#. Case-Based Reasoning

Klaus-Dieter Althoff

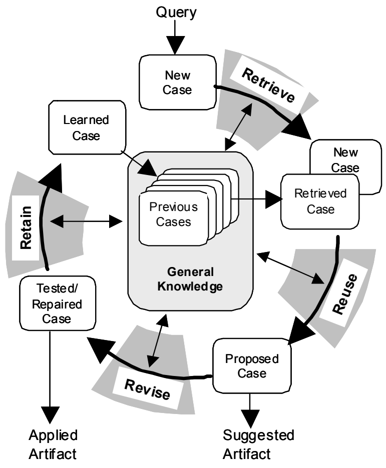
Fraunhofer Institute for Experimental Software Engineering (IESE)
Sauerwiesen 6
D-67661 Kaiserslautern, Germany
Email althoff@iese.fhg.de

Abstract

Case-Based Reasoning (CBR) is an Artificial Intelligence approach to learning and problem solving based on past experience. CBR combines aspects from the knowledge-based systems as well as from the machine learning field. In this chapter we present the main characteristics of CBR technology, describe the main CBR subtasks and methods, and the main underlying principles. CBR is presented more from the user and usage point of view: We consider different application types, survey CBR applications up to now, and briefly present methodologies for developing and evaluating CBR systems. Finally, important issues concerning the past, present, and future of CBR overview the field for the reader. A huge number of references supports the reader with information where to get more detailed technical descriptions on various CBR related topics of interest.

Introduction

Case-Based Reasoning (CBR) is an approach to learning and problem solving based on past experience. A past experience is stored in the form of solved problems ("cases") in a so-called case base. A new problem is solved based on adapting solutions of known similar problems (see Figure 1) to this new problem.



Case := Problem (characterization)/Solution (artifact) Pair

Query: Query at hand defines new case (problem without solution)

Retrieve: New case is used to find most similar case among the known (previous) cases

Reuse: New and retrieved case are combined to a proposed case including the suggested artifact

Revise: Suggested artifact is applied and evaluated

Retain: Useful experiences from applying the artifact are retained by adapting the case base and the conceptual knowledge

Figure 1. Case-Based Reasoning Process Model

For solving a new problem a query is submitted to a CBR system to retrieve the solutions of the most similar problems/cases in the case base. The query is considered as a potential new case. The classical view on a case imposes that it consists of a problem description and a described solution to this problem. Recent applications have shown that this view is too restrictive [63]. A more general view on cases is that a case consists of a characterization (a more or less structured set of information entities) and - optionally - one or more artifacts [31, 32]. The characterization part is mandatory and usually contains both formal parts, which can be interpreted by the CBR system, and informal parts, which can be understood by the user only. The characterization can be completed by links to any kind of artifact. Examples for artifacts are web pages, pictures, audio-visual media, or formally described pieces of knowledge. During the retrieval a part of the formal characterization of the new case is specified. The CBR system uses this for matching against the characterizations of the other cases in the case base. The most similar cases are retrieved and parts of their characterizations and associated artifacts are combined to yield a proposed case. Applying the suggested artifact(s) associated with the proposed case results usually in a repaired case and validated and/or revised artifact(s). The repaired case with its applied artifact(s) can be stored as a learned case, merged with previously available cases, and/or only extracted parts of it are stored. These problem solving steps are also known as the CBR cycle and/or CBR process model [7].

CBR has been used to create numerous applications in a wide range of domains including

- financial analysis,
- risk assessment,
- technical maintenance,
- process control,

- quality control,
- medical diagnosis,
- software support systems,
- forecasting,
- planning,
- design,
- classification of objects,
- photo-interpretation,
- real estate appraisal,
- electronic commerce,
- customer support,
- knowledge management,
- software engineering,
- etc. [23, 26, 52, 51].

The underlying idea of CBR is simple: Do not solve problems from scratch but remember how you (or someone else) solved a similar problem and apply this knowledge to solve your current problem. The notion of *case-based reasoning* was introduced by Roger Schank. The motivation of his work was based on a cognitive science perspective and one early application area was the cognitive oriented research on story understanding. Another root of current CBR research was the motivation to use knowledge-based methods for legal reasoning, initiated by Edwina Rissland [154]. A third root for current CBR might be found in research on analogical reasoning (e.g., [83]), though early work on CBR and analogical reasoning were quite independent from each other for many years.

After more then twenty years of existence "modern CBR" still consists of work done in the areas mentioned above. However, it has also become an engineering discipline for using knowledge and software engineering methods to develop real-life applications. In the beginning of the 90's the first commercial CBR tools appeared and, since then, more mature tools have been developed [23, 173]. While for some years the strong focus on various real-life applications seemed to separate current CBR research and development activities from its origins, in the most recent years methods and ideas from the more humanities oriented parts of CBR again become very important. Complex application domains like knowledge management, education and training, and continuous quality improvement require broadly applicable techniques. It is one of the biggest challenges of current CBR research and development to not only develop solutions for very specific (small) problems but to also develop more general solution strategies that allow, for instance, an integration of computer and human based methods.

To give you a glimpse of what CBR covers, we introduce some of the most important aspects. They already make some basic characteristics of CBR obvious.

1. CBR is an approach for developing knowledge-based systems (e.g., [38], [2]).

Through CBR, knowledge in the form of cases can be formally represented and used for automated problem solving, for instance, to aid some decision-maker [108], to solve diagnostic as well as planning or design problems [37, 73, 122, 181, 53, 127, 58]. Besides case-specific knowledge, CBR systems also exploit general domain knowledge (e.g., a set of rules) to support retrieval, similarity judgment, case adaptation, as well as learning[1]. Thus, CBR can be viewed as an approach for developing knowledge-based systems (see also [131]).

2. CBR is an approach for learning from experience (examples, cases) [9, 100, 84, 38].

   Learning from cases can be implemented using CBR. The learning result then is the whole CBR system including its case base. This result can, for example, be incrementally improved through adding new cases.
3. CBR is an explanation model for human problem solving and learning [157, 104, 105, 158, 115].

   This is - in part – the origin of CBR.
4. CBR is a very natural approach for the development of knowledge, especially in the context of teaching and tutoring [159, 186, 187, 139, 164].

   Of course, here is some strong relationship to 3.
5. CBR is a knowledge based extension of the nearest neighbor classifier paradigm known from pattern recognition [17, 11, 98, 60].

   On an abstract level nearest neighbor classification is very similar to CBR. However, CBR enables also to use explicit knowledge to improve retrieval and reuse. In addition, it offers an approach to combine quantitative and qualitative approaches because both kinds of knowledge can be easily integrated within a CBR system [174].
6. CBR is a holistic system approach [26, 93].

   Here CBR is different, for instance, from many approaches in the machine learning community that are more oriented towards developing optimized algorithms for specific problems and, therefore, do not consider requirements (for building software systems) from knowledge/software engineering very much.

   This is also a very important aspect to be considered if learning (knowledge-based) *systems* have to be developed.
7. CBR is an organizational[2] approach for experience management [40, 175, 185, 130].

   Recent work on knowledge management and continuous quality improvement in software engineering [102, 95, 119, 112] made it very obvious that due to CBR's various relations to work in the humanities, some of CBR's underlying principles can be successfully applied to the human

[1] Some explanations for differentiating case-specific from general domain knowledge can be found in the Sections on *Case-Based Reasoning Task-Method Decomposition Model, Case-Based Reasoning Systems,* and *Case-Based Reasoning Knowledge Container Model.*

[2] as well as technical

(and organizational) aspects of experience management [143, Tautz 2000, 176, 75, 156].

8. CBR is an open environment for integrating different kinds of techniques. It is a matter of fact that various hybrid systems have been built with the use of CBR [16, 120, 12, 138, 4, 5, 80, 21]. In addition, methods from other disciplines have been successfully integrated into CBR approaches (e.g., *similarity measures*: [64, 55]; *object-centered representation*: [56, 177]; *inductive learning*: [128, 166, 61, 71]; *databases*: [165, 190, 66, 163]; etc.)
9. Knowledge based information retrieval [100, 126, 190, 155].
   CBR can also be viewed as an approach to retrieve relevant information from some information base. Here CBR offers opportunities to integrate knowledge into the retrieval process, to improve the retrieval system through learning, and to adapt the retrieval results to the current problem at hand.

In the next section CBR is described from various perspectives that reflect the current state of the art/practice. We then present one introductory CBR application example and give a short overview on already existing/documented CBR real-life applications, present approaches on the evaluation of CBR systems and applications, and describe some basic aspects on systematic development of CBR applications. Finally, a historical overview on the development of CBR is given, current CBR information resources are briefly surveyed, and some future directions are outlined.

Description of CBR

We shall describe CBR from various points of view. First we shall introduce the task-method decomposition model of CBR by Aamodt & Plaza [7], which details the CBR process model described above. Since CBR is very much concerned with developing software *systems* we shall contrast CBR systems to database systems as well as (other) knowledge-based systems. This will be completed with a briefly described organizational model for running CBR systems, which includes basic roles involved as well as a goal-oriented procedure of how to develop knowledge for CBR systems. The knowledge container model - first introduced by Michael Richter [149] - is an explanation model for some basic characteristics of CBR systems, which is widely accepted in the CBR community and should not be left out in a handbook chapter like this. Finally, in this section some case-based reasoning methods are introduced for different kinds of application tasks, namely classification, diagnosis, decision support, and knowledge management.

Case-Based Reasoning Task-Method Decomposition Model

At the highest level of generality, a general CBR cycle may be described by four tasks [7]: *Retrieve* the most similar case or cases, *reuse* the information and knowledge in that case to solve the problem (a process called "solution transfer"), *revise* the proposed solution, and *retain* the parts of this experience likely to be useful for future problem solving (Figure 1).

Note that the tasks referred to here are internal reasoning tasks, and differ from application problem tasks like diagnosis, planning, scheduling, etc. The four CBR tasks each involve a number of more specific subtasks. An initial description of a problem (top of Figure 1) defines a new case. In the *retrieve* task this new case is used to find a matching case from the collection of previous cases. The retrieved case is combined with the input case - in the *reuse* task - into a solved case, that is, a proposed solution to the initial problem. The *revise* task tests this solution for success, for example, by applying it to the real-life environment or having it evaluated by a teacher, and repaired if failed. This task is important for learning, since the system needs feedback on how successful its proposed solution actually was. *Retain* is the main learning task, where useful experience is retained for future reuse, by updating the case base and possibly also the general domain knowledge.

As indicated in Figure 1, general knowledge usually plays a part in this cycle, by supporting the CBR processes. This support may range from very weak to very strong, depending on the type of CBR method. By general knowledge we here mean general domain-dependent knowledge, as opposed to the specific domain knowledge embodied by cases. For example, in diagnosing a patient by retrieving and reusing the case of a previous patient, a model of anatomy together with causal relationships between pathological and other physiological states may constitute the general knowledge used by a CBR system. A set of rules may have the same role.

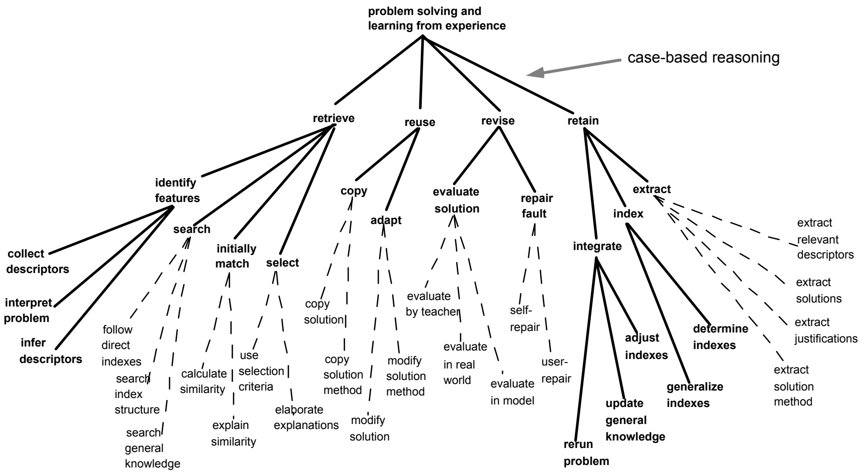


Figure 2. Task-method decomposition of case-based reasoning [7]

The tasks from Figure 1 are further decomposed in Figure 2 [7, 22]. The tasks have node names in bold letters, while methods are written in plain text. The links between task nodes (bold lines) are task decompositions, that is, part-of relations. The links between tasks and methods (stippled lines) identify alternative methods applicable for solving a task. The top-level task is *problem solving and learning from experience* and the method to accomplish the task is a *case-based reasoning* method. This splits the top-level task into the four major CBR tasks corresponding to Figure 1. All the four tasks are necessary in order to perform the top-level task. The *retrieve* task is, in turn, partitioned in the same manner (by a retrieval method) into the tasks *identify features, search* (to find a set of past cases), *initially match* (the relevant descriptors to past cases), and *select* (the most similar case(s)). All task partitions in the figure are considered complete; that is, the set of subtasks of a task are intended to be sufficient to accomplish the task, at this level of description. The figure does not show any control structure over the subtasks. Control is specified as part of the problem-solving method. The retrieval method, for example (not explicitly indicated in the figure), specifies the sequence and loop-backs for the subtasks of *retrieve*. A method specifies the algorithm that identifies and controls the execution of subtasks, or solves the task directly, while accessing and utilizing the domain knowledge needed to do this. The methods shown in the figure are high level method classes, from which one or more specific methods should be chosen. The method set as shown is incomplete; that is, one of the methods indicated may be sufficient to solve the task, several methods may be combined, or there may be other methods that have not been mentioned.

