INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 95
PRAHA, AUGUST 22-24, 1995

INCORPORATING EXPERIENCE IN THE METHODOLOGICAL DESIGN PROCESS

Mehmet H. Göker, M.S.E., M.Sc. Prof.Dr.-Ing. H. Birkhofer

ABSTRACT:
Experiments were conducted to improve our understanding concerning the experience-based problem solving approach human beings prefer when they are solving routine problems. The results of these experiments and the methods and structure provided by Engineering Design Methodology were combined to implement a Conceptual Design Assistant (CoDA). The system is based on the Case-Based Reasoning approach and is able to support the design engineer with experience about methods and objects.

ZUSAMMENFASSUNG:

Um das erfahrungsbasierte Vorgehen, das Menschen beim Lösen von Routineaufgaben bevorzugen, besser zu verstehen, wurden Untersuchungen durchgeführt. Aufbauend auf den Ergebnissen dieser Untersuchungen und den Strukturen und Methoden der Konstruktionsmethodik wurde ein Konstruktions-Assistent für die Konzeptphase (CoDA) entwickelt. Das System basiert auf dem Ansatz des Fallbasierten Schließens und ist in der Lage den Konstrukteur mit Methodik und Objektwissen zu unterstützen.

1. MOTIVATION

Human beings do not reason from first principles but rely extensively on their memory of experiences while they are performing routine tasks [1, 2, 3]. The elementary methods and structure of the domain are used only, when we are faced with an unknown situation where we do not already know the solution, i.e. when we are novices in the domain [4, 5, 6].

Engineers behave in a similar manner while designing. Their experience plays a central role throughout the design process. Their ability to transfer the problem at hand into tasks to which solutions can be found, the approach they take and the quality of their solution depend on their capability to relate the problem to prior experience and domain knowledge (see also [7]). If the problem is familiar, engineers solve it directly. Only novices, or experts faced with a problem they do not have experience with, use the methods and structure of the domain extensively.

The elementary methods and structure form the grammar of a domain. A grammar provides us with means to analyze old, and to generate new cases (experiences) in that domain. We propose that Engineering Design Methodology (EDM) [e.g. 8, 9, 10, 11] can be used as a grammar for engineering design, and that systems that assist human engineers in performing design tasks can be realized by combining this grammar with experience in the form of cases.

Although EDM-research has been continuing for more than thirty years now, its acceptance among design engineers is not as broad as expected [5, 6]. This is actually not quite surprising if we consider the proposed similarity between a grammar of a domain and EDM. Like a grammar, engineers tend to apply EDM consciously especially when they are hard-pressed for solutions and know no way out. To design by purely using EDM equals to try to talk a foreign language by using only its grammar and a dictionary.

In this paper we describe the results of a research project in which a computer system based on the Case-Based Reasoning (CBR) methodology [1, 2, 3] is being developed. During the project, research in Cognitive Psychology, CBR and EDM was done. It was shown how the experience of a person develops during the design process, how it is indexed, retrieved, adapted and evaluated, and how it effects the solutions he produces. These results are used to develop and adapt the grammar provided by EDM to implement a CBR system, the Conceptual Design Assistant CoDA, which incorporates both experience and the grammar of the design domain.

2. COGNITIVE SCIENCE

To determine the development, influence and application of experience during problem solving we observed and videotaped test persons while they were solving simple design problems with the computer program "The Incredible Machine[1]". This program provides a design environment in which one can design simple machines to transport objects by using the elements provided.

We asked the test persons to build machines to solve four obligatory assignments in fixed order. After solving these, they had to select and solve two more out of twelve available assignments. None of the test persons knew the computer program beforehand. We transcribed the video recordings of the problem solving sessions into various types of protocols and analyzed them. Details of these experiments can be found in [12].

We were able to observe that test persons "understood" and solved the assignments based on the experience they had with similar situations. Towards the end of the test, especially during the two assignments the test persons chose themselves, the assignments were solved merely by using the objects or assemblies learned during the experiment. Even if the quality of the solution suffered, the test persons did not consider using the alternatives available.

Whenever a new solution was build by combining parts, the test persons learned the solution in whole as a new concept. These new concepts were indexed through the function they could fulfill and were treated as new, previously lacking, parts. Test persons stored all functions they knew an object could fulfill in conjunction with it and used these to achieve various goals.

