. Additionally it analyses the role of emerging technologies in overcoming earlier identified drawbacks of CBD tools in architecture.

Keywords: *Case-based design; architectural design; design support systems.*

Introduction

Designers are said to predict the future with their doing. „Designers are … all `futurologists` to some extent" (Lawson, 2006, p. 112). The engagement with and the reuse of knowledge encoded in previously solved design tasks – design precedents – is widely recognized among design professionals as a powerful instrument to support this highly demanding task.

The reuse of experiential knowledge encoded in previously experienced episodes in supporting problem-solving and interpretive tasks lies at the very heart of the AI paradigm of Case-Based Reasoning (CBR) (Schank, 1982; Kolodner, 1993), which apparently reached its middle age. In the early and mid 1990s the idea to apply CBR to the task of designing—in short Case-Based Design (CBD) – led to a considerable number of research initiatives across the world. Because the predicted breakthrough failed to appear, an in-depth evaluation of six CBD tools was conducted in 2001 in search of reasons for this limited success (Heylighen et al., 2001). This study was centered around the analysis of six CBD tools in architecture: Archie-II, CADRE, FABEL, IDIOM, PRECEDENTS, and SEED-Layout.

The analysis and discussion of the findings resulted in a case base of CBD tools for architecture, which followed the model of cognition underlying CBR. This model combines three major aspects, to be taken into consideration when designing any CBD tool: knowledge structure and organization (case content, (re)presentation, memory organization), reasoning processes (reminding/retrieval, manipulation), and learning.

The cognitive model claims that knowledge resides in memory both as specific events (cases) and as generalizations, and that both types of knowledge use the same organizational structures. A majority of the tools in the 2001 study contain general knowledge, but only Archi-II organizes general and episodic knowledge in the same way.

Several reasoning processes play a role in CBD, such as remembering by retrieving relevant cases (mostly by comparing features between cases and the situation at hand) and manipulating the retrieved cases to satisfy the current situation (by structural or derivational adaptation, or by matching). The study showed that case retrieval and structural adaptation (recycling the end product) are fairly well addressed in the analyzed tools, whereas derivational adaptation (recycling the process which led to the solution) and merging multiple design solutions are hardly addressed so far.

Learning plays the key role in the cognitive model, since learning form experiences is the very essence of CBD. However, the 2001 study revealed that learning is not at all addressed in the analyzed tools and therefore closed with the invitation to "put learning at the top of the research agenda" (Heylighen and Neuckermans, 2001, p. 1121) .

An updated case base of CBD tools in architecture

At first sight, CBD research seems to be somewhat outdated, especially since the promised breakthrough did not materialize. Yet a closer look reveals CBD research actually still to be quite active, be it sometimes disguised. This observation, combined with our belief in CBD's potential for aiding professional and student architects, motivated an expanded issue of the 2001 study. As in the first study, we used CBR's cognitive model as a framework for an in-depth analysis of several CBD tools, widely published between 2001 and today. Six of these were selected to become part of our updated case base of CBD tools based on their potential to illustrate directions in current and future CBD research. After briefly introducing its objectives, each case study describes and briefly discusses how the tool implements knowledge structure and memory organization, reasoning processes and learning.

Case 7: 1 SL_CB – A Prototype for Integrating Housing Design and CBR

SL_CB applies the CBR paradigm to support user-controlled automatic housing layout (floor plan) generation (Lee, 2002). Starting from studies of the US-American housing market and design scenarios for single-family houses, this prototype is rooted in SEED and makes use of the SEED-Layout and SEED-Database modules. The prototype was developed as part of Ph.D. research at the School of Architecture and Institute for Complex Engineered System at Carnegie Mellon University.

Description. Cases in SL_CB are represented by abstracted floor plan layouts of housing projects. The tool uses the same structure of case representation and knowledge storage as SEED-Layout. The object database adopts an object-oriented approach to describe important object classes used in the tool, such as Design Units (as representatives for rooms described by dimension and location) and Functional Units (a container for the requirements of Design Units). A collection of Functional Units constitutes the problem specification, called a Layout Problem.

