myEACBR – myCBR as explanation-aware Protégé plugin

Marvin Bredal Lillehaug[1],
Thomas Roth-Berghofer[2], and Anders Kofod-Petersen[1]

[1] Department of Computer and Information Science,
Norwegian University of Science and Technology,
7491 Trondheim, Norway
marvin.lillehaug@gmail.com,anderpe@idi.ntnu.no
[2] Centre for Model-based Software Engineering and Explanation-aware Computing, School of Computing and Technology,
University of West London, London W5 5RF, UK
Thomas.Roth-Berghofer@uwl.ac.uk

Abstract. Explanation, trust, and transparency are concepts that are strongly tied in with users' confidence in, and acceptance of computerised systems. Case-based reasoning (CBR) systems lend themselves easily to generate explanations, as they typically organise and represent knowledge in a way that makes it possible to reason about and thereby generate explanations. The work presented here is a first step towards making a CBR engine explanation-aware. We demonstrate how a plugin for Protégé and myCBR can facilitate explanations for the retrieval phase of a CBR system.

Key words: case-based reasoning, explanation-aware computing

1 Introduction

An increasing number of tasks are delegated to computerised and automated systems. With more and more systems being in charge of a process users have less control over which operations are done and why. This can lead to confusion since the user does not know the motivation behind actions performed; they might not even know which actions are performed. If a system performs actions that is unexpected confidence in the system will decrease unless it can justify its behaviour. Mitigating these problems can be done by putting effort into making systems *explanation-aware*.

The ability to explain yourself, your reasoning and actions, has been identified as one core capability of any intelligent entity [1]. However, the quality of a good explanation is context dependent [2]. This means that reasoning systems must have a built-in support for explanations.

The term explanation has been widely investigated in different disciplines such as cognitive science, artificial intelligence, linguistics, philosophy of science, and teaching. All these disciplines consider certain aspects of the term and make

clear that there is not only one such concept but quite a number of them. Some of these aspects have been applied in knowledge-based systems.

One way of looking at explanations that allows to handle explanations by software systems is to treat explanations as answers to questions. Whenever you experience something unexpected while working with a software system you have a question, and you expect an answer to it. It does not matter whether the question is raised explicitly or not.

Explanations are an important vehicle to convey information to understand one another in everyday conversations. They enhance the knowledge of communication partners in such a way that they understand each other better. Explanation-aware computing (ExaCt) is the vision of software systems being smart in interactions with their users [3]. Explanation-aware Software Design (EASD) aims at making software systems smarter in this regard. EASD looks at ways to guide software designers and engineers to a purposeful explanation-aware software system by making their designers and engineers explanation-aware [4].

The work presented in this paper reports on the first step on enhancing a case-based reasoning engine with explanatory capabilities [5]. We demonstrate how the retrieval phase in myCBR [6] can be made explanation-aware by implementing a plugin for Protégé 4.x and the myCBR SDK[3] 3.x that generates explanations for the similarity values found in the retrieval step of the case-based reasoning cycle. In addition to this, the presented system is able to explain concepts used in the domain model by consulting external knowledge sources.

The rest of the paper is organised as follows: Section 2 gives an overview of related work; Section 3 describes the design and implementation of the explanation-aware case-based reasoner (myEACBR); Section 4 discusses the implementation; the paper concludes with a summary and outlook on future work.

2 Background and Related Work

In order to use information we have to know how it relates to other things. There is a significant difference between knowing that a system works and how or why it works. Further, the user of a system is more likely to trust it when the system can rationalise its behaviour, in particular how it reaches an answer [7].

Generally, there is no point in giving an explanation if its content is not understandable to the user; regardless of whether it is valid and can be considered a true explanation. When constructing an explanation we therefore have to consider who the receiver is and the user's level of understanding. This aspect of explanations and user modelling has received some attention over the years (see, e.g., [8,9]). However, this will not be the focus of the work presented here.

Originally, case-based reasoning emerged from an understanding of reasoning as being a process of explanation [10,11]. Explanations were described as the most common method used by humans to support decision making.

