Knowledge modelling and maintenance in myCBR3*

Alexander Hundt[1,A], Pascal Reuss[1,B], Christian Sauer[2,C], and Thomas Roth-Berghofer[2,D]
[A]alexander.hundt@dfki.de [B]pascal.reuss@dfki.de [C]christian.sauer@uwl.ac.uk
[D]thomas.roth-berghofer@uwl.ac.uk

[1]DFKI German Research Center for Artificial Intelligence, Kaiserslautern, Germany and
[2]University of West London, London, UK

Abstract. One of the main aspects of knowledge management is the task of knowledge maintenance. Building and running knowledge intensive Case-Based Reasoning applications requires fundamental design decisions during the system design phase with regard to the knowledge maintenance within the system as well as accurate knowledge maintenance approaches within the running system. In this paper we will detail on the design decisions available in the *myCBR 3* CBR system design software as well as research in the available and future knowledge maintenance approaches within *myCBR 3* CBR. Next to maintaining the standard knowledge of any CBR system, represented by the four knowledge containers after Richter, this paper also presents existing and currently researched approaches to represent and furthermost maintain context knowledge as well as explanatory knowledge within *myCBR 3* CBR . We will give an overview of the *myCBR 3* CBRs Knowledge Engineering workbench, providing the tools for the modelling and maintenance process and detail on currently explored new features to further integrate knowledge maintenance for context-sensitive and explanation aware CBR systems into our *myCBR 3* CBR software.

1 Introduction

Case-Based Reasoning (CBR) is a methodology introduced by Riesbeck and Schank [9] and Kolodner [4]. CBR is mimicking the human approach of reusing past experience to solve new problems. The basic reasoning model of CBR, the so called CBR cycle, was introduced by Aamodt and Plaza [1]. The CBR cycle consists of four processes: *Retrieve*, *Reuse*, *Revise* and *Retain*. The episodes of experience that CBR reasons upon are stored in cases that consist of pairs of problem and solution descriptions. Problem descriptions are described by tuples of attribute value pairs that describe a problematic or critical situation. The corresponding solution description of a case consists of information how the problem described in the problem description was successfully solved. In the *Retrieve* phase of the CBR cycle the attribute value pairs describing a current problem encountered are matched against the problem descriptions in all cases within

 In: T. Seidl, M. Hassani, C. Beecks (Eds.): Proceedings of the LWA 2014 Workshops: KDML, IR, FGWM, Aachen, Germany, 8-10 September 2014, published at http://ceur-ws.org

the case base of the CBR system. The CBR system employs similarity measures to calculate the distances between singular attribute values and subsequently the similarity between the current problem description and the problem descriptions in the case base's cases. A selectable number of n best matching cases are than retrieved from the case base. In the *Reuse* step the retrieved cases solutions are than applied to the current problem in order to try and solve the problem at hand. this reuse of the past solution(s) may involve the adaption of the solutions described in the retrieved past cases. After applying an adapted solution the final outcome of the solution, either being successful or partly successful or failing is revised in the *Revise* step of the CBR cycle. If the solution was successful the new case, consisting of the current problem description and the successfully applied solution, that may have been adapted, is retained in the CBR systems case base in CBR cycles final *Retain* step.

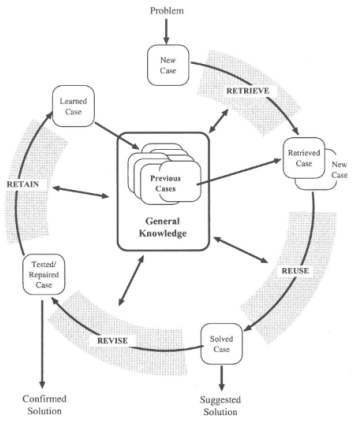

Fig. 1. The CBR cycle

The knowledge that is involved in the reasoning steps of the CBR cycle is represented in formal form within the four knowledge containers of CBR introduced by Richter [7].