During the experiment the test persons did not only increase their knowledge regarding the objects, but also concerning the methods they could apply to achieve their goals.

To solve problems the test persons combined these methods in various degrees of concretization with objects on different levels of concretization. Only through the combination of objects and methods the problem solving process did occur. Table 1 gives an example of the evaluation processes that occurred on various levels of concretization. During the tests, with growing experience, the test persons applied more concrete methods onto more concrete objects.

Evaluation of Solutions	Mental Object	Real Object
Mental evaluation	A hypothetical object is evaluated by thinking how it would behave.	A realized, existing machine is evaluated mentally. Often the test persons simulated, what they think is going to happen when the machine is activated, with the mouse pointer.
Real evaluation (test)	Parts of a planned machine planned are being tested to see how they would behave if the machine was built.	A realized machine is evaluated by starting it on the screen, thus performing a real test.

Table 1: Evaluating solutions on various levels of concretization by combining objects and methods

3. CASE-BASED REASONING

Based on the results of various experiments in Cognitive Science, researchers in the field of Artificial Intelligence devised a methodology, Case-Based Reasoning (CBR), for implementing systems that simulate the experience based problem solving approach of human beings [1, 2, 3].

A Case-Based Reasoner solves new problems by adapting solutions to previously solved similar problems to the new requirements. It is built upon the premise that human beings reason mostly from experience and not from first principles. A CBR System:

[1] "The Incredible Machine " is a registered trademark of Sierra On-Line Inc., Coarsegold, CA.

- finds and retrieves solutions from its case-base that have met the same or similar requirements,
- adapts the retrieved solutions to meet the current requirements,
- evaluates and if necessary repairs the adapted solution,
- suggests the solution it found to the user,
- and learns new cases from the solutions it generates (see Figure 1).

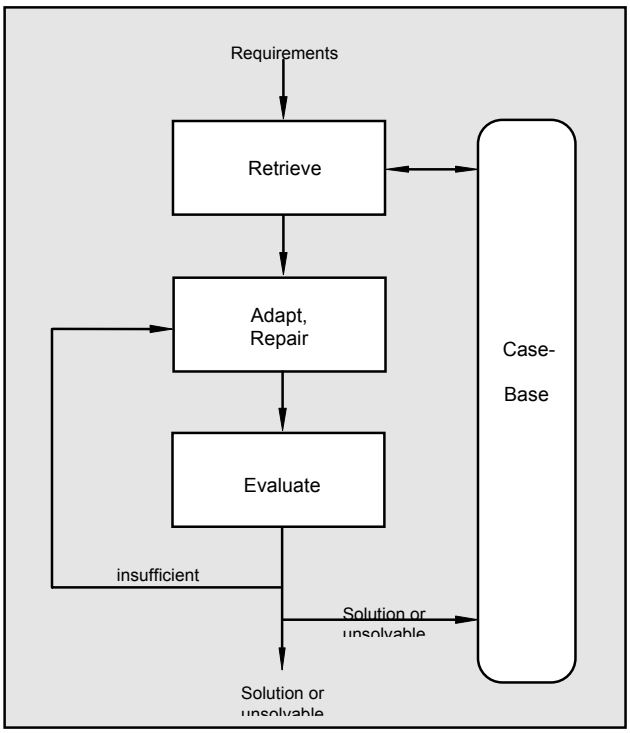

Figure 1: The Case-based Reasoning flowchart

4 ENGINEERING DESIGN METHODOLOGY AND THE CONCEPTUAL DESIGN ASSISTANT CoDA

Our experiments in Cognitive Science showed us how experience influences the problem solving process in real life. Case-Based Reasoning provides the framework to implement a system which reasons by simulating these processes. Engineering Design Methodology (EDM) on the other hand provides the methods and structure (i.e. grammar) necessary to realize a system that can be used in the design domain to support the engineer.

In this section we describe how the grammar provided by EDM is used to implement a Case-Based Conceptual Design Assistant - CoDA.

Figure 2 shows the basic structure of CoDA. For simplicity the reasoning mechanism of the CBR system (retrieval, adaptation, evaluation and repair) is separated from the case-base and is called the Reasoning Engine. The system is also complemented with a thesaurus that is used for indexing and retrieval purposes and the Control-Knowledge Base that contains global methods for the domain.