This structure provides the basis for an analytic framework. It needs to be elaborated and described in more detail. Characterizations of domain knowledge types need to be added, and dependencies between the various knowledge types need to be identified (see [22, 21].

Case-Based Reasoning Systems

In general, the decision how much general knowledge is needed for an application and whether it should be used to complete a CBR system or, vice versa, whether it should be used to build a (general) knowledge-based system, depends on the available knowledge sources that are related to given application, their degree of availability, as well as the respective reliability of the knowledge sources. General knowledge used within a CBR system can help to reduce the number of cases necessary for problem solving, to improve the reliability of potential solutions, to make the overall system more efficient in the handling of routine situations, to exploit *known* generalizations, and to immediately adapt to a changing environment. CBR offers a flexible intermediate position between database systems containing mostly informal knowledge, which can only be interpreted by the user, and knowledge-based systems including widely formalized knowledge, which can be automatically processed by the computer. The advantage offered here by a CBR system is that it can start from a very simple database-system-like situation and can use increasingly more formal knowledge depending on the respective application (Figure 3).

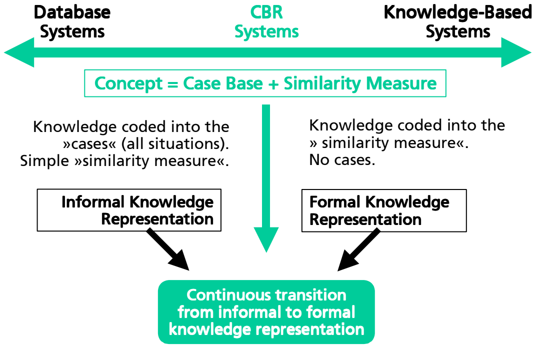

Database Systems
CBR Systems
Knowledge-Based Systems
Concept = Case Base + Similarity Measure
Knowledge coded into the »cases« (all situations). Simple »similarity measure«.
Knowledge coded into the » similarity measure«. No cases.
Informal Knowledge Representation
Formal Knowledge Representation
Continuous transition from informal to formal knowledge representation

Figure 3. Case-based reasoning knowledge model

From the perspective of continuous learning from experience CBR systems can be embedded in industrial and other kinds of organizations as roughly shown in Figure 4. As first mentioned by Henninger [94] and later explained in detail by Althoff and Wilke [40] and Tautz and Althoff [175], CBR and *experience factory* [48, 59, 156] share many similar assumptions and are to some degree complementary. While CBR provides an approach to learning from experience on the technical system implementation level, an experience factory - together with its underlying *quality improvement paradigm* (QIP) ([48], Figure 5, [156]) - supports it as a physical and/or logical infrastructure on a technology-independent organizational level. Various research institutions, groups, and people have already started to benefit from the synergy of these two approaches [40, 175, 52, 44, 99, 174].

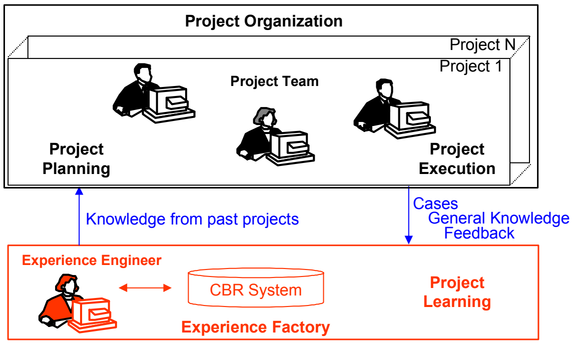


Figure 4. Organizational model for running CBR systems

By adopting an experience factory approach for running CBR systems in organizations, two immediate benefits are available. First project learning can be facilitated through goal-oriented knowledge development based on the QIP (Figure 5, [156]). Second some basic roles necessary to operationalize CBR systems in practice can be described as follows:

- The *manager* provides resources, defines strategic goals, and initiates improvement programs. He defines the structure of the case base and controls its quality.
- The *supporter* is responsible for documenting new experiences and supporting the project teams. He collects and qualifies artifacts from projects according to the reuse criteria and goals of the experience engineer. He supports the project team on request for retrieving cases and/or adapting them to the current project situation.
- The *experience engineer* is responsible for packaging and analyzing existing cases. Together with the manager he identifies new reuse criteria and acquires new cases based on these criteria. He analyzes the case base to discover further improvement potential.
- The *librarian* is responsible for technical tasks like development and maintenance of the case base. He stores and publishes new cases.

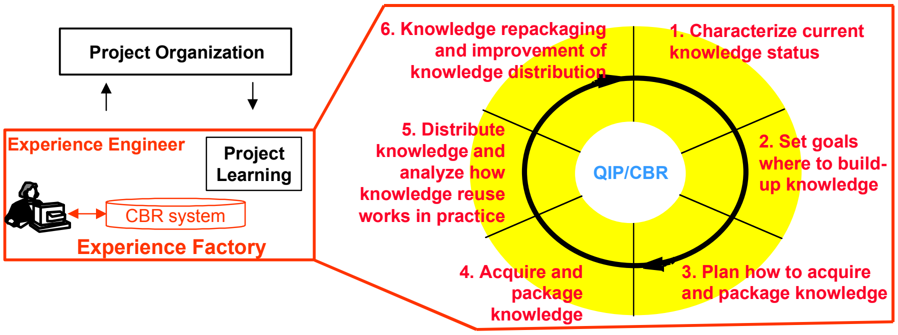

Project Organization
Experience Engineer
Project Learning
CBR system
Experience Factory
6. Knowledge repackaging and improvement of knowledge distribution
1. Characterize current knowledge status
5. Distribute knowledge and analyze how knowledge reuse works in practice
QIP/CBR
2. Set goals where to build-up knowledge
4. Acquire and package knowledge
3. Plan how to acquire and package knowledge

Figure 5. Goal-oriented knowledge development for CBR systems

Case-Based Reasoning Knowledge Container Model

Richter [151] has introduced a helpful explanation model or view on CBR systems (Figure 6). He identified four knowledge containers for a CBR system. Besides the underlying vocabulary, these are the similarity measure, the solution transformation, and the cases. While the first three represent compiled knowledge (i.e., more stable knowledge is assigned to these containers), the cases are interpreted knowledge[3]. As a consequence, newly added cases can be used directly. Therefore, the cases enable a CBR system to deal with dynamic knowledge more efficiently and effectively. In addition, knowledge can be shifted from one container to another when this knowledge has become more stable. For instance, in the beginning a simple vocabulary, a rough similarity measure, and no knowledge on solution transformation are used. However, a large number of cases are collected. Over time, the vocabulary can be refined and the similarity measure defined in higher accordance with the underlying domain. In addition, it may be possible to reduce the number of cases because the improved knowledge within the other containers now enables the CBR system to better differentiate between the available cases. If knowledge about solution transformation also becomes available, it may be possible to further reduce the number of available cases, because then a number of solutions can be derived from one case.

[3] Of course, knowledge for solution transformation or similarity assessment can be captured and represented as cases. Thus, a “small” CBR system is used, for instance, for transforming a case’s solution. However, we would like to avoid this recursive view in an explanation model.

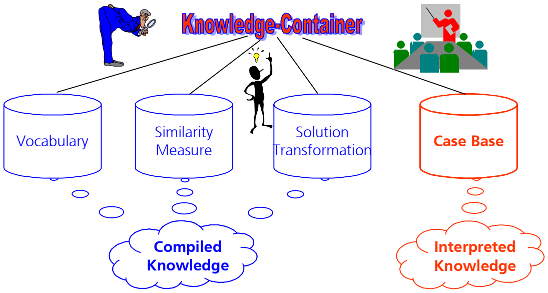


Figure 6. The distribution of knowledge in a CBR system [Richter 1998)

Different Degrees of User Involvement in Solution Development

CBR is a technology that - usually - involves the user in solution development. In the following we briefly present three different types of (basic) CBR systems, where the degree of user involvement increases (Figure 7). In addition, one more complicated application task is presented that requires the integration of basic reasoning methods to support the processes available in the underlying application context.

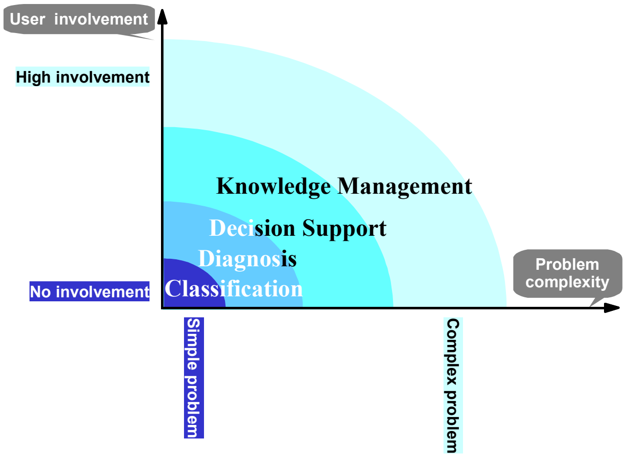


Figure 7. More user involvement enables solution development for more complex problems

Case-Based Classification

The basic classification process maps objects or situations to a given set of classes. We assume that all necessary information is available to carry out the mapping process (Figure 8). Examples of classification tasks are risk assessment of cars, determination of biological objects, analysis of client data, or cost estimation [79]. For basic classification tasks CBR is in competition with many other approaches. While inductive machine learning approaches normally reason with generalized knowledge and, by contrast, databases store all the case data, CBR is more flexible in the sense that it distributes its competence on both the cases and the similarity measure (Figure 3, [194, 150, 84]). Depending on the specific requirements of an application a simple or a well-informed measure can be used. The better the similarity measure is suited for a given task, the fewer cases must be stored. In principle, the CBR approach offers the flexibility to decide this at a very late stage in system development. Always when this flexibility is wanted, CBR is a good choice. Comparing CBR with approaches based on generalized knowledge results in similar advantages and disadvantages like those associated with (case) interpretation and (case) compilation.

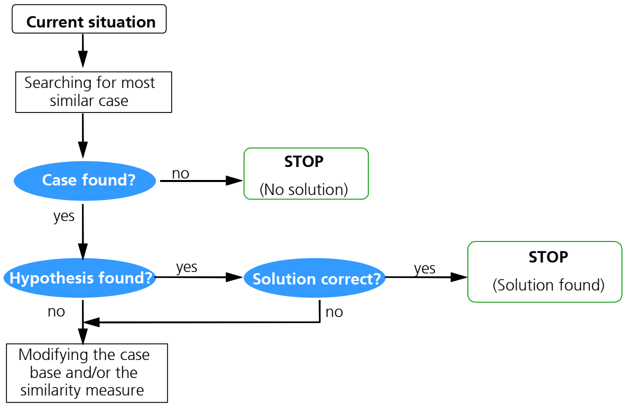


Figure 8. Basic reasoning method for case-based classifcation

CBR systems (e.g., PATDEX: [37, 189]; INRECA: [21, 190] often use at least one similarity measure per class, which can be of high value for certain classification problems. Additionally, both systems can learn the relevancies of certain attributes for the respective classes (e.g., [148]). Aamodt and Nygård [6] pointed out that depending on their role in decision making and learning by experience, cases may be viewed as data, information, or knowledge and, thus, the CBR approach maybe helpful for certain domains where the integration of such issues is important.

Case-Based Diagnosis

Diagnostic processes - as considered here - differ from basic classification processes by the problem of incomplete information. That is, before carrying out the classification process normally (much) more information must be acquired. Thus, the diagnosis task includes, in addition to the classification task, the task of selecting the best test(s) to acquire the missing information as cheaply, fast, and securely as possible. Examples are fault diagnosis of a car motor, a CNC[4] machine tool (see section on Application Example: Fault Diagnosis of Technical Equipment), or an aircraft engine.