- *Vocabulary* defining attributes and their allowed value ranges. This can be value ranges (min, max) for numeric attributes or a list of allowed symbols for symbolic attributes.
- *Similarity Measures* Functions to calculate the similarity between individual attribute values (local similarity measures) as well as the similarity between whole problem descriptions (global similarity measures). These functions are often distance functions for numeric values or comparative tables or taxonomies of symbols for symbolic attributes.
- *Adaptation Knowledge* Often represented by rules that can be applied to adapt the solution description of retrieved cases to match the current problem and enable it to be solved.

– *Cases* The descriptions of past episodes of experience, consisting of pairs of problem and solution descriptions. They employ tuples of attribute value pairs to describe the problematic situation and store a solution to the described problem within a solution description part.

Next to the aforementioned knowledge containers we consider two additional fields of knowledge important and incorporate these into our research: knowledge about context and knowledge about explanations. While these two additional fields do not represent further knowledge containers, they provide significant information that can be leveraged in order to make Case-Based Reasoning systems more comprehensible and also offer ways to improve maintenance capabilities.

In a gist we follow the definition of context as *'a descriptor (such as a word or an image) or set of descriptors that can represent a situation or a scenario.'* [15]. For further detail, context may be perceived as a situation which is depicted as a subset of descriptors that are part of a snapshot of the world at a given moment in time. This snapshot encompasses all existing objects in this world, their relationships and the states they are currently in. For another classification a real world situation, being a real subset of objects can be distinguished from an observed situation which represents the situation determined by the system.

If a system has the ability to classify situations, the knowledge about how to react in a determined situation and furthermore can distinguish between other similar situations, then this classification serves as the context the system is in [19].

We see the benefit of gaining knowledge about a system's context in obtaining a priori knowledge about a given situation without the need for gathering information with additional effort.

An explanation in its basic form may be an answer to a question. As such the knowledge about explanation enables the system to answer questions that are not directly related to a user's search query, yet improve the user experience by providing additional transparency and justification. Usually a user's question may involve certain trigger words like 'why' or 'how'. If put in a computing context the *general explanation scenario* consists of primarily three components. First the user, being the one who interacts with the system via a user interface (UI). Second the problem solver, being the actual software which executes functional tasks to comply with the users request. Finally the explainer itself, being the additional component required to trace the actions of the problem solver and present them via the UI towards the user [11].

As for the expected benefit we see the build up of confidence in a system valuable in terms of confidence in a query result as well as maintenance proposals, depending on how valuable the given explanations are deemed by the user. We primarily pursue the following five kinds and goals of explanations[12].

– Conceptual explanations fulfill the *learning* goal as they offer descriptive information about symptoms
– Why-explanations provide a *relevance* of an answer, thus explaining why the answer is a good answer
– How-explanations elucidate how the reasoner concluded the answer and therefore add *transparency* to the result

- Purpose-explanations present themselves as an explanation similar to conceptual explanations and might therefore be applicable for describing how concepts are related, thus providing *justification* for an explanation
- Cognitive explanations have an exceptional position as they aim at explaining non-physical attributes

The objective of this paper is to demonstrate existing and currently developed approaches to maintain the knowledge within the four knowledge containers as well as the additional explanatory and context knowledge that can be modelled within *myCBR 3*.

The rest of this paper is structured as follows: In section 2 we review related work on knowledge maintenance in CBR. We then introduce the process of knowledge modelling in *myCBR 3* in section 3. Based on the description of how to model CBR knowledge models in *myCBR 3*, we then introduce and discuss existing and currently developing approaches to enable *myCBR 3* to support knowledge maintenance 4. We do so with a focus on the four knowledge containers of CBR in the sections 4.1,4.2,4.3. Finally in section 5 we discus the introduced approaches to knowledge modelling and maintenance and conclude.

2 Related Work

In the introduction we have already reviewed the four knowledge containers of CBR [8], for each of these containers there already exist a number of approaches to maintain the knowledge represented in the container [20]. Maintenance for the knowledge in the contaioners is neccesary due to the fact that any change in the environment a CBR system operates in can affect the accuracy and competence of the CBR system's knowledge model [10]. Therefore the maintenance of CBR systems, particularly maintaining their knowledge models, is an important and on-going task.