4.1 The Structure and Content of the Engineering Design Case-Base of CoDA

Similar to the design catalogues described by Roth [11] we distinguish among three types of cases[2]:

- Class-Cases
- Solution-Cases
- Method-Cases

Class-Cases store knowledge that is not directly associated with a specific problem but what may be considered general engineering knowledge (physical effects, material properties, environmental specifications,..). They provide solutions to a class of problems.

Solution cases map requirements of a certain design phase to a solution or to requirements for a following phase. The solution does not necessarily have to be on the following level of concretization but can also skip one or

[2] Roth distinguishes between object, solution and operation catalogues.

several phases of the design process. A solution that has been put together by joining partial solutions in various levels of concretization will be as abstract as the most abstract of the partial solutions used.

Method-Cases contain tactics, procedures and rules as well as their applicability and boundary conditions.

Figure 2 shows that the case-base contains several sub-case-bases that relate to separate phases of the design process. Each of these sub-case bases contain class, solution and method cases. Although the separation in sub-case bases is artificial and the number of cases that skip several phases will increase due to the learning aspect of this system, it helps clarifying the structure and content of the case-base with respect to the design phases.

The local methods (rules, procedures, tactics..) of EDM are represented as method cases in the case-base whereas the global methods (strategies, heuristics..) are incorporated in the system architecture as control knowledge.