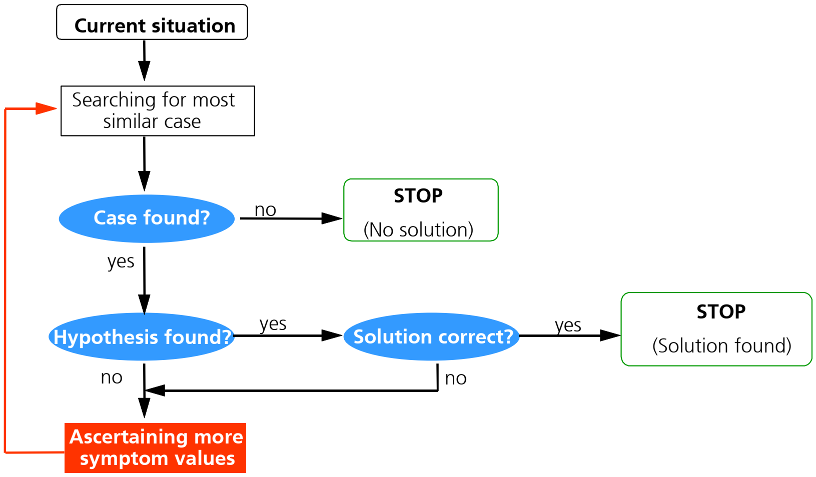


Figure 9. Basic reasoning method for case-based diagnosis

The additional test selection subtask makes the diagnosis task a very specific one (Figure 9). Often general domain knowledge is required to reasonably guide the selection process. For some application tasks (e.g., fault diagnoses of CNC machines) the degree of incomplete information can be very high (>90%). Therefore, approaches based on induction have serious difficulties in coping with such problems in an efficient way. This is again due to the interpretation/compilation contrast we mentioned in the previous section. CBR can also offer a solution to the test selection problem itself by using an additional kind of cases or by structuring its retrieval according to explicitly specified retrieval goals [86]. In principle, general knowledge about the biological or the engineering system can be used to adapt the diagnostic class in the case description.

Case-Based Decision Support

Decision support processes differ from classification processes by the necessity of representing more general knowledge and allowing user-computer interaction to a very high degree [122, 190, 26]. While the classification goal is clearly defined for the classification and the diagnosis processes (static target definition), decision support processes must cope with (more explorative) problems where the classification goal is defined during the problem solving process (dynamic target definition). Examples are looking for a house to buy, looking for a last-minute trip without clearly defining the location and the kind of travel, and looking for a well-suited used or new truck for a company.

[4] Computerized Numerical Control

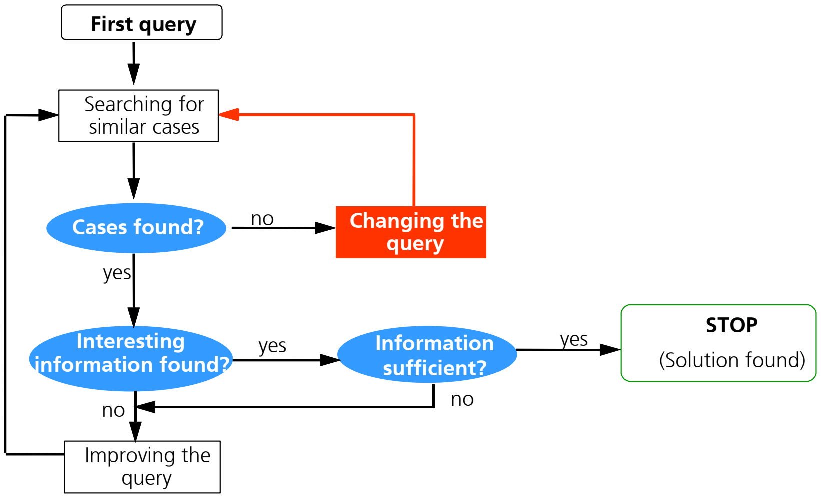


Figure 10. Basic reasoning method for case-based decision support

To support a basic algorithm for case-based decision support as given in Figure 10, usually more domain knowledge [122, 190], efficient and effective retrieval structures [42, 191, 123, 21], as well as improved similarity assessment is necessary [148, 190, 195, 21, 86].

Case-Based Knowledge Management

A much more complicated application task than the three already mentioned ones is knowledge management. It is used here just as an example. Other complicated tasks include most planning and design problems [183, 137].

For knowledge management no basic reasoning method is directly applicable. Instead various reasoning methods are relevant to support knowledge management processes[5] including the ones mentioned above (e.g., [75, 8, 143, 88]). Figure 11 presents an example of a CBR architecture for knowledge management [32], which also shows the high-level processes of structuring knowledge, recording knowledge in the experience base, and reusing/adapting/aggregating it. Since more complex application tasks require the integration of different reasoning methods, we present four different levels of how to integrate them in principle (Figure 12).

[5] According to Probst and Romhardt [144] there are at least processes for knowledge goal definition, knowledge acquisition, knowledge processing, knowledge preservation, knowledge distribution, knowledge utilization, and knowledge valuation (see also [34]).

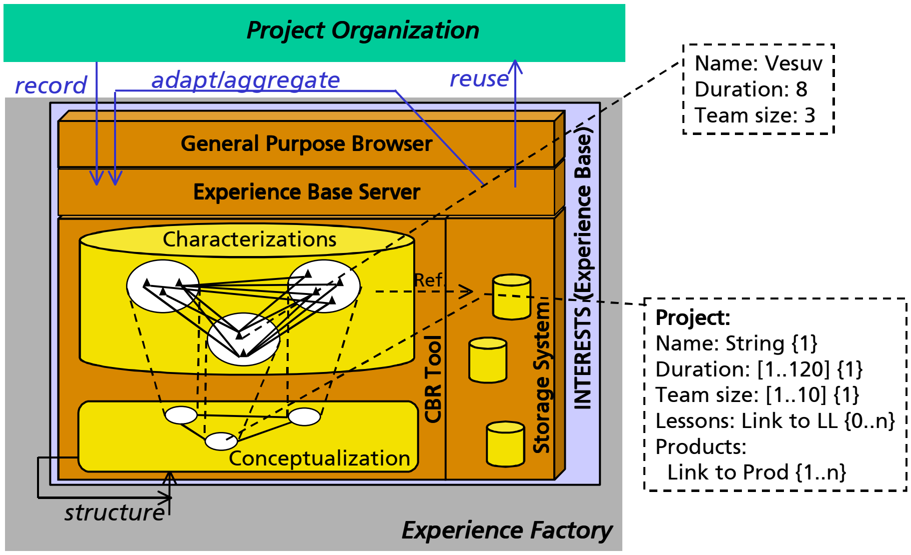


Figure 11. Intelligent Retrieval and Storage System supporting some high-level experience factory processes

These integration levels depend on the degree of interactiveness between certain reasoning methods [23, 21, 15, 132]:

- *Toolbox level:* Here the integration is restricted to the common use of parts of the knowledge base. Reasoning is done by only one method. Exchange of reasoning results is not possible.
- *Cooperative level:* During the reasoning process different reasoning methods can be used. The exchange of intermediate results is possible using a common representation formalism [128].
- *Workbench level:* Beyond switching between available reasoning methods, they can also be aggregated into a combined reasoning method. For instance, a diagnostic system can carry out its classification subtask using a rule-based module, and the test selection subtask using a CBR component [19].
- *Seamless level:* This is the highest level of integration. Different reasoning methods have been integrated within a new algorithm based on an integrating element. Thus, the change of reasoning methods is hidden from the user [2].

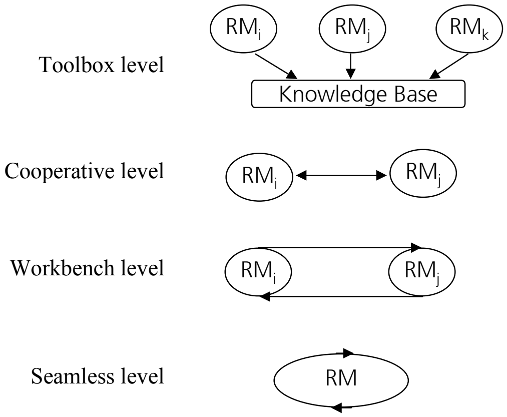


Figure 12. Integration of different reasoning methods

Usage of Case-Based Reasoning Technology

CBR is an approach to develop knowledge-based systems. It provides a huge number of formal computer science techniques for executing, among others, the subtasks mentioned in the CBR task-method decomposition model in Figure 2. Some of the characteristics of a domain that indicate whether a CBR approach would be suitable are [23]:

1. There exist records of previously solved problems,
2. historical cases are viewed as an asset that ought to be preserved,
3. if there is no case history, it is intuitively clear that remembering previous experiences would be useful,
4. specialists talk about their domain by giving examples,
5. experience is at least as valuable as textbook knowledge,
6. approximate solutions are acceptable.

While the first five indicators need not to be fulfilled all, of course, the sixth indicator is more like a requirement or an assumption. It is aiming at the very "heart of CBR" as a "nearest neighbor" technology [152], which provides solutions that are more or less appropriate (similar). In some sense CBR is an approximation approach like Fuzzy technology. If no approximate solution is requested but one that is 100% correct, for instance, a logic based approach is better suited than CBR. Here perhaps other problem solving methods are more appropriate [131].

However real-life applications very often require that not only one approach is used. The integration of different reasoning methods [106, 29], as for instance described in the section above, is one important solution to this problem. For the integrated development of knowledge systems (see also Section on CBR Application Development), the contribution of CBR is – besides integrating knowledge technologies with learning and approximation – the introduction of some "locality principle" for knowledge. Cases can be understood, communicated, and validated as they are, that is (to a high degree) independent from the rest of the CBR system. Cases are especially independent from each other. For example, if one important case is missing in a case base, this would only result in an answer of the CBR system where this case might be missing as a similar one. The other similar cases can still be provided to the user. This is different, for instance, than in rule-based or other kinds of knowledge-based systems where rules or other kinds of knowledge usually interact in a much more complicated way. Thus, CBR and cases introduce more transparency into knowledge-based systems [131]. How the different kinds of knowledge interact within a CBR system has partly been described in the Section on the Case-Based Reasoning Knowledge Container Model.

The issues of competence and maintenance of CBR systems are currently hot research topics. Among others, Mark Keane and especially Barry Smyth, together with people working in his group, already started to look deeper into the issue of competence of a CBR system [168, 169, 197, 118]. In the recent past several research efforts on the maintenance of CBR systems have been started [82, 92, 117, 147, 134]. Open research issues include a comparison of competence development and maintenance effort of CBR systems and other knowledge-based systems, the influence of machine learning techniques on competence development and maintenance effort of CBR systems, as well as the competence development and maintenance effort of hybrid knowledge-based systems [2, 5, 120] that also include a CBR part.

CBR Applications

We shall first present a simple application example on fault diagnosis for technical equipment, which hopefully demonstrates some basic and informative features of CBR systems. Then we give an overview of past and present CBR applications. In addition, both subsections include various links to CBR applications described in the literature.

Application Example: Fault Diagnosis of Technical Equipment

We shall briefly present a simple example of a CBR application from the domain of fault diagnosis of technical equipment. More concretely, the application task of diagnosing a CNC[6] machining center is chosen, briefly called *CNC domain*. This application was developed in the second half of the 80's and is based on real-life data but remains a research prototype: the PATDEX system. However, it was the starting point for many other CBR systems and applications [21].

The CNC domain is difficult to handle because of the huge amount of unknown values. In addition, the overall number of attributes and classes is very

[6] Computerized Numerical Control

high. This domain also illustrates the necessity of having integrated systems [140, 19], even if treated as a more closed domain, that is, ignoring influences outside of the machine like vibrations of the workshop floor. However, it is not possible to view it as a static domain [37, 19, 74]. One reasonable approach here is to use/view cases as exceptions [37, 19].

The CNC domain is a good example for the need of having a well-suited and flexible test selection component that includes, among others, the following features:

- Asking for the next symptom value.
- Accepting any symptom value at any step in the user-system interaction.
- Accepting any number of symptom values.
- Accepting user hypotheses [33].

The PATDEX system [189] was originally designed for the CNC domain. Though PATDEX's similarity measure is well-suited for it, in the similarity assessment scenario below it will retrieve the wrong case (CaseIncorrect). However, it is able to improve this situation by its test selection component (asking for symptom values to improve the given information) and/or by the application of causal and/or heuristic rules that allow the derivation of additional symptom values based on known ones. In addition, PATDEX can use weights for the relative importance of the given attributes for the respective classes (diagnoses). These weights can be automatically updated to avoid misclassifications [189; 148]. The use of default values for symptoms and its special handling of pathologic/abnormal symptom values also allow an improved treatment of situations similar to that given in the similarity assessment scenario below. Later extensions of PATDEX also allowed using a deep model of the underlying CNC machine's behavior to improve the similarity measure [54].

Let us consider the following scenario. The CNC machine has stopped with an error message code “i59”. In addition, the maintenance technician knows that some specific input/output status “IN32” has the value “logical-1” and that the valve 5Y2 is not switched. He formulates a query to the PATDEX system that includes this information (right column of Table 1). We assume that PATDEX’s case base consists of two cases, namely CaseCorrect (left column of Table 1) and CaseIncorrect (middle column of Table 1), and both cases consist of a number attributes with attribute values (for a diagnostic problem solving situation also called symptom and symptom value). Some of these attribute values are not known (“?”), because they have not been recorded for this case (e.g., because they have been considered as irrelevant). As can be derived from the respective names, CaseCorrect reflects the underlying fault situation, that is, for this scenario it is assumed that the values included in CaseCorrect are correct and that the correct diagnosis is “IOCardDefect”.