Sørmo et al. [1] present a framework for explanations in intelligent systems with a special focus on case-based reasoning. Specifically, they identify five goals

[3] Software Development Kit

that explanations can satisfy: *Transparency* is concerned with how an answer was reached. This can simply be a trace of the reasoning process (for experts); *Justification* deals with why the answer is good. This is closely related to the transparency goal, although transparency is typically for experts while justification is typically for non-experts; *Relevance* deals with how relevant a question is, that is both from the user and system; *Conceptualisation* is the goal that handles the meaning of concepts; Finally, *learning* is in itself a goal, as it teaches us about the domain in question. These goals are defined from the perspective of a human user. His expectation on what constitutes a good explanation is situation dependent and has a historic dimension (compare, e.g., Leake [12]).

Roth-Berghofer [13] has explored some fundamental issues with different useful kinds of explanation and their connection to the different knowledge containers of a case-based reasoning system. Five different kinds of explanation are identified: *conceptual explanations*, which map unknown new concepts to known ones, *why-explanations* describing causes or justifications, *how-explanations* depicting causal chains for an event, *purpose-explanations* describing the purpose or use of something, and *cognitive explanations* (also called *action explanations*) explaining or predicting the behaviour of intelligent systems. Roth-Berghofer and Cassens further on tie these different kinds of explanation to the different knowledge containers of case-based reasoning systems [14], namely case base, similarity measure, adaptation knowledge, and vocabulary.

Other work has focused on using CBR as a mechanism to realise ambient intelligent systems [15] and the importance of explanations generated through CBR in that context [16]. The knowledge intensive CBR framework CREEK was used [17]. Its main asset is that cases are submerged into the general domain model, which is realised through a multi-relational semantic network where an object-oriented, frame-based representation is used for both cases and domain model. CREEK supports the following goals (from [1]): *transparency* through visualisation of case matches; *justification* by allowing the user to investigate how concepts matches; and *conceptualisation* by allowing the user to investigate the knowledge base. For an overview of explanatory capabilities in CREEK see [18].

myCBR is an open source case-based reasoning tool[4]. Key motivation for implementing myCBR was the need for a compact, easy-to-use tool for building prototype CBR applications in teaching, research, and small industry projects with minimal effort [6]. myCBR focuses on the similarity-based retrieval step of the CBR cycle [19], providing sophisticated, knowledge-intensive similarity measures. Up to version 2, myCBR was a plugin for the ontology editor Protégé.

Protégé is an application for editing ontologies. Initially built for a few specialised programs in medical planning it has since then evolved into a much more general-purpose set of tools with a large community of users and contributors [20]. Currently there are two (albeit incompatible) versions of Protégé in active use, Protégé 3.x and Protégé 4.x. Protégé has been developed with a strong focus on being modular. To support the modularity goal the OSGi frame-

[4] `http://www.mycbr-project.net`

work[5] has been used as core plugin infrastructure, resulting in that all plugins are executed completely isolated from all other plugins and are only aware of the functionality offered by the API provided by Protégé 4. By implementing the correct interfaces and extending the correct classes it is possible to add tabs, renderers, views and many other both visible and background components. All description logic reasoners are for instance implemented as plugins.

The major change in Protégé's software architecture made *myCBR 2* incompatible with Protégé 4. This and the demand for an SDK triggered the development of the *myCBR 3* SDK and GUI. The SDK provides a completely rewritten foundation for the OSGi-based GUI. The work reported here shows one way of using the *myCBR 3* SDK.

3 Explanation-aware Case-based Reasoner (myEACBR)

Protégé is based on the idea that as much functionality as possible should be implemented as plugins. The "plugin part" of our implementation is defined by three files. The only source file we have in this regard is `RetrievalComponent` which extends `AbstractOWLViewComponent`, which is roughly equivalent to JPanel in Swing. It allows one to add components to be viewed on the screen. `RetrievalComponent` contains all of our components, both the ones that are shown on screen as well as the "back-end" objects.