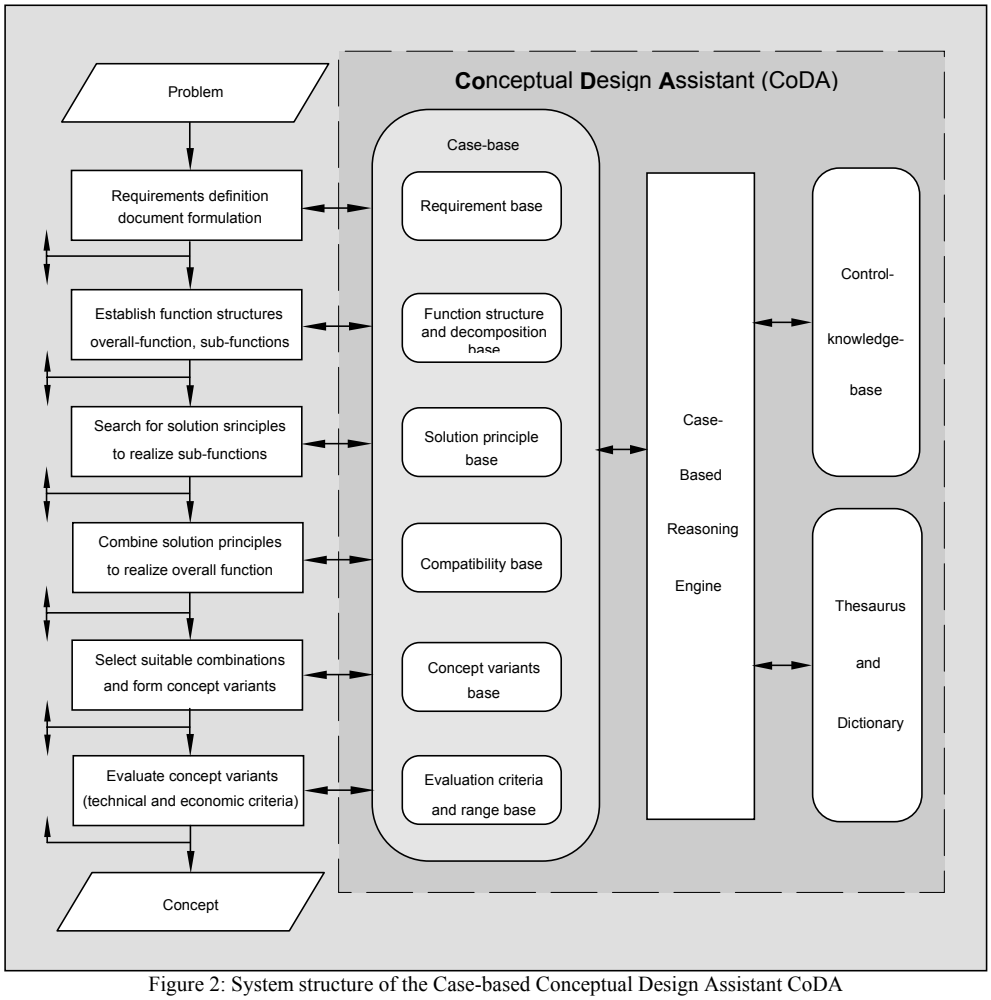

Figure 2: System structure of the Case-based Conceptual Design Assistant CoDA

Cases are represented as attribute-value pairs. Only the attributes that are necessary to be able to determine if the object is applicable in the given situation are taken into the list of attributes (i.e. object parameter lists [13]). It is definitely not the aim of the case-base to describe the objects with all their properties, in their entirety. The last attribute is therefore an identification number with which further documentation regarding the object can be obtained.

The cases can be structured according to two hierarchies (see also [14]). The abstraction / concretization hierarchy is based on a "is_a" relation and mirrors the development of the product according to the EDM phases. The aggregation / decomposition hierarchy on the other hand, is based on the "is_part_of" relation. While the abstraction / concretization hierarchy can be used by engineers that are designing a new concept and thus are interested in alternatives on a certain conceptual level the aggregation / decomposition hierarchy is more suitable for the retrieval of parts on a rather concrete level, e.g. for replacement or adaptation purposes.

The branches of the hierarchical structure represent object classes. All objects that belong to the same class on a certain level of hierarchy have the same attributes in their descriptions and differ in their values for these attributes. Objects that belong to different classes also differ in their attributes.

By describing relations between the attributes of a case or between cases it is possible to represent at least part of the product structure in the case base. These relations can be used in the adaptation process and serve as constraints when parts that have to work together have to be selected.

The ability to learn new cases is one of the most important aspects of case-based reasoning systems. In CoDA the proposed solutions as well as intermediate steps can be saved as new cases in the case-base. Since they are problem specific these will be Solution-Cases most of the time. The result of the learning process will be, that there will be more and more solution cases that skip one or more design phases. This will yield a Solution-case-base with cases of various concretization levels. We believe this is also the case in human engineers.

4.2 Requirements, Similarity Measures and Case Retrieval

The properties required of a new solution are entered into the system the same way a new case is entered. The system then tries to retrieve a case or cases that are most similar to the ideal solution described by the requirements the user entered.

The structure of the Case-Base serves as the basis for the efficient storage and retrieval of cases. Cases can be retrieved in three ways:

- inductive retrieval
- nearest neighbor retrieval
- template matching

In inductive retrieval the hierarchies, i.e. the differences in the attributes of the cases in the Case-Base are used as a basis for retrieval. Nearest neighbor matching assumes that the more properties the requested solution and the found case have in common the more similar they are. This is the approach that is also used in standard EDM evaluation procedures [15]. By using template matching one can search for a certain group (template) of properties to occur in the retrieved case. This allows us to select a group of objects that belong to separate classes but share relevant properties.

4.3 Adapting Cases and Repairing Solutions

The cases that have been retrieved from the case base have to be adapted to meet the requirements entered by the user. If the new solution, that has been generated from the retrieved case by adaptation is found to be insufficient after the evaluation process, it has to be repaired.

Both adaptation and repair can be performed property-oriented or on a structural level. While in property-oriented adaptation and repair the properties are adjusted directly to meet the new requirements, during structural adaptation the structure (e.g. function structure) of the retrieved or newly created solution has to be modified.

Adaptation and repair on the structural level require a deeper understanding of the domain than the simple property oriented approach which can be coded into the relations between the attributes of the cases.

Nevertheless adaptation and repair are topics that cannot be automated entirely in a domain as complex and complicated as engineering design.

4.4 Evaluation

The evaluation step requires the selection of attributes that can serve as evaluation criteria and the determination of the weighting factors for these criteria. The evaluated solution and the ideal solution described through the requirements belong to the same object class and thus have the same attributes. This allows us to apply nearest neighbor methods to determine the quality of the created solution. The more similar the solution and the requested object are, the better the suggested solution is. The methods provided by EDM [8,15] can be applied directly.

4.5 The Thesaurus and the Control-Knowledge Base

The thesaurus and dictionary provide a standard for describing and accessing cases. The cases in the Case-Base are indexed using the terms in the dictionary. To ensure that the user is not constrained by the set of terms provided by the dictionary a thesaurus functionality, in which the terms the user finds suitable and the system uses can be linked with each other, is added.