CaseCorrect	**CaseIncorrect**	**Query**
ErrorCode = i59	ErrorCode = i59	ErrorCode = i59
I/OStateOut7 = logical-1	?	?
Valve5Y1 = switched	Valve5Y1 = not-switched	?
I/OStateOut24 = logical-0	?	?
Valve5Y2 = not-switched	?	Valve5Y2 = not-switched
PipesClampingReleaseDevice = ok	?	?
I/OStateIn32 = logical-1	I/OStateIn32 = logical-1	I/OStateIn32 = logical-1
DIAGNOSIS = IOCardDefect	DIAGNOSIS = Magnetic-SwitchDefect	**?**
SIM(Query,CaseCorrect) = $\frac{1 \bullet 3}{1 \bullet 3 + 0.5 \bullet 4}$ = $\underline{\underline{0.6}}$	SIM(Query,CaseIncorrect) = $\frac{1 \bullet 2}{1 \bullet 2 + 0.5 \bullet 2}$ = $\underline{\underline{0.67}}$	

Table 1. CaseCorrect, CaseIncorrect, the Query case, and the respective similarities

We assume that the underlying similarity measure is as follows [33]:

$$\text{SIM}(\text{Case}_1,\text{Case}_2) = \frac{a \bullet card(E)}{a \bullet card(E) + b \bullet card(D) + c \bullet card(U_1) + d \bullet card(U_2)}$$

- E set of symptoms with the *same* values for Case_1 and Case_2
- D set of symptoms with *different* values for Case_1 and Case_2
- U_1 set of symptoms with *known values for Case*$_1$ but not for Case_2
- U_2 set of symptoms with *known values for Case*$_2$ but not for Case_1
- a=1, b=2, c=1/2, d=1/2 (the default weights used by PATDEX)
- card cardinality of sets

Given the above defined Query case, PATDEX comes up with the wrong diagnosis (because the similarity between CaseIncorrect and the Query case is higher than that between CaseCorrect and the Query case, see Table 1). The reason for this is that PATDEX uses its well-informed similarity measure based on Tversky's contrast model [180] and explicitly considers unknown symptom values. However, the first wrong judgment (Table 1) can be improved (Table 2) by the use of general domain knowledge, namely the application of causal or heuristic rules:

- Applying a causal rule for deriving additional symptom values based on the already known symptom values
- For instance, the following causal rule could be applied:
  IF ErrorCode=i59 & Valve5Y2=not-switched
  THEN I/OStateOut24=logical-0.

Table 2 shows the outcome of applying the causal rule. Now the similarity between the (automatically improved) Query-2 case and CaseCorrect is higher

(0.73) than the similarity between CaseIncorrect and Query-2 (0.57). Thus, the application of the rule adapts the result of PATDEX in the right way.

CaseCorrect	**CaseIncorrect**	**Query-2**[7]
ErrorCode = i59	ErrorCode = i59	ErrorCode = i59
I/OStateOut7 = logical-1	?	?
Valve5Y1 = switched	Valve5Y1 = not-switched	?
I/OStateOut24 = logical-0	?	I/OStateOut24 = logical-0
Valve5Y2 = not-switched	?	Valve5Y2 = not-switched
PipesClampingReleaseDevice = ok	?	?
I/OStateIn32 = logical-1	I/OStateIn32 = logical-1	I/OStateIn32 = logical-1
DIAGNOSIS = IOCardDefect	DIAGNOSIS = Magnetic-SwitchDefect	**?**
$\mathrm{SIM}(\mathrm{Query},\mathrm{CaseCorrect}) = \frac{1 \bullet 4}{1 \bullet 4 + 0.5 \bullet 3} = \underline{\underline{0.73}}$	$\mathrm{SIM}(\mathrm{Query},\mathrm{CaseIncorrect}) = \frac{1 \bullet 2}{1 \bullet 2 + 0.5 \bullet 3} = \underline{\underline{0.57}}$	

Table 2. Situation after applying rule R1

The above improvement is based on the availability of the causal (or heuristic) rules. An alternative improvement strategy (results in Table 3), which is not necessarily based on the availability of such knowledge, is the acquisition of further information using a complete test selection subcomponent:

- Applying test selection to extend and thereby improve the available information.
- For instance, the following test could be carried out:
  Test 5Y1: "What is the status of Valve5Y1?"
  Result of Test5Y1: "Valve5Y1=switched"

The reason to ask such a question is to carry out some kind of differential diagnosis as known from medical diagnostics, because CaseCorrect and CaseIncorrect have different values for this attribute. Table 3 shows that the outcome of knowing one attribute value more is that now the similarity between CaseCorrect and the Query-3 case is much higher than the similarity between CaseIncorrect and Query-3.

[7] "Query-2" is the improvement of "Query" that has been achieved by automatically applying the mentioned causal rule.

CaseCorrect	**CaseIncorrect**	**Query-3**[8]
ErrorCode = i59	ErrorCode = i59	ErrorCode = i59
I/OStateOut7 = logical-1	?	?
Valve5Y1 = switched	Valve5Y1 = not-switched	Valve5Y1 = switched
I/OStateOut24 = logical-0	?	?
Valve5Y2 = not-switched	?	Valve5Y2 = not-switched
PipesClampingReleaseDevice = ok	?	?
I/OStateIn32 = logical-1	I/OStateIn32 = logical-1	I/OStateIn32 = logical-1
DIAGNOSIS = IOCardDefect	DIAGNOSIS = MagneticSwitch-Defect	**?**
$\mathrm{SIM(Query,CaseCorrect)} = \frac{1 \bullet 4}{1 \bullet 4 + 0.5 \bullet 3} = \underline{\underline{0.73}}$	$\mathrm{SIM(Query,CaseIncorrect)} = \frac{1 \bullet 2}{1 \bullet 2 + 2 \bullet 1 + 0.5 \bullet 1} = \underline{\underline{0.44}}$	

Table 3. Situation after ascertaining status of Valve5Y2

Important problems that need to be handled in the CNC domain are system-guided and/or user-guided completion of available information, the handling of a huge number of attributes, classes, cases, and unknown values (>90%). An attribute-value based case representation using symbolic value ranges was sufficient for this application task. Tolerating the high number of unknown symptom values required a Tversky-like similarity measure (e.g., the one presented above). In addition, general domain knowledge is helpful (but not necessary) for supporting both the test selection and the similarity measure.

Overview on CBR Applications[9]

Real-life applications have shown that CBR is a practically important technology. Many researchers and practitioners view CBR technology as one of the most successful artificial intelligence technologies [Bartsch-Spörl et al. 1999). This is documented through many application oriented books that appeared during the last few years. Althoff [21] describes four applications carried out in the INRECA[10] project in the years 1992-1995:

- A help-desk on robot diagnosis,

[8] "Query-3" is an alternative improvement of "Query". Here one additional attribute value is entered by the maintenance technician (PATDEX has asked for that value).

[9] It is far beyond the scope of this chapter to give a complete overview on this. Instead, a few exemplary applications are presented.

[10] The INRECA project lasted from 1992-1995 and was funded by the Commission of the European Union (Esprit III P6322). The partners of INRECA were AcknoSoft (coordinating partner, France), tec:inno (Germany), IMS (Ireland), and the University of Kaiserslautern (Germany).

- a maintenance support system for troubleshooting jet engines,
- a system for assessing wind risk factors in Irish forests, and
- a research prototype system for supporting the reuse of object-oriented software.

Overviews on CBR applications in design [137], planning [183], and in diagnosis and decision support [124] appeared in a special issue of the AI Communications. More detailed descriptions are presented in Lenz et al. [1998a), which also describes application domains like software engineering, medicine, tutoring, and help systems. Bergmann et al. [52] briefly summarize more than a dozen applications carried out by the partners of the INRECA-2[11] consortium (1996-1999). The special issue on CBR of the International Journal of Engineering Applications of Artificial Intelligence [51] includes, among others, applications on image processing. While Stolpmann and Wess [173] focus on customer relationship management in detail, Wilke [196] and Cunningham [72] concentrate on electronic commerce applications. Althoff et al. [23] report on a number of early CBR applications on planning and design (including Lockheed's CLAVIER application on the configuration of autoclave loads, one of the first (1990) breakthrough applications for CBR technology like AcknoSoft's application on troubleshooting aircraft engines at CFM International in 1992), especially on classification like application tasks. Cheetham & Graf [69], Finnie et al. [79], Jarmulak [98], and Azuaje et al. [41] focus on specific classification and data interpretation tasks. Schulz [162] describes the use of CBR in corporate decision processes.

A very recent but popular application domain is knowledge management [88, 31, 13, 14, 77, 87, 85, 32, 178, 75]. See also the section on Case-Based Knowledge Management above.

Evaluation of CBR Systems and Applications

An important part of systematically developing software systems is the evaluation of developed applications, of different candidate software tools, and/or candidate development methodologies (Figure 13). In the CBR community the second kind of evaluation was treated first and in most detail. Based on the motivation of evaluating industrial CBR applications using commercial CBR tools, which was underlying the INRECA project, Traphöner et al. [179] presented a first criteria catalogue. Althoff et al. [25] started with a subset of these criteria, further extended and modified them, and refined them in a way that they could be

[11] The European Union funded INRECA-2 project lasted from 1996-1999. The partners of INRECA-2 were AcknoSoft (coordinating partner, France), tec:inno (Germany), IMS (Ireland), DaimlerChrysler (Germany), and the University of Kaiserslautern (Germany).

used for detailed comparison of five well-known commercial CBR tools[12]. A revised version of this was made available for a broader public [23].

Within the INRECA project the tool comparison was used to define the final architecture of the INRECA tool to be developed within the project. This tool was compared to commercial tools based on the same criteria [21]. In addition, a more general framework for evaluating CBR systems was established [22]. This was used to compare the INRECA system with many other CBR research systems. The key idea of this framework was to combine different kinds of criteria – technical and ergonomic criteria, and domain and task criteria – to relate "CBR tools" and "CBR applications" with one another (Figure 14). For example, describing a CBR application with technical criteria, namely those to be fulfilled if the application should be developed, enables us to compare the application with different CBR tools. That is, to decide which is the most appropriate (e.g., in the sense of most similar) CBR tool.

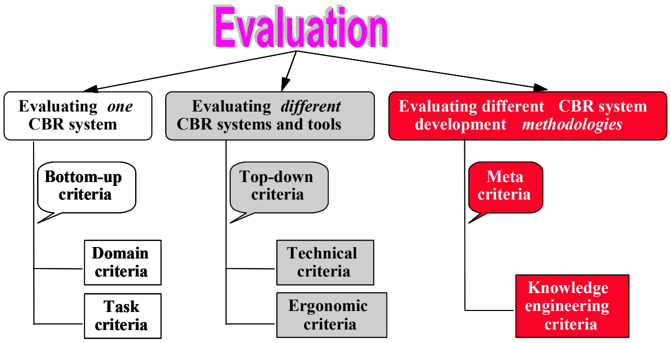

Evaluation
Evaluating *one* CBR system
Evaluating *different* CBR systems and tools
Evaluating different CBR system development *methodologies*
Bottom-up criteria
Top-down criteria
Meta criteria
Domain criteria
Task criteria
Technical criteria
Ergonomic criteria
Knowledge engineering criteria

Figure 13. Different views on evaluating CBR systems

Of course, this works also the other way round: Associating CBR tools with domain and application task criteria, to decide which tool can be applied for which application domain and/or task. In practice such associated criteria are not very detailed. However, they are already valuable on a more general level. For instance, such criteria were used for an implementation of a case-based information system on CBR tools and applications called CBR-PEB (CBR Product Experience Base[13]; [26, 46, 36].

[12] CBR Express, Esteem, Kate tools, Remind, S3-Case (1994 versions).

[13] CBR-PEB is available via WWW: http://demolab.iese.fhg.de:8080/.

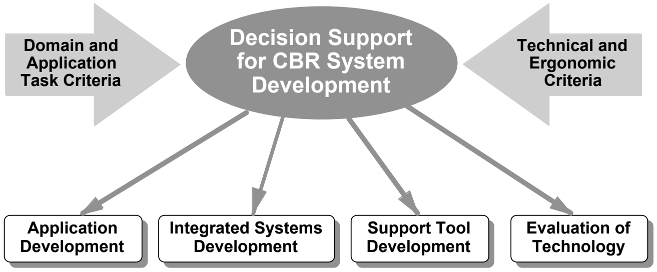

Domain and Application Task Criteria
Decision Support for CBR System Development
Technical and Ergonomic Criteria
Application Development
Integrated Systems Development
Support Tool Development
Evaluation of Technology

Figure 14. Decision support criteria for CBR system development

The criteria set described in Althoff [21] is not only helpful for comparing tools and defining goals for new tool development, but it also supports the evaluation of CBR technology in a more general sense, as for instance posed by Cohen [70], and systematic application development. Based on this, two different methodologies for building CBR applications have been developed (see next section on CBR Application Development).

In the scope of building one of these methodologies, it turned out that the *goal-question-metric* paradigm [49, 170, 156] known from experimental software engineering, is a good candidate technology for evaluating one (single) CBR application. Experiences were gathered during the development (and evaluation) of the CBR-PEB application [135]. Currently the combination of evaluation with maintenance is an interesting topic for research ([134], see Section on Usage of Case-Based Reasoning Technology).

An in-depth description of all important evaluation issues of CBR technology will appear in Althoff & Nick [35].