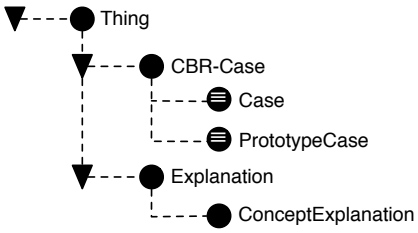

Fig. 1. The CBR ontology

Representing cases in OWL has been implemented by constructing a very small and simple CBR ontology (Figure 1). The ontology consists of the class *CBR-Case*, which contains two sub-classes *Case* and *PrototypeCase*. The two most notable classes here are the *PrototypeCase* and *ConceptExplanation*. The class *PrototypeCase* is the prototype for a particular type of case, thereby defining the most frequent attributes in that kind of case. The *ConceptExplanation* is the location where generated explanations are stored for later use. A concept has the following properties:

hasConceptName Distinct name of the concept explained;
hasExplanationSource Name of the class that constructed the explanation;

[5] `http://www.osgi.org/`

hasLink Link to an Internet resource used when constructing the explanation;
hasTextualExplanation The textual explanation itself.

To test our implementation we use the dinner domain. We have created an ontology importing the CBR ontology introduced in the above section and a wine and food ontology provided by W3C.[6] The two ontologies have been imported into a new ontology with the shared namespace.

The top-level of the resulting ontology is shown in Figure 2. In addition to these two imports the ontology contains two primitive classes, `DinnerCase`, as a sub class of `CBR-Case`, and `Person`, as a sub class of `Thing`. Instances of `DinnerCase` are supposed to be cases of dinners that have been recorded, and instances of `Person` are normal persons that for instance have eaten a dinner. An instance of `DinnerCase` typically has data properties and object properties representing the person(s) that has eaten the dinner, where it was eaten, what was eaten, the type of beverages and so on.

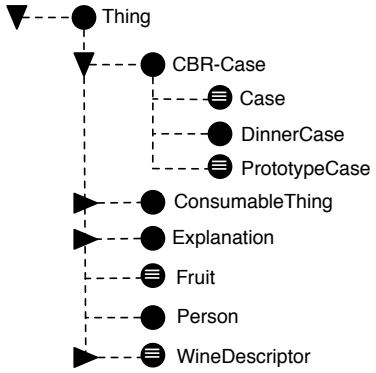

Fig. 2. Extract of the dinner ontology

We chose to create cases directly in Protégé instead of creating a separate interface for this in our plugin. This is done by creating instances of `DinnerCase` and assigning properties to the instance.

Before the user has added any attributes to the query it has attributes for the ID, always being "a query"; the author, defaulting to the system user name; and the time when the query was initialised. The next thing to specify is what concept the property should belong to, done by selecting a class from the class hierarchy. If the attribute is supposed to represent the person eating the dinner, Person is the appropriate class to choose in this step. After this the name of the attribute has to be specified in a standard input field. To represent the person eating dinner this has to be `eatenBy`, as this is the name we have used in all our test cases.

[6] `http://www.w3.org/TR/owl-guide/wine.rdf`

What happens after the name has been specified depends on the type of attribute specified. For most attribute types a standard input field is used to input the desired value. This dialogue accepts only certain input based on attribute type, so that if an invalid value (e.g. "food" in a field for an integer attribute) is specified the input field reappears. The input field is shown until either the value is valid or "cancel" is pressed. When attribute type Instance is selected a dialogue showing all defined instances of the specified concept is shown. When the query is fully specified "Execute query" is pressed, and the query is executed. The top queries are then listed in the middle of the screen ordered by their similarity with the query.

3.1 Protégé Explanations

In our test ontology, most dinner cases are inferred to be members of the `MealCourse` class as well as `DinnerCase`. Protégé will explain this by showing that the dinner case is a member of `DinnerCourse` because it has the property `hasDrink`, which is specified to have the domain of `MealCourse`. It has also been inferred that `MealCourse` is disjoint with the class `NonConsumable`. The explanation provided is that `MealCourse` is a subclass of `ConsumableThing`, and `NonConsumable` is equivalent to the complement of this class.

This type of explanation given by Protégé is not of very much use for a novice user. The explanations are simply a trace of the axioms that result in the inferred statement that we have requested an explanation for. We do however doubt that the intent of these explanations is to explain things to novice users. It is more likely that they are intended for debugging, and perhaps understanding, the active ontology, and it is not very likely that persons with these goals are novices.