The global methods of EDM that are applicable throughout the design process form the control knowledge of the system. These are methods that deal with the global consistency and properties of the technical object as well as methods for abstraction / concretization and aggregation / decomposition and are contained in the control-knowledge base.

5. CURRENT STATUS AND CONCLUSION

Currently the project is in the implementation phase. We are working with cases from the drive technology domain. A first working prototype of the system will be tested in an industrial environment at the end of this year.

Our aim is to support the engineer with methods and domain experience in a natural way and not to obstruct by useless prerequisites, models and details. Case-Based Reasoning provides a technology to implement systems that can support the engineer in a very efficient and effective way. The approach is based on knowledge units that are intuitively understandable and obtainable, which makes the system easy to use and maintain. We believe, that in contrast to systems where the knowledge structure behind the system is off-limits for the user, the intuitively understandable knowledge in the form of cases will make the Conceptual Design Assistant an appreciated tool for engineers.

References

[1] C. Riesbeck, R. Schank, "Inside Case-Based Reasoning", Lawrence Erlbaum Associates, Publishers, Hillsdale 1989

[2] K. Hammond, "Case-Based Planning - Viewing Planning as a Memory Task", Academic Press Inc, HBJ Publishers, San Diego, 1989

[3] J. Kolodner, „Case-Based Reasoning“, Morgan Kaufmann Publishers Inc, San Mateo, 1993

[4] L. Kochevar, P.Johnson, "Problem Solving is What You Do When You Don't Know What to Do", Eleventh Annual Conference of the Cognitive Society, pp 615-622, 16-19 August 1989, Ann Arbor, Michigan, Lawrence Erlbaum Associates,Hillsdale 1989

[5] J.Müller, "Akzeptanzbarrieren als berechtigte und ernstzunehmende Notwehr kreativer Konstrukteure", Proceedings International Conference on Engineering Design ICED 91, pp769-776

[6] G. Pahl (ed.), "Psychologische und pädagogische Fragen beim methodischen Konstruieren - Ergebnisse des Ladenburger Diskurses von Mai 1992 bis Oktober 1993", Verlag TÜV Rheinland GmbH., Köln, 1994

[7] A. Rutz, "Konstruieren als gedanklicher Prozeß", Technische Universität München, Dissertation, 1985

[8] G.Pahl, W.Beitz, "Konstruktionslehre", 3nd. Ed., Springer Verlag, Berlin, Heidelberg, 1993

[9] Verein Deutscher Ingenieure, VDI Guideline 2221: "Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte", VDI Verlag, Düsseldorf 1986

[10] Verein Deutscher Ingenieure, VDI Guideline 2222, Part 1: "Konstruktionsmethodik, Konzipieren technischer Produkte", VDI Verlag, Düsseldorf 1977

[11] K. Roth, "Konstruieren mit Konstruktionskatalogen", 2nd edition, Vol. 1-2, Springer Verlag, Berlin, Heidelberg, 1994

[12] M.Göker, H. Birkhofer, "Problemlösen mit 'The Incredible Machine' - Ein Experiment zum Fallbasierten Schließen", Proceedings, Workshop on Case-Based Reasoning, German Expert System Conference XPS 95, 28.2.1995-3.3.1995, Kaiserslautern, Germany

[13] Normungsauschuß Sachmerkmale (NSM) der DIN Deutsches Institut für Normung e.V., "Sachmerkmal-Leisten, Begriffe und Grundsätze - DIN4000", Beuth Verlag, Berlin 1991

[14] A. Suhm, "Produktmodellierung in wissensbasierten Konstruktionssystemen auf der Basis von Lösungsmustern", Dissertation Universität Karlsruhe, Verlag Shaker, Aachen, 1993

[15] Verein Deutscher Ingenieure, VDI Guideline 2225, "Konstruktionsmethodik - Technisch-Wirtschaftliches Bewerten", VDI Verlag, Düsseldorf 1977

Authors :

M. Göker, M.S.E., M.Sc. and Prof. Dr. -Ing. H. Birkhofer

Institute of Machine Elements and Engineering Design, Technical University Darmstadt

Magdalenenstr. 4 D-64289 Darmstadt - Germany

e-mail : {goker, birkhofer}@muk.maschinenbau.th-darmstadt.de

Tel : +49 6151 162155 Fax : +49 6151 163355