Maintaining a knowledge model comprises tasks such as revising the knowledge within the model to cater for changes in the domain, add new knowledge to the model or remove knowledge that became deprecated. For example for the case base it is vital to control the case base's size as well as to detect inconsistent cases see for example the work of [16,10]. As a concrete example, a case base maintenance approach, introduced by Smyth and McKenna, is based on a performance model of a CBR system [17] which is used the to identify less competent cases to delete them from the case base.

Another approach to case base maintenance was introduced by Leake and Wilson [5] highlighting the importance of conducting case base maintenance by balancing the competence-performance dichotomy of a CBR system. In addition Leake and Wilson suggest that case base maintenance should be guided by important constraints including size limits of case base such as long and short term performance goals in expected future problems.

Next to the best researched maintenance of the knowledge container case base, there exists a number of further research on the other three knowledge containers. Out of these the knowledge container similarity meassures is the second best researched with regard to the maintenace of the knowledge in this cotainer, see for example the work of [3]. Another approach to maintain the casebase and the similarity knowledge is from [6]. They improve the quality of similarity measures by enhancing the coverage of cases.

3 Knowledge modelling in myCBR

3.1 Case-Based Reasoning framework *myCBR 3*

Developing a CBR systems knowledge model requires a systematic approach to capture and formalise the domain knowledge into the four knowledge containers of CBR. Such a knowledge modelling task is often a process that involves a significant effort, it is therefore desirable to rely on modelling software for this crucial initial development process for a CBR system. *myCBR 3*[1] is such a modelling software. It is an open source tool targeting at developing knowledge models [18]. It is emphasizing the ability to rapidly prototype a cbr knowledge model, especially the contents of the vocabulary and similarity measure containers. Next to the modelling facilities *myCBR 3* also offers a similarity-based retrieval tool as well as a software development kit (SDK). *myCBR 3* offers a veriety of GUIs within its Workbench that enables a knowledge engineer to model and test a knowledge model, especially sophisticated similarity measures. The knowledge model can further be tested within the *myCBR 3* Workbench, using the inbuilt retrieval test tool to refine and update the knowledge model which are both functions that play a key role within the task of knowledge maintenance.

Using the *myCBR 3* SDK allows for an easy integration of the knowledge model into a java-based application. The follwing code example shows a simple retrieval on a myCBR case base.

```
    /*Initialize the retrieval engine*/
    Retrieval ret = new Retrieval(concept, casebase);
    SequentialRetrieval = new Retrieval(project, ret);

    Instance query = ret.getQueryInstance();

    /*Here the query has to be set*/
    query.mapInputToAttributes();

/*The resulting List contains k cases sorted by similarity */
    List<Pair<Instance, Similarity>> result =
     seqret.retrieveKSorted(casebase, query, k);

    printResults();
```

3.2 Knowledge modelling with the myCBR Workbench

As mentioned earlier the *myCBR 3* Workbench provides powerful GUIs for modelling CBR knowledge models. A key focus of the Workbench is laid on the modelling of

[1] http://www.mycbr-project.net

knowledge-intensive similarity measures. Furthermore the Workbench provides task-oriented view-configurations for either modelling your knowledge model, perform information extraction or case base management. To enable the testing and refinement of developed knowledge models the Workbench offers a similarity-based retrieval functionality. The *myCBR 3* SDK employs a simple-to-use data model which facilitates the integration of the knowledge model into any java-based application. Both, the retrieval process as well as the case loading are fast and therefore allow for a seamless integration and use within applications built on top of a knowledge model developed with *myCBR 3*.

myCBR 3 allows for each attribute to have several similarity measures, which in turn allows for allows for experimenting with different similarity measures to record variations. Next to experimentation this feature can also be used to select an appropriate similarity measure at run-time via the API to accommodate for different contexts such as for example different types of users.