SL_CB aims to provide flexible and multiple ways for case classification in the case base. This is accomplished by separating precedent knowledge of concrete instances from generalized knowledge expressing underlying concepts in a classification database (CKB). A third database called case base (CB) is used to attach additional information, particularly automatically recognized component-based case features, and match operators to cases, which can additionally support retrieval.

SL_CB supports case retrieval and manipulation. Cases can be retrieved by name or index, which is a combination of classification and components. Retrieved Layout Problems serve as starting point to further refine the search procedure. As in SEED, the developers thereby acknowledged the fact that design problems develop with the design process. Adaptation in SL_CB is completely based on the functionality in SEED-Layout.

Discussion. The CBD model claims that general and instance knowledge are equally important in understanding and problem solving. SL_CB adopted a hybrid approach which chimes with this model: integrating generalized knowledge both as classifications of instance knowledge in terms of applied architectural concepts, and as adaptation knowledge in the representation of the Functional Units (e.g. as dimensional requirements or required adjacencies).

The CBD model also suggests that reminding is an unintentional process occurring during the processing of new information (Schank, 1982; Kolodner, 1993). SL_CB requires very precise input of design requirements as starting point for retrieval, based on a detailed understanding of the design problem by the user, thereby leaving little room for unintentional, but potentially fruitful associative reminding of cases stored in the tool (parallels can be drawn with case 10 further on).

Case 8: TRACE

TRACE, a prototype developed as part of Ph.D. research at Carnegie Melon's School of Architecture, aims to offer computer support for design composition or 2D form generation (Mubarak, 2004). It uses the concept of derivational analogy (Carbonell, 1986) to manipulate generative paths that led to a previous solution, in order to solve the current design problem. This concept presumes that the generative path includes all aspects of the reasoning process used in deriving the earlier solution, including methods, alternatives, decisions, justification.

Description. TRACE represents cases in the form of architectural solutions and the path which led to them in an abstract way (floor plans, form and function diagrams, attributes, solution trace (Sol-Trace)). The Sol-Trace is a sequence of operations and elements, stepwise introduced to reach the final design solution. Indexing as well as retrieval and matching mechanisms take advantage of the tool's object-oriented approach to represent cases and Sol-Traces.

TRACE supports two reasoning processes: reminding (retrieval) and manipulation (mainly replay of the derivational paths). Retrieval can be user-driven, when searching for particular case characteristics, or automatic.

Discussion. TRACE considers the process leading to the design solution – rather than the solution itself – as the core of the case description, and stores this process for automatic manipulation and generation of new solutions. This is another notable exception to the observation that CBD researchers tend to define (parts of) the final design product as the major source of experiential knowledge in architecture (Heylighen et al., 2004). Unfortunately, TRACE only models how the design solution may have been generated without inferring the design intend or rational behind design decisions, which seems to contradict the original concept of derivational analogy.

Just like in CADRE (in the 2001 study), case representation in TRACE relies on a very abstract (simple geometrical shapes and transformations) and computer readable format, so as to accomplish automatic case manipulation. As a result, essential characteristics of cases and their integration are lost.

A major obstacle of TRACE is identified by the author himself: the complex definition, identification and encoding of design strategies makes case acquisition and maintenance highly difficult.

It remains to be seen whether trying to demonstrate the power of derivational analogy by modeling (only) compositional steps, and in a very abstract way at that, will prove to be a very fruitful approach.

Case 9: CaseBook

Although not explicitly developed as a CBD tool, CaseBook (Inanc, 2000) was selected for this study because it shares many aspects with other CBD tools and (be it on purpose or not) starts from the premises of the cognitive model. CaseBook stores and retrieves floor plans of housing projects with the focus on spatial-functional relations. It is closely related to architectural archives and information retrieval systems. The developer mainly focused on supplying a flexible classification schema by means of lazy, post-coordinated retrieval strategies. The prototype was developed at the TU Delft Design Knowledge System Research Center.