CBR Application Development

On the basis of the process model described Figure 1 it is already possible to identify the key issues in building a useful and usable CBR system [26]:

- First of all, it is necessary to have or to be able to acquire enough[14] cases to achieve a representative coverage of the application domain.
- Next, it is essential to represent the cases in a way that is storable in a computer memory, accessible by a software program and capable to capture the true meaning of a case.

[14] Leake & Wilson [118] point out that unlike a rule-based system, which needs to cover the space of *possible* domain problems, CBR systems only need cases for a representative sample of the problems that actually arise. This may be a small subset of the full application domain, which can make the task much easier (if no user ever asks a particular question, the system won't need a case for answering that question).

- In order to do efficient case retrieval, it is important to have or to develop an indexing procedure that works automatically and brings back a certain number of similar cases in a rather short time.
- The program that assesses the similarity between the actual and the stored cases' problem description has to be both precise and robust enough to enable the transfer of solutions without too much risk.
- If the domain is of a kind where it makes sense to adapt suggested solutions to the actual situation then another key issue is the ability of the CBR system to carry out the necessary adaptation steps.
- Last but not least, CBR systems usually come together with organizational changes. Therefore, it is of major importance to achieve a proper integration of the system into mostly new or changed business processes and to get the CBR system accepted and properly used by all persons in an organization that are expected to benefit from its use.

Meanwhile a number of development strategies and methodologies of how to develop CBR systems have been presented. The DISER methodology [174, 176, 29) describes how to systematically develop experience factory applications based on CBR technology. DISER's methods for case base design include the definition of the objectives and the respective subject areas underlying the CBR system to be developed. In addition, all scenarios concerned with the case base have to be identified. Based on this information the case base conceptualization is detailed (Figure 15). When this information is available it can be defined how the recording of the cases shall work to fill the case base (Figure 16).

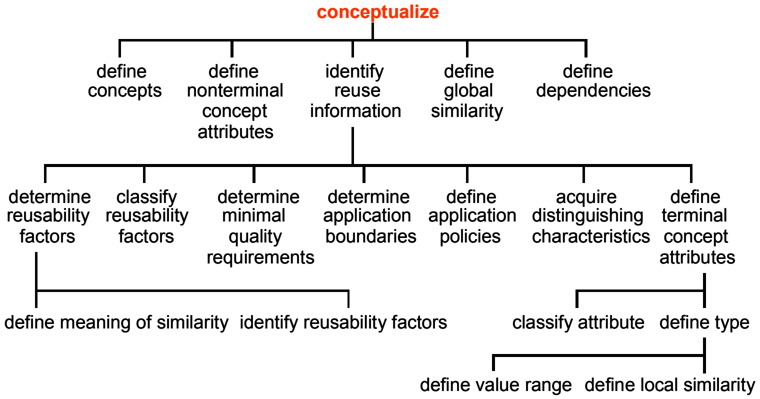

conceptualize
define concepts
define nonterminal concept attributes
identify reuse information
define global similarity
define dependencies
determine reusability factors
classify reusability factors
determine minimal quality requirements
determine application boundaries
define application policies
acquire distinguishing characteristics
define terminal concept attributes
define meaning of similarity
identify reusability factors
classify attribute
define type
define value range
define local similarity

Figure 15. Developing the conceptual structure for a CBR system

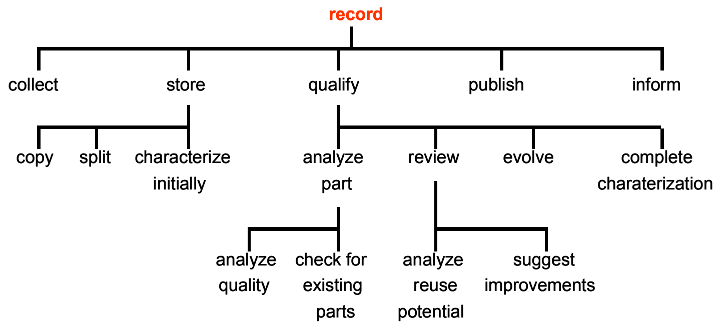
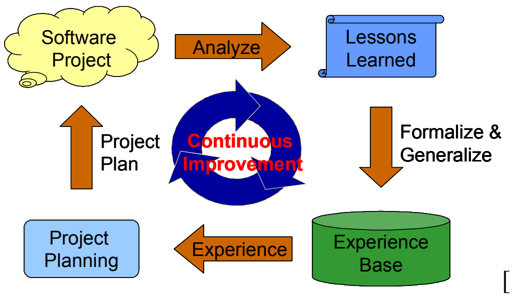


Figure 16. Case acquisition and case base development

The INRECA-2 methodology [52, 50] is based on the experiences collected by project partners. Here the basic organizational background is also an experience factory (Figure 17). Application development is supported on three different levels: a common generic level, a cookbook level, and a specific project level (Figure 18).



[1]

Figure 17. Experience Factory for developing CBR applications (adapted from [50])

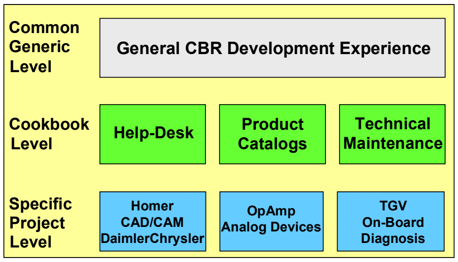


Figure 18. INRECA Experience Base (adapted from [50])

Case-Based Reasoning: Past, Present, and Future

CBR has its roots in Roger Schank's work on scripts [160] and dynamic memory [157]. The notion of *case-based reasoning* dates back to the beginning of the eighties [103, 157). The underlying motivation was to model human learning and problem solving behavior. In the sequel, several exemplary research prototypes were built that made the underlying ideas more concrete (some overviews can be found in [109]). Based on these early experiences, CBR became an interesting approach for several different reasons:

- CBR offers the possibility of modeling and analyzing human learning and problem solving behavior.
- CBR supports, in an intuitively reasonable and understandable way, the handling of application domains where the notion of a case is already naturally existing (law, medicine, etc.).
- CBR integrates learning and problem solving "per construction".

In Europe CBR research started a few years later (an overview can be found in [7]). Compared to the USA it has been more focused towards knowledge-based systems development, and to the integration of case-based and other approaches (e.g., [45, 33, 1, 171, 141, 2, 37]. Research efforts on CBR were considerably focused - especially in the USA - by the DARPA workshops in 1988, 1989, and 1991 [107, 89, 43]. A similar effect was achieved by comparable national and European events [39, 97, 193, 101].

From 1995 on - based on the successful experiences of the European Workshops on Case-Based Reasoning EWCBR in Otzenhausen/Kaiserslautern 1993 (organized by Michael Richter, Klaus-Dieter Althoff, and Stefan Wess) and Chantilly 1994 (organized by Mark Keane, Michel Manago, and Jean-Paul Haton) the North American and the European CBR communities decided to join their efforts and start a biennial International Conference on Case-Based Reasoning [182, 116, 27], alternating with EWCBR[15] [78, 167, 57]. This process of building a real international CBR community was additionally supported by the development of a number of web repositories and email lists [18, 67, 68, 96].

For many years, CBR has been a well established topic of many regularly held scientific conferences. In the meantime, a basis of knowledge, principles, methods, and algorithms has been achieved that most researchers seem to agree on. Thus, this can be seen as the core of the field. Besides the usual introductory articles [108, 38, 7, 114, 136, 47], introductory textbooks are available [153, 192, 109, 113, 184, 125, 173]. A detailed description of several interesting early systems is given in [192], being well completed by those described in Kolodner [109], and Aamodt & Plaza [7]. The books also describe several interesting systems and approaches.

From a technological point of view, CBR has achieved a certain degree of maturity. This is documented by both the applications that have been carried out up to now (see section on CBR Applications) and the commercial tools currently available on the market. In 1991, the first CBR tools were offered. Though the market was completely dominated by American products at that time, the situation has been considerably changed by the introduction of new European CBR

[15] Starting in 2002, the European Workshop on CBR will turned into a conference.

tools. An overview of early commercial tools is given by Harmon [91]. More recently, a detailed evaluation of a selection of the most important CBR tools was carried out [23]. Currently - October 2000, more than two dozen CBR tools exist.

CBR has started to play a role in knowledge and software engineering (as can be seen by this book chapter), business informatics [162], and knowledge management. One reason here is the enormous amount of applications available [173] especially on help-desk and service support applications. And CBR will play an important role in e-commerce, customer-relationship management, and corporate knowledge management. To embed CBR technology in the business processes of various organizations, it appears to be very beneficial to view and run CBR applications as experience factories [99, 44, 30, 174]. This creates a direct link between CBR and software and knowledge engineering, for instance with topics like maintenance and meta-knowledge [129]. In addition, roles, tasks, and methods (e.g., goal-oriented measurement and evaluation) available from experience factory can be used for running real-life applications on knowledge management.

While CBR can be viewed as an approach for putting (some kind of) knowledge-based systems into practice either as implementation or as plug-in technology, the relationship to another artificial intelligence subfield will become important soon: Machine Learning[16]. One interesting issue here is the use of CBR systems as an intelligent platform for machine learning procedures, for instance, learning of retrieval structures (e.g., based on decision trees: [61, 21]; Kohonen networks: [76]; index learning: [89, 81, 62, 146, 121, 198], learning of similarity assessments procedures (e.g., adaptation of importance degrees of case features: [133, 195], learning of case structure/contents from databases and/or textual descriptions (e.g., textual CBR, text mining: [121]), or learning of user-adapted access structures (n-step CBR: [188], [152]).

In addition, machine learning techniques for knowledge discovery in case bases will probably be used more and more to support systematic knowledge development, a task of increasing importance for knowledge management applications and the running of lessons-learned systems. The back of knowledge development is the task of knowledge maintenance. Both tasks can be supported with goal-oriented measurement and evaluation [134]. So, techniques from experimental software engineering, machine learning, and CBR should be combined to strengthen the knowledge management method spectrum. Here the positive experience with using CBR as a plug-in technology can be reused. In addition, knowledge engineering approaches could be integrated here for knowledge elicitation and modeling.

A big challenge for CBR, and also for other information technologies, will be the possibility to represent interrelated case bases of different granularities and subject areas within, for example, an internationally operating company or a group of related companies that use the same knowledge. Here a huge knowledge

[16] This supplements CBR being already a machine learning method, namely through case learning.

network has to be built that can be accessed from various points of view, on different levels of abstraction, and on related subject areas.

An advantage of CBR could be to offer an "open environment" for integrating different kinds of approaches. This already happened very often in the past and could be a key issue to develop integrated solutions for complicated real-life problems [4, 21, 16]. For example, experience reuse is a very general task that can definitely be supported by CBR and that is of importance for nearly every application area and research field.

One example of a mutual benefit of integrating CBR with other technologies has already been mentioned above: Combining CBR with the experience factory (EF) approach [175, 174]. While CBR can offer a technology for implementing the experience repository included in an experience factory, it can in addition be used as a strategy on an organizational level [31]. Vice versa, an experience factory offers a physical and/or logical infrastructure for operationalizing a CBR system in an organization's environment and technologies like goal-oriented measurement to evaluate CBR applications (Figure 19).

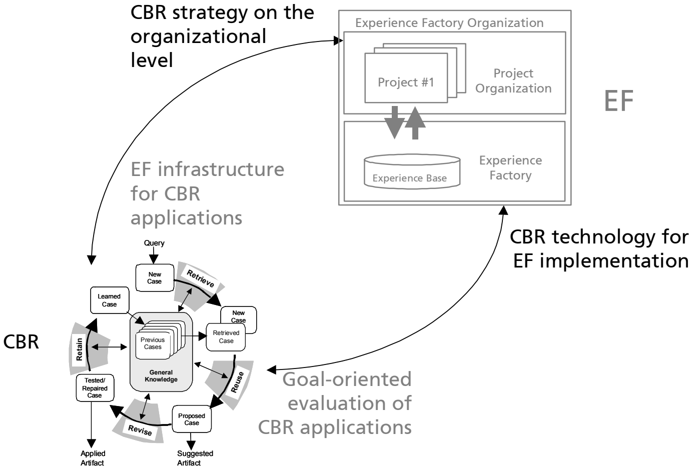


Figure 19. Mutual benefits of integrating CBR with other technologies: the experience factory example

What has to happen or to be guaranteed is that CBR remains a research field that brings up new, innovative research methods and tools helpful for information technology or - even broader - the information society?

- The collaboration and scientific communication between humanities oriented and engineering oriented research groups/practitioners within the CBR community should be preserved and/or revitalized.
- CBR should remain a community that is open for people from outside. By this, integrations as shown in Figure 19 should be possible with various communities, among others knowledge management, knowledge engineering, software engineering, machine learning, statistics, information re-

trieval, decision theory, pattern recognition, neural networks, and fuzzy technologies.

- It should be possible to use CBR more and more as a plug-in technology [47].
- The current advantage of being a community that integrates "real-life applications" and "basic research" should be preserved. It is one very characteristic feature of the CBR community that has worked well during the past years. It also provides integration with other technologies will take place (through applications) and become an integrated part of the community (through basic research).