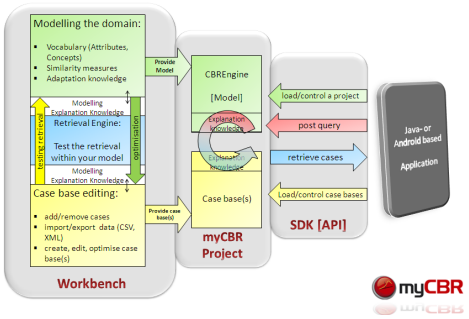

Fig. 2. Integration of the *myCBR 3* workbench and SDK in CBR project development

3.3 Modelling the Case Structure and Similarity Meassures

The *myCBR 3* Workbench offers two different views to edit either the knowledge model or the case base(s). In this section we will shortly introduce the modelling view for the knowledge model as shown in 3. The concept of modelling a knowledge model in the Workbench follows the approach that initially a case structure is created. based on the initial case structure the vocabulary is then defined and the necessary individual local similarity measures for each attribute description (eg. CCM in 3) are then created, followed finally by the global similarity measure for a concept description (Car in 3).

The modelling view of the Workbench (see figure 3) is showing the case structure on the left side, available similarity measures for a selected attribute or concept beneath it and the definition of a similarity measure or attribute in the center. The modelling of similarity meassures in the Workbench takes place on either the attribute level for local similarity measures or the concept level for global similarity measures.

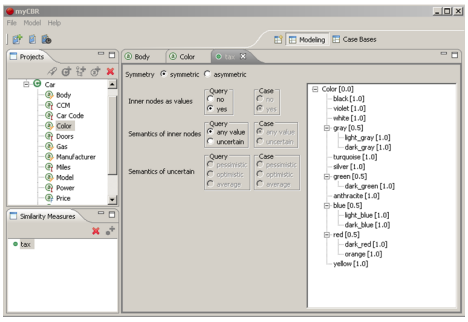

Fig. 3. Example of the knowledge model view in the *myCBR 3* Workbench

3.4 Building a Vocabulary

As mentioned earlier, the vocabulary wityhin a *myCBR 3* knowledge model consists of concepts and attributes. A concept can consist of one or more attribute descriptions as well as attributes referencing different concepts. this representation allows the user to create object-oriented case representations. In the current version *myCBR 3* allows for the import of vocabulary items, e.g. concepts and attributes, from existing CSV files as well as from Linked (Open) Data (LOD) sources. A versatile feature for the case import is the re-construction of case structures when importing case data from CSV files. Within /mycbr3 an attribute description can have one of the following data types: Double, Integer, Date and Symbol. For each of these data types *myCBR 3* provides similarity functions editors.

Next to the four knowledge containers of CBR, *myCBR 3* allows for the representation of explanatory knowledge, being knowledge used to create explanations of the systems reasoning and results. For example *myCBR 3* allows for providing canned explanations as well as references to online sources for concept explanations. *myCBR 3* further allows to represent context knowledge via the definition of a multiple of similarity measures for both, attributes as well as cases.

4 Knowledge maintenance in myCBR

Having modelled a knowledge model the next important task is to maintain the knowledge represented within the model. In this section we will introduce and review current approaches to knowledge maintenance being implemented at the time for the *myCBR 3* Workbench. We further introduce already publishhed work on approaches to adaption knowledge modelling and the potential yto use this approach for future adaption knowledge maintenance.

4.1 Case knowledge

A recent approach to integrate knowledge maintenance facilities for case bases into *myCBR 3* is described in [2]. The approach aimed to provide a maintenance perspective

in *myCBR 3* to assess data on case usage. The case usage data was then used to generate quality measures which were employed to trigger maintenance measures for the knowledge in the case base, vocabulary and similarity measures.

The maintenance perspective was implemented by extending *myCBR 3*, enabling to generate usage-data on the access of individual cases during retrieval. Based on experimentation a set of threshold values for quality measures were established which were used to monitor the top performing and most retrieved cases as well as the least performing and least retrieved cases. Monitoring the best as well as the least performing cases allowed for the analysis of well performing cases and the adjustment of less well performing cases. Next to simply deleting the least performing cases, the less performing cases can be adapted within the maintenance perspective. The approach also included the necessary features to reverse these changes, in case the performance deteriorates after the changes. The conceptual approach of automating the measurement of the quality measures and the subsequent triggering of the maintenance tasks can be seen as a control loop to manage the maintenance of the case base.