Description. Cases in CaseBook are floor plans of residential units, converted into a structured graphic format by the Graphical Case-Editor module. Case representations are completed with structured non-graphical information (on the site, the architect etc.) and non-structured information in the form of multimedia data. An automatic feature extraction module (AFE) extracts topological and quantitative information from the layouts and provides this information for the post-coordinated retrieval mechanism.

CaseBook limits implemented reasoning processes to reminding, which comes in a higher degree of automation than in most other analyzed tools. The lazy classification approach is based on demand-driven computation. Classification depends on similarity measurements according to user-defined criteria during retrieval, based on the information provided by the AFE module. Users can select features as search criteria and adjust their relative importance via the Criteria-Editor. Retrieval is supported through queries by example, recognizing the importance of graphical representation for designers. Users are able to enter an example floor layout in the Case-Editor as input for the search process.

Discussion. Concentrating on the improvement of search mechanisms, the developers did not address manipulation or learning. Since the tool is actually highly interactive, it would be easy to gain valuable data from users' search behavior to let the tool dynamically learn from its users.

Integrating graphic-based query avoids the identified keyword barrier, and recognizes the importance of visual information for designers, but puts a heavy burden upon the users: they need a fairly precise idea of the intended spatial-functional relations to get fruitful results for the further design process. The tool provides means for any kind of associative search, which was recognized by other developers as an important search strategy in any CBD tool.

Case 10: MONEO

MONEO – Latin for to warn, admonish, remind, advise, or instruct – aims at supporting architects by reducing the usually extensive amount of time needed to search for similar projects in the pre-design phase (Taha, 2006). The prototype was developed as part of Ph.D. research at the Faculty of Engineering, Alexandria University.

Description. In MONEO cases are formalized (and computer-readable) descriptions of floor plans of one story housing units extended by factual data (construction time and location, architect), as well as visual material (floor plans, sections, photographs). A hierarchical object-oriented model represents cases by a collection of objects, in turn represented as attribute-value pairs.

Given the importance of visual information in design, the idea was to support reminding in a visual way, via a three-step search procedure involving three modules. In the first step the user enters a request concerning each functional unit of a residential unit via a multi-tab form. Based on this, the Bubble-Diagram-Generator generates a bubble-diagram in the second step. The user may graphically edit the number of rooms, connections, size, and location. At the end of this phase, the user defines the number of cases to be presented and the level of similarity, which triggers an automatic request. The request is transformed from a user-friendly graphical format into a computer readable representation (attribute-value pairs). Module 3 is in charge of the actual retrieval mechanism and delivers the result in image and text. Re-transformed in a bubble diagram this result can serve as the starting point for another search loop.

Manipulation and learning are not addressed in this prototype. However, the developers express their conviction that MONEO supports an easy way of incorporating new cases and as such provides means for the dynamic growth of the case base.

Discussion. MONEO's major asset is its attempt to support fast and easy retrieval of similar cases by means of graphical representation of floor plans in the form of bubble diagrams. Since architects know, think and work in a visual way (Cross, 1982), supporting visual search is of utmost importance, indeed. However, in order to successfully apply MONEO in the early phases of housing design, the user must have a fairly good understanding of the design problem at hand, and be able to specify number, location and dimensions of rooms already at the very start of the design process. This does not only assume that architects think in terms of 'rooms', but more importantly flies in the face of the notoriously ill-defined nature of design problems, which can only be understood gradually as the design process proceeds.

Case 11: Case Base for Architecture - CBA

Case Base for Architecture (Lin et al., 2003) is representative for the many tools that can be categorized as Case-Based Learning or Case-Based Teaching tools. Other examples are EDAT (Akin, 2002), WebPad (Oxman, 2004), CaseBox (Chen et al., 2005), eCAADe (Lee et al., 2006) and DNA (Tuncer et al., 2006). In designing CBA the developers mainly focused on providing a smart query interface to support associative search, retrieval and representation of semantic relations among cases.