Acknowledgment

The ideas and concepts presented here have been influenced by many persons. I am especially grateful - among others - to Agnar Aamodt, Brigitte Bartsch-Spörl, Ralph Bergmann, Christiane Gresse von Wangenheim, Markus Nick, Michael M. Richter, Dieter Rombach, Ralph Traphöner, Carsten Tautz, and Stefan Wess. Thanks go also to David Aha, Matthias Brandt, Brigitte Bartsch-Spörl, Ralph Bergmann, Hans-Dieter Burkhard, Padraig Cunningham, Werner Dubitzky, Christiane Gresse von Wangenheim, David Leake, Markus Nick, Enric Plaza, Michael M. Richter, Barry Smyth, Carsten Tautz, Ian Watson, and Gerhard Weber as well as the anonymous reviewers for commenting on an earlier version of this chapter.

References

1. Aamodt, A. (1989). Towards robust expert systems that learn from experience - an architectural framework. In: J. Boose, B. Gaines & J.-G. Ganascia (1989), *Proc. EKAW-89*, 311-326.
2. Aamodt, A. (1991). A Knowledge-Intensive, Integrated Approach to Problem Solving and Sustained Learning. *Ph.D. Thesis*, University of Trondheim.
3. Aamodt, A. (1993). A Case-Based Answer to Some Problems of Knowledge-based Systems. In: Sandewall, E. and Jansson, G. (eds.), *Proceedings Scandinavian Conference on Artificial Intelligence*, IOS Press.
4. Aamodt, A. (1994). Explanation-Based Case-Based Reasoning. In: *Wess, Althoff & Richter, (1994)*, 274-288.
5. Aamodt, A. (1995). Knowledge Acquisition and learning by experience the role of case-specific knowledge. In: *Kodratoff and Tecuci (1995),* 197-245.
6. Aamodt, A. and Nygård, M. (1995). Different roles and mutual dependencies of data, information, and knowledge - an AI perspective on their integration. *Data and Knowledge Enigneering 16 (1995), 191-222*.
7. Aamodt, A. and Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches. *AI Communications Vol. 7*, *No. 1*, 39-59.

8. Abecker, A., Bernardi, A. & Sintek, M. (2000). Proactive Knowledge Delivery for Enterprise Knowledge Management. In: G. Ruhe & F. Bomarius (eds.), *Learning Software Organizations - Methodology and Applications*. Springer-Verlag, Lecture Notes in Computer Science.
9. Aha, D. W. (1991). Case-Based Learning Algorithms. In: Bareiss (1991), 147-158.
10. Aha, D. W. (ed.) (1994). *Proceedings of the Case-Based Reasoning Workshop*. AAAI-1994, AAAI Technical Report.
11. Aha, D. W. (1997). Editorial on Lazy Learning. *Artificial Intelligence Review, 11,* 7-10.
12. Aha, D. W. (1998). The Omnipresence of Case-Based Reasoning in Science and Application. *Knowledge-Based Systems, 11(5-6),* 261-273.
13. Aha, D.W. (1999). The AAAI-99 KM/CBR Workshop: Summary of Contributions. *In Gresse von Wangenheim & Tautz (1999)*, II-37–II-44.
14. Aha, D.W., Becerra-Fernandez, I., Maurer, F., and Muñoz-Avila, H. (Eds.) (1999). *Exploring Synergies of Knowledge Management and Case-Based Reasoning: Papers from the AAAI 1999 Workshop* (Technical Report WS-99-10). Menlo Park, CA: AAAI Press.
15. Aha, D. W., Breslow, L. A. & Muñoz-Avila, H. (2000). Conversational case-based reasoning. *To appear in Applied Intelligence*.
16. Aha, D. W. & Daniels, J. (eds.)(1998). *Case-Based Reasoning Integrations: Papers from the AAAI 1998 Workshop.*
17. Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning, 6, 37-66.*
18. AI-CBR: http://www.ai-cbr.org/.
19. Althoff, K.-D. (1992). *Eine fallbasierte Lernkomponente als integrierter Bestandteil der MOLTKE-Werkbank zur Diagnose technischer Systeme*. Doctoral Dissertation, University of Kaiserslautern; also: Sankt Augustin, Germany: infix Verlag.
20. Althoff, K.-D. (1995). Evaluating Case-Based Reasoning Systems. In: Proc. Workshop on *Case-Based Reasoning: A New Force In Advanced Systems Development*, S. 48-61, published by Unicom Seminars Ltd., Brunel Science Park, Cleveland Road, Uxbridge, Middlesex UB8 3P, UK.
21. Althoff, K.-D. (1997). *Evaluating Case-Based Reasoning Systems: The INRECA Case Study*. Habilitationsschrift (Postdoctoral Thesis), Department of Computer Science, University of Kaiserslautern, July 9, 1997.
22. Althoff, K.-D. & Aamodt, A. (1996). Relating case-based problem solving and learning methods to task and domain characteristics: towards an analytic framework. *AI Communications, 9 (3)*, 1-8.
23. Althoff, K.-D., Auriol, E., Barletta, R. and Manago, M. (1995a). *A Review of Industrial Case-Based Reasoning Tools*. AI Intelligence, Oxford, UK.
24. Althoff, K.-D., Auriol, E., Bergmann, R., Breen, S., Dittrich, S., Johnston, R., Manago, M., Traphöner, R. & Wess, S. (1995b). Case-Based Reasoning for Decision Support and Diagnostic Problem Solving: The INRECA Approach. In: B. Bartsch-Spörl, D. Janetzko & S. Wess (eds.), XPS-95-Workshop: Fallbasiertes Schließen - Grundlagen und Anwendungen (3rd German Workshop on CBR: GWCBR-99), LSA-REPORT 95-02, Centre for Learning Systems and Applications, Dept. of Computer Science, University of Kaiserslautern, Germany.
25. Althoff, K.-D., Auriol, E., Holz, H., Manago, M., Meissonnier, A., Priebisch, C., Wess, S. & Wilke, W. (1994). *An Evaluation of Commercial Case-Based Reasoning Tools.* INRECA (Esprit P6322), Deliverable D5.

26. Althoff, K.-D. & Bartsch-Spörl, B. (1996). Decision Support for Case-Based Applications. In: *Wirtschaftsinformatik 1/96*, special issue on case-based decision support (edited by D. Ehrenberg), 8-16.
27. Althoff, K.-D., Bergmann, R. & Branting, L. K. (eds.) (1999a). *Case-Based Reasoning Research and Development*. Proc. of the Third International Conference on Case-Based Reasoning (ICCBR'99), Seeon Monastery, Germany, July 27-30, Springer Verlag, LNAI 1650.
28. Althoff, K.-D., Birk, A., Gresse von Wangenheim & Tautz, C. (1998a). Case-Based Reasoning for Experimental Software Engineering. In: Lenz et al. (1998a), 235-254.
29. Althoff, K.-D., Birk, A., Hartkopf, S., Müller, W., Nick, M., Surmann, D. & Tautz, C. (1999c). Managing Software Engineering Experience for Comprehensive Reuse. In Proc. of the 11th International Conference on Software Engineering and KnowledgeEngineering (SEKE'99).
30. Althoff, K.-D., Bomarius, F., Müller, W. & Nick, M. (1999d). Using a Case-Based Reasoning for Supporting Continuous Improvement Processes. In: P. Perner (ed.), Proc. German Workshop on Machine Learning, Technical Report, Institute for Image Processing and Applied Informatics, Leipzig, 8 pages.
31. Althoff, K.-D., Bomarius, F. & Tautz, C. (1998b). Using Case-Based Reasoning Technology to Build Learning Software Organizations. Proc. 13th Biennial European Conference on Artificial Intelligence (ECAI98) Workshop on "Building, maintaining, and using organizational memories" (OM-98).
32. Althoff, K.-D., Bomarius, F. & Tautz, C. (2000a). Knowledge Management for Building Learning Software Organizations. *Accepted for the Information System Frontiers Journal*.
33. Althoff, K.-D., Kockskämper, S., Maurer, F., Stadler, M. and Wess, S. (1989). Ein System zur fallbasierten Wissensverarbeitung in technischen Diagnosesituationen. *In: Retti, J. and Leidlmeier, K. (eds.), 5th Austrian Artificial-Intelligence-Conference*, 65-70, Springer Verlag.
34. Althoff, K.-D., Müller, W., Nick, M. & Snoek, B. (2000b). KM-PEB: An Online Experience Base on Knowledge Management Technology. *To appear in E. Blanzieri & L. Portinale (eds.), Advances in Case-Based Reasoning – Proc. of the 5th European Workshop on Case-Based Reasoning (EWCBR'00)*.
35. Althoff, K.-D. & Nick, M. (2001). *Evaluating Case-Based Reasoning Systems. Springer Verlag (to appear)*.
36. Althoff, K.-D., Nick, M. & Tautz, C. (1999b). CBR-PEB: A Tool for Implementing Reuse Concepts of the Experience Factory for CBR-System Development. In Proc. 5th German Conference on Knowledge-Based Systems (XPS'99) Workshop on Case-Based Reasoning (GWCBR'99); CBR-PEB is publicly accessible via http://demolab.iese.fhg.de:8080/.
37. Althoff, K.-D. & Wess, S. (1991). Case-Based Knowledge Acquisition, Learning and Problem Solving in Diagnostic Real World Tasks. Proc. EKAW-91, Glasgow & Crieff (edited by D. Smeed); also published in: M. Linster & B. Gaines (eds.), Proc. of the European Knowledge Acquisition Workshop (EKAW'91), GMD-Studien Nr. 211, Sept. 1992, 48-67.
38. Althoff, K.-D. & Wess, S. (1992). Case-Based Reasoning and Expert System Development. In: F. Schmalhofer, G. Strube & T. Wetter (eds.), *Contemporary Knowledge Engineering and Cognition,* Springer Verlag, 146-158.
39. Althoff, K.-D., Wess, S., Bartsch-Spörl, B. and Janetzko, D. (eds.) (1992). Ähnlichkeit von Fällen beim fallbasierten Schließen. *Proc. of the 1st Meeting of the German Working Group on Case-Based Reasoning*, University of Kaiserslautern, June 1992.

40. Althoff, K.-D. & Wilke, W. (1997). Potential Uses of Case-Based Reasoning in Experience Based Construction of Software Systems and Business Process Support. In: R. Bergmann & W. Wilke (eds*.), Proc. of the Fifth German Workshop on Case-Based Reasoning*, Centre for Learning Systems and Applications, University of Kaiserslautern, LSA-97-01E, 31-38.
41. Azuaje F., Dubitzky W., Black, N. & Adamson K. (1999). Improving Clinical Decision Support Through Case-Based Fusion. *IEEE Transactions on Biomedical Engineering, Special Issue on Biomedical Data Fusion, 46 (10)*, 1181-1185.
42. Azuaje F., Dubitzky W., Black, N. & Adamson K. (2000). Case retrieval strategies for CBR: a categorised bibliography. *The Knowledge Engineering Review (in press)*.
43. Bareiss, R. (ed.) (1991). *Proc. 3rd DARPA Workshop on Case-Based Reasoning*, Morgan Kaufmann.
44. Bartlmae, K. (1999). An Experience Factory Approach for Data Mining. In Proc. LWA-1999, University of Magdeburg, Germany, 5-14.
45. Bartsch-Spörl, B. (1987). Ansätze zur Behandlung von fallorientiertem Erfahrungswissen in Expertensystemen. *KI, 4*, 32-36.
46. Bartsch-Spörl, B., Althoff, K.-D. & Meissonnier, A. (1997). Learning From and Reasoning About Case-Based Reasoning Systems. In: P. Mertens & H. Voß (eds.), *Expertensystem 97 (XPS-97)* - Proc. of the 4th German Conference on Knowledge-Based Systems, infix Verlag.
47. Bartsch-Spörl, B., Lenz, M. & Hübner, A. (1999). Case-Based Reasoning - Survey and Future Directions. In: F. Puppe (ed.), XPS-99*: Knowledge-Based Systems - Survey and Future Directions*. Proc. of the 5th German Biennial Conference on Knowledge-Based Systems, Springer Verlag, 67-89.
48. Basili, V.R., Caldiera, G. & Rombach, D. (1994a). Experience Factory. In Marciniak, J.J. (ed.)*, Encyclopedia of Software Engineering*, vol 1, 469–476. John Wiley & Sons.
49. Basili, V.R., Caldiera, G. & Rombach, D. (1994b). Goal-Question-Metric Paradigm. In Marciniak, J.J. (ed.)*, Encyclopedia of Software Engineering*, vol 1. John Wiley & Sons.
50. Bergmann, R. (1999a). Developing Industrial Case-Based Reasoning Applications Using the INRECA Methodology. *Slide copies, invited talk, IJCAI 1999 Workshop on Automating the Construction of Case-Based Reasoners*.
51. Bergmann, R. (1999b). Preface to the special issue on "Engineering Applications of Case-Based Reasoning. Special Issue of the International Journal "Engineering Applications of Artificial Intelligence", Vol. 12 (6), 661-663, Elsevier.
52. Bergmann, R., Breen, S., Göker, M., Manago, M. & Wess, S. (1999). *Developing Industrial CBR Applications. The INRECA-Methodology*. Springer Verlag, LNAI 1612.
53. Bergmann, R., Muñoz-Avila, H., Veloso, M., Melis, E. (1998). Case-based reasoning applied to planning tasks. In: *Lenz et al. (1998a)*, 169-200.
54. Bergmann, R., Pews, G. and Wilke, W. (1994). Explanation-Based Similarity: A Unifying Approach for Integrating Domain Knowledge into Case-Based Reasoning for Diagnosis and Planning Tasks. In: *Wess, Althoff & Richter (1994)*, 182-196.
55. Bergmann, R. & Stahl, A. (1998). Similarity Measures for Object-Oriented Case Representations. In: *Smyth & Cunningham (1998)*, 25-36.
56. Bergmann, R., Wilke, W., Vollrath, I., Wess, S. (1996). Integrating General Knowledge with Object-Oriented Case Representation and Reasoning. *Proc. 4th German Workshop on Case-Based Reasoning (GWCBR-96)*.
57. Blanzieri, E. & Portinale, L. (eds.) (2000). *Advances in Case-Based Reasoning: Proc. of the Fifth European Workshop on Case-Based Reasoning*, Springer Verlag.