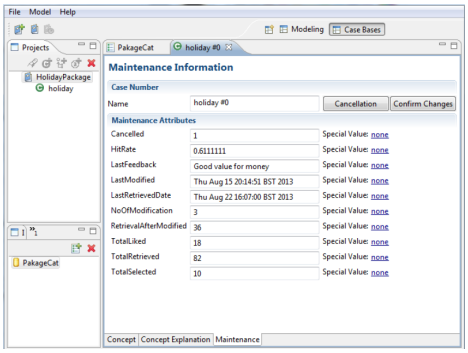

Fig. 4. The added maintenance view in *myCBR 3*

The described approach implemented the maintenance attributes temporarily by simulating these attributes in *myCBR 3*. the simulation of the attributes was achieved by labeling the maintenance attributes in a specific way. Figure 4 shows the implemented maintenance view with the maintenance attributes implemented for a holiday package recommender system.

4.2 Similarity knowledge

As stated earlier, *myCBR 3* allows for similarity based retrieval tests. These tests allow for the evaluation of the efficiency and accuracy of a knowledge model built with *myCBR 3*. The tests can be performed, of course, at the development state of the knowledge model but, more importantly in the context of this paper, also on a knowledge model that is already included in a life application. So, for example, we assume a product recommender system built on top of a *myCBR 3* knowledge model. The knowledge model

can be tested parallel within the Workbench environment, which allows for establishing the need for maintenance measures as well as to test the outcome of maintenance measures in a sort of "'dry-dock"' environment. If the similarity meassure maintenance is finished within the *myCBR 3* workbench the updated knowledge model can be reloaded seamless in the life application via the *myCBR 3* API. This functionality allows for the seamless incorporation of user feedback as well as for the continuous refinement of the similarity knowledge in the model and thus for maintaining the accuracy of the model. Currently the authors are investigating the approach to adapt the approach to acquire and use usage data and retrieval test data from the knowledge model, described in the previous subsection 4.1. The aim of this on-going work is to provide a similar maintenance view in *myCBR 3* to cater for similarity meassure maintenance.

4.3 Adaptation knowledge

As mentioned in the sections above the tool *myCBR 3* supports the *revtrieve* step of the CBR cycle. In the near future *myCBR 3* will also support the *reuse* step of the CBR cycle. This subsection describes the functionality that our tool will provide.

To support the *reuse* step *myCBR 3* is combined with the open source tool JBOSS Drools. Drools consists of five projects: Drools Guvnor, Drools Expert, jBPM5, Drools Fusion and OptaPlaner. For our purpose only Drools Expert, which contains the rule engine, is combined with *myCBR 3*. There are several reasons for choosing Drools for the adaptation:

- 100% JAVA, so it is easy to integrate in *myCBR 3*
- license compatible to *myCBR 3* license
- performance of the Rete algorithm
- scalability of the rule bases
- independent lifecycle

The use of Drools Expert allows us to define completion and adaptation rules and process them to enrich the query and adapt the retrieved cases. In our rule concept a rule belongs to a rule base and has five properties:type, case base, precondition, condition expression and conclusion expression. The first property defines the type of the rule, either completion rule or adaptation rule. The second property defines to which case based a rule is assigned. A rule can be assigned to several case bases at once. Completion rules are not assigned to a case base, because they are used to enrich a query. Adaptation rules has to be assigned to at least one case base. The precondition property allows to define a set of conditions, that is used to determine if a rule has to be checked for firing or not. This way the number of rules the Rete algorithm has to process can be significantly reduced and therefor the performance of the rule processing is increased. The condition and the conclusion properties are used to define the rule itself. The condition expression is a set of one or more single conditions which consists of an attribute of the case structure, an operator and a value. The single conditions are combined with logical operators AND or OR. The conclusion expression works the same way, but the logical operator is always AND.

For the implementation of the rule concept the *myCBR 3* SDK and the myCBR workbench are extended. The SDK is extended with several new classes to support the

rule properties mentioned above. The myCBR workbench is extended with a new view that contains a rule editor. Figure 5 shows the new rule editor.