Description. Cases in CBA are built or designed office buildings and housing projects, represented by general info (attribute-value pairs), analytical info (graphics and images, encoded through annotations for computability) and textual information.

Like eCAADe, CBA uses data mining techniques to enhance case search and retrieval, be it in a slightly different way. eCAADe adopts these techniques and concepts from recommender systems to analyze user behavior patterns, in order to adapt case retrieval to different user needs. In CBA, data mining techniques extract a lexicon of terms from the available texts about cases already stored in the case base. Subsequently domain knowledge is applied to filter out useful keywords regarding underlying architectural concepts. The developers used the Function, Behavior, Structure model (Maher et al., 1995) to build up a context ontology for classifying the extracted keywords. Based on statistical co-occurrence of the context of these keywords, their semantic relation is calculated automatically. The keywords are now used as inverted-indexes, the determined semantic relations among them allow to establish case relations and similarity.

Reasoning processes supported in CBA are concentrated on enhanced search processes (reminding). Recognizing the importance of support for associative search in CBD tools, the developers introduced search based on semantic relations between keywords which represent meaningful architectural concepts.

An advanced version of the prototype uses concept mapping to represent concept knowledge and semantic relations between keywords as a graph (network of concepts) to support associative search in a graphical way (Lin et al., 2005).

Discussion. Concerning integration of general and episodic knowledge in CBD tools, Web-Pad (Oxman, 2004) builds on the promising approach adopted by its predecessor PRECEDENTS (in the 2001 study). Generalizations in the form of high-level architectural concepts underlying the keywords used for indexing serve to organize specific events.

Also worth mentioning is that CBA's developers acknowledge the importance of real experiential knowledge in CBD. Besides general and analytical knowledge, a third level of knowledge, called "recommendation", contains case creator und user feedback for each single case and incorporates highly valuable experiences of professionals. Cases are thus more than just representations of the finished product, the solution, in that they incorporate information on the outcome of the design as intended by the cognitive model. Unfortunately developers did not describe in detail how this valuable information is gained.

Case 12: DYNAMO

DYNAMO, which stands for Dynamic Architectural Memory Online, aims to stimulate and support (student) architects in sharing ideas, knowledge and insights embedded in specific building projects (Heylighen, 2000; Heylighen et al. 2004). It is conceived as an (inter)active workhouse rather than a passive warehouse, emphasizing the idea of dynamic learning. The first version was developed as part of PhD research at the K.U.Leuven, Department of Architecture.

Description. Cases in DYNAMO are built or unrealized projects. They are represented by various media (sketches, drawings, digital models, pictures, text) and labeled by various features (architect, location, aspects of form and space, function, construction and context). These metadata serve as filter criteria during retrieval and as links to projects with analogous characteristics.

Learning is supported in various ways. First of all, users are considered as active contributors to the case base; they can share experiences with other users by supplementing or re-indexing projects, or submitting new ones. In addition, DYNAMO uses collaborative filtering to extract otherwise hidden data from user interactions with the case base (log files) to determine and continually update inter-project relationships. These constitute an association matrix that guides search and retrieval by providing a list of cases that are most strongly associated and tailor-made recommendations based on user preferences (Heylighen et al., 2006). More recently, DYNAMO's case base is being networked with other repositories so as to enhance its contents by cases and information outside the case base (Neuckermans et al., 2007).

Discussion. DYNAMO tries to incorporate CBD's cognitive model quite literally and to extrapolate it beyond the individual. Since learning from experiences is the very essence of CBD, and is hardly supported by the tools analyzed in the first study, DYNAMO's developers decided to focus on dynamic learning as key concept and to explore several learning strategies.