58. Börner, K. (1998). CBR for Design. In: *Lenz et al. (1998a),* 201-234.
59. Bomarius, F., Althoff, K.-D. & Müller, W. (1998). Knowledge Management for Learning Software Organizations. *Software Process - Improvement and Practice 4,* 89-93 (1998) (printed 1999).
60. Brandt, M. (2000). *Fallbasierte Aufwandsschätzung von Softwareentwicklungsprojekten*. Technical Report No. 33, Institue for Business Informatics, University of Leipzig, Germany.
61. Breslow, L. A. & Aha, D. W. (1997). Simplifying decision trees: A survey. *Knowledge Engineering Review, 1*, 1-40.
62. Burkhard, H.-D. (1995). *Case Retrieval Nets*. Technical Report, Humboldt University Berlin.
63. Burkhard, H.-D. (1998). Extending some concepts of CBR - Foundations of case retrieval nets. In *Lenz et al. (1998a*), 17-50.
64. Burkhard, H.-D. & Richter, M. M. (2000). On the Notion of Similarity in Case-Based Reasoning and Fuzzy Theory. In: *Pal et al. (2000)*.
65. Carbonell, J. G. (eds.) (1989). Special Volume on Machine Learning. *Artificial Intelligence,* 40.
66. Carrick, C. & Yang, Q. (1999). Activating CBR Systems through Autonomous Information Gathering. *Proc. Internat. Joint Conf. on Artificial Intelligence 1999 (IJCAI-99)*.
67. CBR-Web: http://www.cbr-web.org/.
68. CBR-PEB: http://demolab.iese.fhg.de:8080/.
69. Cheetham, W. & Graf, J. (1997). Case-Based Reasoning in Color Matching. In: *Leake & Plaza (1997)*, 1-12.
70. Cohen, P. R. (1989). Evaluation and Case-based Reasoning. In: *Hammond (1989b)*, 168-172.
71. Craw, S., Wiratunga, N. & Rowe, R. (1998). Case-Based Design for Tablet Formulation. In: *Smyth & Cunningham (1998)*, 358-369.
72. Cunningham, P. (1999). *Intelligent Systems in Electronic Commerce*. Keynote speech at the Third International Conference on Case-Based reasoning (ICCBR-1999).
73. Cunningham, P. & Smyth, B. (1994). A Comparison of model-based and incremental case-based approaches to electronic fault diagnosis. In D. W. Aha (ed.), *Proceedings of the Case-Based Reasoning Workshop at AAAI-1994.*
74. De la Ossa, A. (1992). Knowledge Adaptation: Analogical Transfer of Empirical Knowledge Across Domains for Case-Based Problem Solving. Doctoral Dissertation, University of Kaiserslautern.
75. Decker, S. & Staab, S. (2000). *AI Techniques for Knowledge Management*. ECAI-2000 tutorial (http://kmtutorial.aifb.uni-karlsruhe.de/).
76. Dubitzky, W. & Azuaje, F. (2000). A Genetic Algorithm and a Growing Cell Structures Approach to Learning Case Retrieval Structures. *In: Pal et al. (2000)*, 115-146.
77. Dubitzky, W., Büchner, A. & Azuaje, F. (1999). Viewing knowledge management as a case-based application. *Proceedings of the AAAI'99 Workshop on Knowledge Management and CBR Synergy*, AAAI Press, Technical Report WS-99-10, pp. 23-27, 1999.
78. Faltings, B. & Smith, I. (eds.)(1996). *Advances in Case-Based Reasoning: Proc. of the Third European Workshop on Case-Based Reasoning*. Springer Verlag.
79. Finnie, G. R., Wittig, G. E. & Desharnais, J.-M. (1997). Estimating software development effort with case-based reasoning. In *Leake & Plaza (1997)*.
80. Fox, S. E. (2000). A Unified CBR Architecture for Robot Navigation. In: *Blanzieri & Portinale (2000),* 406-417.

81. Fox, S. E. & Leake, D. B. (1995). Modeling Case-Based Planning for Repairing Reasoning Failures. In Proceedings of the 1995 AAAI Spring Symposium on Representing Mental States and Mechanisms, AAAI Press, 31-38.
82. Funk , P., Minor, M., Roth-Berghofer, T. & Wilson, D. (eds.) (2000*). Flexible Strategies for Maintaining Knowledge Containers*. Proc. of a Workshop at the 14th European Conference on Artificial Intelligence (ECAI 2000).
83. Gick, M. L. & Holyoak, K. J. (1980). Analogical Problem Solving. *Cognitive Psychology, 12*, 306-355.
84. Globig, C. & Lange, S. (1994). On Case-Based Representability and Learnability of Languages. In: S. Arikawa & K. P. Jantke (eds.), *Proc. 4th Internat. Workshop on Analogical and Inductive Inference (AII'94),* LNAI 872, Springer Verlag,106-121.
85. Göker, M. & Roth-Berghofer, T. (eds.) (1999). Workshop on "The Integration of Case-Based Reasoning in Business Processes". In *Schmidt & Vollrath (1999)*.
86. Gresse von Wangenheim, C., Althoff, K.-D. & Barcia (1999). Intelligent Retrieval of Software Engineering Experienceware. In Proc. of the 11th International Conference on Software Engineering and Knowledge Engineering (SEKE'99).
87. Gresse von Wangenheim, C. & Tautz, C. (eds.) (1999). Workshop on "Practical Case-Based Reasoning Strategies for Building and Maintaining Corporate Memories". In *Schmidt & Vollrath (1999)*.
88. Gupta, K. M. (1997). Case Base Engineering for Large Scale Industrial Applications. In *Proceedings of AAAI Spring Symposium on Artificial Intelligence in Knowledge Management* (http://ksi.cpsc.ucalgary.ca/AIKM97/AIKM97Proc.html).
89. Hammond, K. (1989a). *Case-Based Planning*. London: Academic Press.
90. Hammond, K. (ed.) (1989b). *Proc. of the 2nd Darpa Workshop on Case-Based Reasoning*. Holiday Inn, Pensacola Beach: Morgan Kaufmann.
91. Harmon, P. (1992). Overview: Commercial Case-Based Reasoning Products. *Intelligent Software Strategies*, January 1992.
92. Heister, F. & Wilke, W. (1998). An architecture for maintaining case-based reasoning systems. In *Smyth & Cunningham (1998)*.
93. Hendler, J. (1995). Experimental AI systems. In: *Journal of Experimental and Theoretical AI 7*, 1-5.
94. Henninger, S. (1995). Developing Domain Knowledge Through the Reuse of Project Experiences. *Proc. Symposium on Software Reusability (SSR '95), Seattle WA, April 1995.*
95. Henninger, S. (1997). Capturing and Formalizing Best Practices in a Software Development Organization. *In Proceedings of the 9th Int. Conference on Software Engineering & Knowledge Engineering (SEKE'97).*
96. ICCBR: http://www.iccbr.org/.
97. Janetzko, D. and Schult, T. (eds.) (1993). *Fälle in hybriden Systemen*. Proceedings of the Second German Workshop on CBR, University of Freiburg, Germany.
98. Jarmulak, J. (1999). *Case-Based Reasoning for NDT Data Interpretation*. PhD Thesis, Delft University of Technology, The Netherlands.
99. Kalfoglou, Y & Robertson, D. (2000). Applying Experienceware to Support Ontology Deployment . In Proceedings of the International Conference on Software Eng. and Knowledge Eng.
100. Kamp, G., Lange, S. & Globig, C. (1998). Related Areas. In: Lenz et al. (1998a), 327-352.
101. Keane, M., Haton, J.-P. & Manago, M. (1995). *Advances in Case-Based Reasoning: Proc. of the Second European Workshop on Case-Based Reasoning*. Springer Verlag.
102. Kitano, H. & Shimazu, H. (1996). The experience sharing architecture: a case study in corporate-wide case-based software quality control. In: *Leake (1996a)*.

103. Kolodner, J. L. (1980). Retrieval and organizational strategies in conceptual memory: A computer model. *Ph.D. Thesis*, Yale University.
104. Kolodner, J. L. (1983a). Maintaining Organization in a Dynamic Long-Term Memory. *Cognitive Science,* 7, 243-280.
105. Kolodner, J. L. (1983b). Reconstructive Memory: A Computer Model. *Cognitive Science,* 7, 281-328.
106. Kolodner, J. L. (1987). Extending Problem Solver Capabilities Through Case-Based Inference. *Proc. of the 4th International Workshop on Machine Learning,* 167-178.
107. Kolodner, J. L. (ed.) (1988). *Proc. of the 1st Darpa Workshop on Case-Based Reasoning*. Holiday Inn, Clearwater Beach: Morgan Kaufmann.
108. Kolodner, J. L. (1991). Improving Human Decision Making Through case-Based Decision Aiding. *AI Magazine, 91(2)*, 52-68.
109. Kolodner, J. L. (1993). *Case-Based Reasoning*. Morgan Kaufmann.
110. Kolodner, J. L., Simpson, R. L. & Sycara, K. P. (1985). A process model of case-based reasoning in problem solving. *Proc. IJCAI-85*, 284-290. Los Angeles, CA: Morgan Kaufmann.
111. Koton, P. (1988). Reasoning about evidence in causal explanations. *Proc. AAAI-88*, 256-261.
112. Lam, W. & Shankararaman, V. (1998). Managing Change During Software Development: An Incremental, Knowledge-Based Approach. In Proceedings of 10th Int. Conference on Software Engineering and Knowledge Engineering, San Francisco Bay, California, 1998.
113. Leake, D. B. (ed.) (1996a). *Case-based Reasoning - Experiences, Lessons & Future Directions*. AAAI Press.
114. Leake, D. B. (1996b). CBR in Context: The present and the Future. In: *Leake (1996a).*
115. Leake, D. B. (1998). Cognition as Case-Based Reasoning. In: *The Blackwell Companion to Cognitive Science*.
116. Leake, D. B. & Plaza, E. (eds.) (1997). *Case-Based Reasoning Research and Development*. Proc. of the Second International Conference on Case-Based Reasoning (ICCBR'97), Springer Verlag.
117. Leake, D. B. & Wilson, D. C. (1998). Categorizing case-base maintenance: Dimensions and directions. In *Smyth & Cunningham (1998)*, 196-207.
118. Leake, D. B. & Wilson, D. C. (2000). Remembering Why to Remember: Performance-Guided Case-Base Maintenance. In *Blanzieri & Portinale (2000).*
119. Lees, B., Hamza, M. & Irgens, C. (1997). Applying Case-Based Reasoning to Software Quality Management. In *Leake & Plaza (1997).*
120. Lees, B. (ed.) (1999). Workshop on "Hybrid Case-Based Reasoning Systems". In *Schmidt & Vollrath (1999).*
121. Lenz, M. (2000). *Case Retrieval Nets as a Model for Building Flexible Information Systems*. Sankt Augustin, Germany: infix Verlag, DISKI 236.
122. Lenz, M., Auriol, E. & Manago, M. (1998c). Diagnosis and Decision Support. In: *Lenz et al. (1998a*), 51-90.
123. Lenz, M. & Burkhard, H.-D. (1996). Case Retrieval Nets: Basic Ideas and Extensions. In: *KI-96: Advances in Artificial Intelligence – Proc. 20th Annual German Conference on Artificial Intelligence*, LNAI 1137, Springer Verlag, 227-239.
124. Lenz, M. Burkhard, H.-D., Pirk, P., Auriol, E. & Manago, M. (1996). CBR for diagnosis and decision support. *Ai Communications 9(3), September 1996, 138-146.*
125. Lenz, M., Bartsch-Spörl, B., Burkhard, H.-D. & Wess, S. (eds.) (1998a). Case-based reasoning technology: from foundations to applications. Springer-Verlag, LNAI 1400.