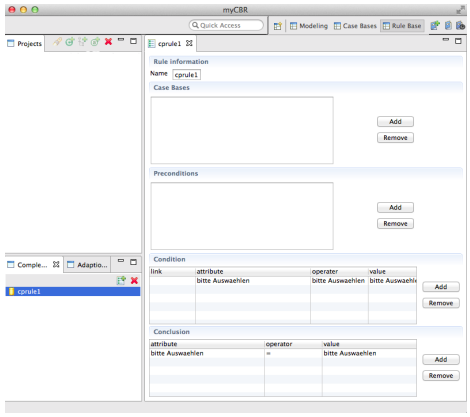

Fig. 5. myCBR rule editor

With the help of this editor new rule base can be created. A rule base can contain completion rules and adaptation rules. When a case base is added the editor presents a list of possible case base the rule can be applied to. This is the first action that has to be done, because based on the assigned case based, the condition and conclusion has different lists of attributes and values that can be chosen. To define the condition and conclusion of a rule the editor uses select lists. The list for the attributes contains all attributes from the assigned case bases and the value list contains all values that are allowed for the selected attribute. The operator list contains at the moment only $\leq, \geq, =, \neq$, but more operator will be implemented in the future. If more than one condition is defined a user can choose from the link list whether the conditions are link with AND, OR or XOR.

The next step after implementing the rule editor and the rule processing, is to define evaluation and maintenance strategies for the adaptation knowledge. A simple evaluation strategy may be to check the selected attributes and values against the defined vocabulary. This way inconsistency could be found after some terms have been removed from the vocabulary. Another evaluation strategy is to check the rules if there are conflicts among themselves. Maintaining the rules can be done with the help of the rule editor. A new maintenance view could be used to display the evaluation results to the knowledge engineer. The engineer then has to decide which maintenance actions must be done, either changing a rule or removing it.

5 Discussion and Conclusions

In this paper we have presented the *myCBR 3* Workbench and its use to model and maintain knowledge for CBR systems. We did so by reviewing the existing process of

modelling knowledge for the four knowledge containers of CBR and from there on elaborate on currently available knowledge maintenance approaches as well as approaches currently being developed within *myCBR 3*.

A topic for discussion can be seen in the question of the pay-off between the initial effort to implement the maintenance functionalists within the *myCBR 3* GUI, establish and create the maintenance measurement attributes and useful threshold values, compared to the actual benefits from the maintenance approaches. The authors conclude, based preliminary testing and feedback from knowledge engineers and non CBR-expert domain experts, that the effort of implementing the maintenance measures is well worth it. From a series of published research project [13] [14] it can definitely said that the pay off of the GUI-based new functionalists is high as these can be used even by non CBR experts to implement maintenance measures with their CBR knowledge models.

Based on the benefits that were gained from the implementation of knowledge maintenance for the existing 4 knowledge containers of CBR the nect step in our work is the introduction of maintenance measures for explanatory and context knowledge, as these functionalities, explanation awareness and context-awareness, are themselves still in a prototype status within *myCBR 3*. However the authors assume the benefit from these functions as so high that their implementation in a future release of the software is highly likely. An additional point to argue for the integration of the maintenance measures for explanatory and context knowledge in *myCBR 3* lies in the fact that implementing these measures alongside the implementation of the explanation aware and context aware functions offers the opportunity to take the importance of the maintenance functions into account.

So the authors conclude that the approaches to knowledge maintenance within *myCBR 3* developed so far and currently under development are useful and desirable. This conclusion is based on experimental results as well as on feedback from domain experts working with prototypes of *myCBR 3*, published in a number of workshop and conference submissions. Furthermore it can be concluded that it is a rewarding task to develop these approaches as they reduce the pressure on the initial knowledge modelling with regard to the absolute need of formalising the knowledge 100 per-cent correct at the first (development) step, as the maintenance measures, along with the performance measuring functionalities, for example the case performance measuring, can easily be used to amend in reaction to a changing domain or to refine a probably not optimally designed initial knowledge model.

Additionally the highly modularised code structure of *myCBR 3* allows for easy expansion, adaption, so a lot more approaches to representing maintenance knowledge, maintaining knowledge and controlling the triggering of maintenance measures can be easily developed. Finally, the integration of the approaches presented in this paper is currently on-going and the maintenance functions presented will be part of the a new release version of mycbr3.x in the near future.