Exceptional are continually performed experiments to evaluate chosen conceptual and implementational steps. Their results support an ongoing fine-tuning process, which distinguishes DYNAMO most of the other tools, and enable the tool to learn from its experiences. Future experiments will show how the implementation of data mining techniques and the integration of DYNAMO within a wider network of repositories will influence the usability of the system and its performance in supporting (student) designers.

Discussion and future work

First of all, the study reported on in this paper aimed at determining the position of current CBD research within the field of CAAD. The most obvious finding studying this respect is that a major shift took place from applying CBR to create Design Automation Systems towards the application of CBR in Design Aiding Systems, especially for educational purposes. Whereas in the former study four out of six tools aimed at some form of automated case manipulation, in the current study only two out of six (SL_CB and TRACE) adopt this approach. To our knowledge, TRACE is the first CBD tool in architecture to use derivational analogy.

One way of learning suggested by CBD's cognitive model is through acquiring new experiences (cases). Our study confirms the findings from the 2001 study that the acquisition bottleneck is still not sufficiently addressed. Some developers try to make the proposed case acquisition process as simple as possible (SL_CB, MONEO), but the majority of the systems do not even address this issue at all. An exception to this rule is DYNAMO, which directly translated the first study's findings in system requirements.

Secondly, the study aimed to identify the role of emerging concepts and technologies in overcoming earlier identified drawbacks in CBD systems. When starting to analyze current CBD systems, we hypothesized that concepts and technologies such as Fuzzy Logic (to support case retrieval without perfect matches), Neural Networks (to support classification and pattern matching), or Genetic Algorithms (to support learning weights of case features), would play a major role in the schedule of CBD developers as they do in other domains (Pal et al., 2001). Yet, our study did not notice any usage of the aforementioned techniques. Interestingly, however, it did notice the growing importance of data mining techniques to extract meaningful data for memory organization, reminding and learning.

On the one hand, data mining techniques serve to support reminding. CBA, for instance, supports associative search by using extracted key concepts from textual case material to organize cases in a semantic network. eCAADe uses concepts borrowed from recommender systems based on these techniques for adaptive search and retrieval based on computed user preferences. DYNAMO analyzes user behavior to determine cases' relatedness and thereby provide means for associative search.

In addition, data mining techniques serve to support learning in the broadest sense. Worth mentioning are efforts to use automatic feature extraction (CaseBook) and data mining techniques (CBA) for extracting knowledge implicitly available in descriptive case material, to provide means for collecting indexing vocabulary. In this way, tools learn not only from their users, but also from content available in the case base.

Finally, the study's detailed look into the recent past of CBD in architecture enables us to act like futurologists and take a look into its future. Therefore we conclude with directions for future research, taking into consideration identified drawbacks and shortcomings of current CBD tools.

When applying CBD to architectural design, a major threshold is and remains the acquisition bottleneck. One possible approach to overcome this threshold is to further integrate CBD in architects' design environment, c.q. CAAD systems (Heylighen, 2000). Geometrical and topological information could easily be extracted from existing CAAD models by the use of formats such as Industry Foundation Classes (IFC) and their underlying standardized information structure. Another option is to network a CBD tool with case bases of other tools, as DYNAMO is currently exploring (Neuckermans et al. 2007).

As observed and criticized earlier, and confirmed by this study, cases in case bases represent only the finished design product, the architectural project or parts of it in different levels of abstraction. A major challenge remains the development of mechanisms to integrate actual experiential knowledge. Also here data mining techniques may play an important role.

References

Akin, Ö.: 2002, Case Based Instruction Strategies in Architecture, Design Studies, 23(4), pp. 363-435.

Carbonell, J. G.: 1986, Derivational analogy; A theory of reconstructive problem solving and expertise acquisition, in Machine Learning - An Artificial Intelligence Approach, R.S. Michalski, J.G. Carbonell, T.M. Mitchell, Morgan Kaufmann, II, pp. 371-392.

Chen, H.-S., Tseng, L.-T., et al.: 2005, CaseBox: A Tool for Case-Based Learning, in eCAADe Conference Proceedings, pp. 77-81.