126. Lenz, M., Hübner, A. & Kunze, M. (1998b). Textual CBR. In: *Lenz et al. (1998a).*
127. Maher, M. L., Balachandran, B. & Zhang, D.M. (1995). *Case-Based Reasoning in Design*. Lawrence Erlbaum Associates.
128. Manago, M., Althoff, K.-D., Auriol, E., Traphöner, R., Wess, S., Conruyt, N. & Maurer, F. (1993). Induction and reasoning from cases. In: M. M. Richter, S. Wess, K.-D. Althoff & F. Maurer (eds.), *Proc. First European Workshop on Case-Based Reasoning (EWCBR'93), SEKI-REPORT SR-93-12, Universität Kaiserslautern, 313-318.*
129. Menzies, T., Althoff, K.-D., Kalfoglou, Y. & Motta, E. (2000). Issues with Meta-Knowledge. *Accepted for the International Journal of Software Engineering and Knowledge Engineering (IJSEKE)*.
130. Minor, M. & Hanft, A. (2000). The Life Cycle of Test Cases in a CBR System. In: *Blanzieri & Portinale (2000)*, 455-466.
131. Motta, E. (2000). Knowledge Level Modelling. *In this volume*.
132. Muñoz-Avila, H., Hendler, J. A. & Aha, D. W. (1999). Conversational case-based planning. *Review of Applied Expert Systems, 5, 163-174.*
133. Muñoz-Avila, H. & Hüllen, J. (1996). Feature Weighting by Explaining Case-Base Planning Episodes. In *Faltings & Smith (1996)*, 280-294.
134. Nick, M. & Althoff, K.-D. (2000). Systematic Evaluation and Maintenance of Experience Bases. In *Funk et al. (2000)*.
135. Nick, M., Althoff, K.-D. & Tautz, C. (1999). Facilitating the Practical Evaluation of Organizational Memories Using the Goal-Question-Metric Technique. Proc. Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW'99), Banff, Oct. 16-21.
136. Ong, L. S. & Narasimhalu, A. D. (1998). Case-Based Reasoning. In J. Liebowitz (ed.), *The Handbook of Applied Expert Systems*, CRC Press, 11-1 - 11-16.
137. Oxman, R. and Voß, A. (1996). CBR in Design. *Ai Communications 9(3), September 1996, 117-127.*
138. Pal, S.K., Dillon, T.S., Yeung, D.S. (Eds.) (2000). *Soft Computing in Case-Based Reasoning*. London: Springer Verlag.
139. Papagni, M., Cirillo, V., Micarelli, A. (1997). Ocram-CBR – A Shell for Case-Based Educational Systems. In D. Leake & E. Plaza (1997), 104-113.
140. Pfeifer, T. and Richter, M. M. (eds.) (1993). *Diagnose von technischen Systemen - Grundlagen, Methoden und Perspektiven der Fehlerdiagnose*. Deutscher Universitäts-Verlag.
141. Plaza, E. & López de Mántaras, R. (1990). A Case-Based Apprentice that Learns from Fuzzy Examples. In: Ras, Zemankova, Emrich (eds.), *Methodologies for Intelligent Systems* 5, 420-427, North Holland.
142. Porter, B. W., Bareiss, R. & Holte, R.C. (1990). Concept Learning and Heuristic Classification in Weak-Theory Domains. *Artificial Intelligence, 45*.
143. Prased, N. V. N. & Plaza, E. (1996). Corporate Memories as Distributed Case Libraries. *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-based Systems Workshop*. Banff, Canada. Vol 2, pp. 40: 1-19.
144. Probst, G. & Romhardt (1998). The components of knowledge management - a practical approach (in German). URL: www.cck.uni-kl.de/wmk/papers/public/Bausteine/.
145. Puppe, F. & Günter, A. (eds.) (1993). Expertensysteme 93. Proc. of the 2nd German Conference on Expert Systems. Springer Verlag.
146. Racine, K. & Yang, Q. (1997). Maintaining Unstructured Case Bases. In *Leake & Plaza (1997*), 553-564.

147. Reinartz, T., Iglezakis, I. & Roth-Berghofer, T. (2000). On Quality Measures for Case Base Maintenance. In: *Blanzieri & Portinale (2000),* 247-259.
148. Richter, M. M. (1992). Classification and Learning of Similarity Measures. *Proc. 16th Annual Conference of the German Society for Classification*, Springer Verlag.
149. Richter, M. M. (1995a). The Knowledge Contained in Similarity Measures. Invited talk, First International Conference on Case-Based Reasoning (ICCBR'95), Sesimbra, Portugal.
150. Richter, M. M. (1995b). Some elementary remarks on case-based reasoning. *Unpublished manuscript.*
151. Richter, M. M. (1998). Introduction (to Case-Based Reasoning). In *Lenz et al. (1998a*), 1-16.
152. Richter, M. M. (2000). Personal communication.
153. Riesbeck, C. K. & Schank, R. C. (1989). *Inside Case-Based Reasoning*. Hillsdale, NJ: Lawrence Erlbaum Associates.
154. Rissland, E. L. (1983). Examples in Legal Reasoning: Legal Hypotheticals. *Proc. IJCAI-93.*
155. Rissland, E. L. & Daniels, J. J. (1995). Using CBR to Drive IR. In: C. S. Mellish (ed.), *Proc. IJCAI-95*, 400-408.
156. Ruhe, G. (2000). Learning Software Organizations*. In this volume*.
157. Schank, R. C. (1982). Dynamic Memory: A Theory of Learning in Computers and People. Cambridge University Press.
158. Schank, R. C. (1989). Case-Based Explanation. In: Car*bonell (1989),* 353-385.
159. Schank, R. C. (1998). *Inside Multi-Media Case Based Instruction*. Lawrence Erlbaum Associates.
160. Schank, R. C. & Abelson, R. (1977). *Scripts, Plans, Goals, and Understanding*. Lawrence Erlbaum Associates, Hillsdale, New Jersey.
161. Schmidt, S. & Vollrath, I. (eds.) (1999). Challenges for Case-Based Reasoning - Proc. of the ICCBR'99 Workshops. LSA-Report LSA-99-03E, University of Kaiserslautern, Germany.
162. Schulz, R. (1998). *Fallbasierte Entscheidungsunterstützende Systeme - ein Ansatz zu Lösung betrieblicher Entscheidungsprobleme*. Doctoral Dissertation, Faculty of Economics, University of Leipzig, Germany.
163. Schumacher, J. & Bergmann, R. (2000). An Efficient Approach to Similarity-Based Retrieval on Top of Relational Databases. In. *Blanzieri & Portinale (2000),* 273-284.
164. Seitz, A. (1999). A Case-Based Methodology for Planning Individualized Case Oriented Tutoring. In *Althoff et al. (1999), 318-328.*
165. Shimazu, H., Kitano, H. and Shibata, A. (1993). Retrieving Cases from Relational Data-Bases: Another Stride Towards Corporate-Wide Case base Systems. In: R. Bajcsy (ed.), *Proc. of the 13th IJCAI,* 909-914.
166. Smyth, B. & Cunningham, P. (1995). A Comparison of Incremental Case-Based Reasoning and Inductive Learning. In: *Keane et al. (1995)*.
167. Smyth, B. & Cunningham, P. (eds.)(1998). *Advances in Case-Based Reasoning: Proc. of the Forth European Workshop on Case-Based Reasoning*. Springer Verlag.
168. Smyth, B. & Keane, M. (1995). Remembering to Forget: A Competence-Preserving Case Deletion Policy for Case-Based Reasoning Systems. In: C. S. Mellish (ed.), *Proc. IJCAI-95*, 377-383.
169. Smyth, B. & McKenna, E. (1998). Modelling the competence of case-bases. In*: Smyth & Cunningham (1998),* 196-207.
170. Solingen, R. van & Berghout, E. (1999). *The goal/question/metric method, a practical method for quality improvement of software development.* McGraw-Hill.

171. Someren, M. van, Zheng, L. L. & Post, W. (1990). Cases, models, or compiled knowledge? A cooperative analysis and proposed integration. In: B. Wielinga, J. Boose et al. (eds.), *Current Trends in Knowledge Acquisition*, IOS Press, 339-355.
172. Stanfill, C. & Waltz, D. L. (1986). Towards memory based reasoning. *Communications of the ACM*, 29, 1213-1229.
173. Stolpmann, M. & Stefan Wess, S. (1998). *Optimierung der Kundenbeziehung mit CBR-Systemen. Intelligente Systeme für E-Commerce und Support*. Addison Wesley.
174. Tautz, C. (2000). *Customizing Software Engineering Experience Management Systems to Organizational Needs*. Doctoral Dissertation, Departmant of Computer Science, University of Kaiserslautern, Germany.
175. Tautz, C. & Althoff, K.-D. (1997). Using Case-Based Reasoning for Reusing Software Knowledge. In: *Leake & Plaza (1997),* 156-165.
176. Tautz, C. & Althoff, K.-D. (2000). A Case Study on Engineering Ontologies and Related Processes for Sharing Software Engineering Experience. *Proceedings 12th International Conference on Software Engineering and Knowledge Engineering (SEKE'00).*
177. Tautz, C. & Gresse von Wangenheim, C. (1998). *A Representation Formalism for Supporting Reuse of Software Engineering Knowledge.* IESE-Report 051.98/E, Fraunhofer IESE, Kaiserslautern, Germany.
178. Traphöner, R. (2000). Worth Writing Home About: Some Notes on Methodology in a Big Software Organisation. In: K.-D. Althoff & W. Müller (eds.), *Proc. 2nd Internat. Workshop on Learning Software Organisations*, Fraunhofer IESE, Kaiserslautern, Germany.
179. Traphöner, R., Manago, M., Conruyt, N. & Dittrich, S. (1992). *Industrial Criteria for Comparing the Technologies of INRECA*. INRECA (Esprit P6322), Deliverable D4.
180. Tversky, A. (1977) Features of Similarity. *Psychological Review*, 84, 327-362.
181. Veloso, M. (1994). *Planning and Learning by Analogical Reasoning*. Springer Verlag.
182. Veloso, M. & Aamodt, A. (eds.) (1995). *Case-Based Reasoning Research and Development*. Proc. of the First International Conference on Case-Based Reasoning (ICCBR'99), Springer Verlag.
183. Veloso, M., Munoz-Avila, H. & Bergmann, R. (1996). Case-based planning: selected methods and systems. *AI Communications 9(3), September 1996*, 128-137.
184. Watson, I. (1997). *Applying Case-Based Reasoning - Techniques for Enterprise Systems*. Morgan Kaufmann.
185. Watson, I. (1998). CBR is a methodology not a technology. In: R. Miles, M. Moulton & M. Bramer (eds.), *Research & Development in Expert Systems XV*, 213-223. London: Springer.
186. Weber, G. (1996). Episodic learner modeling. Cognitive Science, 20, 195-236.
187. Weber, G. & Schult, T. J. (1998). CBR for tutoring and help systems. In Lenz et al. (1998a), 255-271.
188. Weibelzahl, S. & Weber, G. (1999). Benutzermodellierung von Kundenwünschen durch Fallbasiertes Schließen. . In Proc. LWA-1999, University of Magdeburg, Germany, 295-300.
189. Wess, S. (1993). PATDEX - ein Ansatz zur wissensbasierten und inkrementellen Verbesserung von Ähnlichkeitsbewertungen in der fallbasierten Diagnostik. In: *Puppe and Günter (1993)*, 42-55.
190. Wess, S. (1995). *Fallbasiertes Schließen in wissensbasierten Systemen zur Entscheidungsunterstützung und Diagnostik*. Doctoral Dissertation, University of Kaiserslautern, Germany.

191. Wess, S., Althoff, K.-D. and Derwand, G. (1994a). Using k-d Trees to Improve the Retrieval Step in Case-Based Reasoning. In: *Wess, Althoff & Richter (1994)*, 167-181

192. Wess, S., Althoff, K.-D., Maurer, F., Paulokat, R., Praeger, R. & Wendel, O. (eds.) (1992). *Fallbasiertes Schließen - Eine Übersicht, Bände 1-3.* SEKI-REPORT SWP-92-08/09/10, University of Kaiserslautern, Germany, 425 pages

193. Wess, S., Althoff, K.-D. and Richter, M. M. (eds.) (1994b). *Topics in Case-Based Reasoning*. Springer Verlag.

194. Wess, S. & Globig, C. (1994). Case-Based and Symbolic Classification Algorithms - A Case Study Using Version Space. In: *Wess, Althoff & Richter (1994)*, 77-91.

195. Wettschereck, D., Aha, D. W. & Mohri, T. (1997). A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review, 11*, 273-314.

196. Wilke, W. (1999). *Knowledge Management for Intelligent Sales Support in Electronic Commerce*. Doctoral Dissertation, University of Kaiserslautern, Germany.

197. Zhu, J. & Yang, Q. (1999). Remembering to Add: Competence-preserving Case Addition Policies for Case Base Maintenance. Proc. *International Joint Conference in Artificial Intelligence (IJCAI-99), August 1999.*

198. Perner, P. (1998). Different Learning Strategies in a Case-Based Reasoning System for Image Interpretation. In B. Smyth & P. Cunningham, *Advances in Case-Based Reasoning*, Springer Verlag, 251-261.