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. Artificial Intelligence Communications, 7(1), 39–59 (1994)

2. Aul, V., Sauer, C.S., Bulbul, E., Wilson, D.C., Roth-Berghofer, T.: Knowledge maintenance in mycbr. In: Proceedings of the 18th UKCBR 2013 Workshop. Springer (2013)
3. Bergmann, R., Stahl, A.: Similarity measures for object-oriented case representations. Springer (1998)
4. Kolodner, J.L.: Case-Based Reasoning. Morgan Kaufmann Publishers, Inc. San Mateo (1993)
5. Leake, D.B., Wilson, D.C.: Categorizing Case-Base Maintenance: Dimensions and Directions. In: Smyth, B., Cunningham, P. (eds.) Advances in Case-Based Reasoning, Proceedings of the 4th European Workshop on Case-Based Reasoning, EWCBRY98, Dublin, Ireland. pp. 196–207. Springer-Verlag, Berlin (1998)
6. Patterson, D., Anand, S.S., Hughes, J.: A knowledge light approach to similarity maintenance for improving case-base competence. In: European Conference on Artificial Intelligence Workshop Notes. pp. 65–78 (2000)
7. Richter, M.M.: Introduction. In: Lenz, M., Bartsch-Spörl, B., Burkhard, H.D., Wess, S. (eds.) Case-Based Reasoning Technology – From Foundations to Applications. LNAI 1400, Springer-Verlag, Berlin (1998)
8. Richter, M.M.: Introduction. chapter 1 in case-based reasoning technology - from foundations to applications. lnai 1400, springer (1998)
9. Riesbeck, C.K., Schank, R.C.: Inside case-based reasoning. Lawrence Erlbaum Associates, Pubs., Hillsdale, N.J. (1989)
10. Roth-Berghofer, T.: Knowledge Maintenance of Case-Based Reasoning Systems – The SIAM Methodology, Dissertation. Ph.D. thesis, Universität Kaiserslautern (2003)
11. Roth-Berghofer, T., Sauer, C.S., Althoff, K.D., Bach, K., Newo, R.: Seasaltexp - an explanation-aware architecture for extracting and case-based processing of experiences from internet communities. In: Proceedings of the LWA 2011 - Learning, Knowledge, Adaptation (2011)
12. Roth-Berghofer, T.R.: Explanations and case-based reasoning: Foundational issues. In: Funk, P., Calero, P.A.G. (eds.) Advances in Case-Based Reasoning. pp. 389–403. Springer-Verlag (September 2004)
13. Sauer, C.S., Hundt, A., Roth-Berghofer, T.: Explanation-aware design of mobile mycbr-based applications. In: Case-Based Reasoning Research and Development, pp. 399–413. Springer (2012)
14. Sauer, C.S., Roth-Berghofer, T., Aurrichio, N., Proctor, S.: Similarity knowledge formalisation for audio engineering. In: Proceedings of the 17th UKCBR 2012 Workshop. Springer (2012)
15. Segev, A.: Identifying the multiple contexts of a situation. Modeling and Retrieval of Context pp. 118–133 (2006)
16. Smyth, B.: Case-base maintenance. In: Tasks and Methods in Applied Artificial Intelligence, pp. 507–516. Springer (1998)
17. Smyth, B., McKenna, E.: Building compact competent case-bases. In: Althoff, K.D., Bergmann, R., Branting, L.K. (eds.) Case-Based Reasoning Research and Development: Proceedings of the Third International Conference on Case-Based Reasoning, ICCBR'99, Seeon Monastery, Germany. pp. 329–342. Springer-Verlag, Berlin (1999)
18. Stahl, A., Roth-Berghofer, T.R.: Rapid prototyping of CBR applications with the open source tool myCBR. In: Proceedings of the 9th European conference on Advances in Case-Based Reasoning. pp. 615–629. Springer-Verlag, Heidelberg (2008)
19. Turner, R.: A model of explicit context representation and use for intelligent agents. Modeling and Using Context pp. 831–831 (1999)
20. Wilson, D.C., Leake, D.B.: Maintaining case-based reasoners: Dimensions and directions. Computational Intelligence 17(2), 196–213 (2001)