Cross, N.: 1982, Designerly Ways of Knowing, Design Studies, 3, pp. 221-227.

Heylighen, A. and Neuckermans, H.: 2001, A case base of Case-Based Design tools for architecture, Compter-Aided Design, 33, pp. 1111-1122.

Heylighen, A., Casaer, M., et al.: 2006, UnaWare: supporting tacit design knowledge exchange, Int. J. Web Based Communities, 2(1), pp. 31-44.

Heylighen, A., Martin, W. M., et al.: 2004, From repository to resource. Exchanging stories of and for architectural practice, The Journal of Design Research. 01.

Heylighen, A.: 2000, In case of architectural design - Critique and praise of Case-Based Design in architecture, Faculteit Toegepaste Wetenschappen, Department ASRO, Leuven, België, K.U. Leuven: 232.

Inanc, B. S.: 2000, CASEBOOK. An information retrieval system for housing floor plans, in CAADRIA 2000, Proceedings of the Fifth Conference on Computer Aided Architectural Design Research in Asia, 18-19 May 2000 (eds.) Singapore, pp. 389-398.

Kolodner, J. L.: 1993, Case-Based Reasoning, San Mateo, Morgan Kaufman Publishers, Inc.

Lawson, B.: 2006, How Designers Think, Oxford, Architectural Press.

Lee, J.-H. and Pan, S.-F.: 2006, eCAADe: An Educational Commendation Mechanism for the Adaptive Semantic Web to Use in the Architectural Design Environment, in eCAADe 24, Communicating Space(s), Proceedings of the 24th Conference on Education in Computer Aided Architectural Design in Europe, Vassilis Bourdakis, Dimitris Charitos (eds.) Volos, Greece, University of Thessaly, pp. 244-251.

Lee, J.-H.: 2002, Integrating Housing Design and Case-Based Reasoning, School of Architecture and Institute of Complex Engineered Systems (ICES), Pittsburgh, Carnegie Mellon University: p. 133.

Lin, C.-Y. and Chiu, M.-L. (2005). 'The Concept Space Explorer'. 11th International CAAD Futures Conference, Technische Universität Wien, Östereich, Östereichischer Kunst- und Kulturverlag.

Lin, C.-Y. and Chiu, M.-L.: 2003, Smart Semantic Query of Design Information in a Case Library, in Design Design: Research and Practice, Proceedings of the CAAD Futures 2003, pp. 125-135.

Maher, M. L., Balachandran, B. M., et al.: 1995, Case-based Reasoning in Design, Mahwah, Lawrence Erlbaum Associates.

Mubarak, K.: 2004, Case based reasoning for design composition in architecture, Pittsburgh, Carnegie Melon University.

Neuckermans, H., Wolpers, M., et al.: 2007, Data and metadata in architectural repositories, in Digitization and Globalization, CAADRIA 2007 Proceedings, CAADRIA, Gang, Y. et al. (eds.) Nanjing (China), pp. 489-497.

Oxman, R.: 2004, Think-maps: teaching design thinking in design education, Design Studies, 25(1), pp. 63-91.

Pal, S. K., Dillon, T. S., et al., Eds: 2001, Soft Computing in Case Based Reasoning, London, Springer.

Schank, R. C.: 1982, Dynamic Memory – A Theory of Reminding and Learning in Computers and People, Cambridge, Mass., Cambridge University Press.

Taha, D.: 2006, A Case Based Approach to Computer Aided Architectural Design. MONEO: An Architectural Assistant System, Graduate School, Faculty of Engineering, Alexandria, Greece, Alexandria University.

Tuncer, B. and Sariyildiz, S.: 2006, Design Analysis Network - An educational environment for architectural analysis, in eCAADe 24, Communicating Space(s), Proceedings of the 24th Conference on Education in Computer Aided Architectural Design in Europe, Vassilis Bourdakis, Dimitris Charitos (eds.) Volos, Greece, University of Thessaly, pp. 206-214.