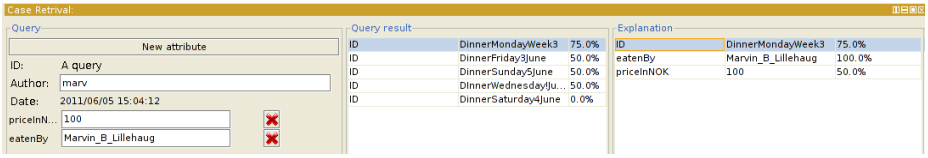

Fig. 3. Case similarity explanation

Similarity Explanations The ability to explain the similarities between query and instance comes mainly from the improvements done to myCBR 3. We think it is useful for the user to be able to see a decomposition of the similarity computation, such that he can better understand how the attributes he specifies in the query affects the result. Similarity explanations are cognitive (or action) explanations (see Figure 3 for an explanation of case similarity).

It is possible to view the selected case's attributes. Thereby its similarity explanation can be shown in a separate panel on the screen. The panel shows the

same information as in the query result, the case name and the total similarity. It also shows each attribute specified in the query and the similarity between its values in the query and the case in the selected case instance.

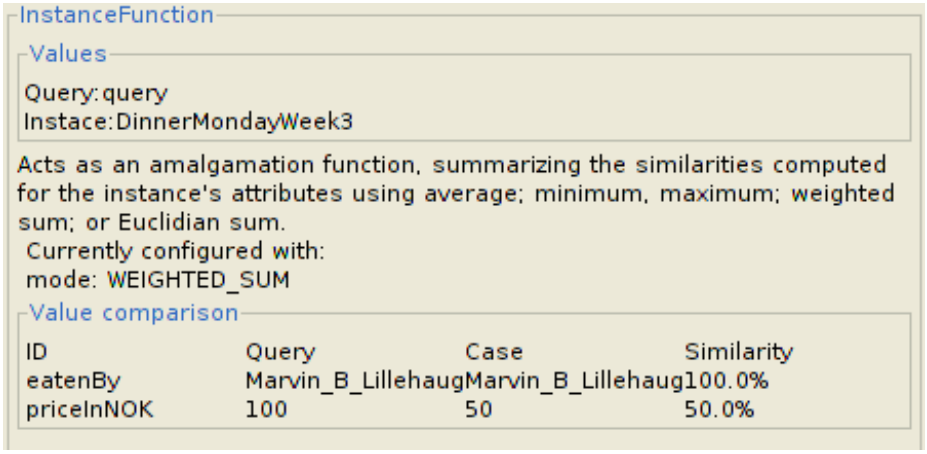

Fig. 4. Justification explanation

In addition to show the query and case instance attribute values with their similarity in Figure 4, an explanation of how the similarity metric works is shown. We can see that the `InstanceFunction` is a function that simply computes an aggregate of the similarity values of the instance's attributes using a configured method. We can also see that the shown function is configured to compute a weighted sum of the similarities when compared to a query.

Concept Explanation myEACBR provides explanation for the concept the selected attribute belongs to. For an example, the two concepts, `Person` and `Cost` for `eatenBy` and `priceInNOK` respectively. The concept explanations originate from both online and offline sources, providing several explanations for each concept given that each source contains information about it. We have implemented four knowledge sources for concepts that we now will describe.

Wordnet is a lexical database containing more than 118,000 different word forms and more than 90,000 different word senses and semantic relations between words.[7] We do however not make use of all these relations, only the description and synonyms of the words being names of the concepts to be explain.

When the user demands a concept explanation the Wordnet dictionary is queried, resulting in zero or more word definitions. When the result contains more than one word, the user is given the ability to choose which meaning of the word should be used. Both the definition of the meaning of the word and

[7] `http://wordnet.princeton.edu`

its synonyms are shown in this dialogue. Since there are other relations between words stored in Wordnet it would be possible to display more information about each word, but this was not prioritised as high as other matters in the project.

Fig. 5. Concept explanations

Wikipedia and Wiktionary The explanations we construct from Wikipedia and Wiktionary[8] are simply the text that is available on the page with the same name as the concept. This could be enhanced by providing some option to the user to change to a different page when there are alternative pages with topics sharing a name, but the API provided did not have such functionality "out of the box" so this feature would have been quite complex to implement.

Wolfram Alpha is a "computational knowledge engine", matching queries to a controlled library and computing answers and relevant visualisations from a core knowledge base of curated, structured data[9]. The results returned when queried contains the meaning which the engine assumes was intended by the query as well as a number of "pods", each containing some bit of information about the subject. An API to query Wolfram Alpha by other means than through a browser is available and is free, but the number of queries available is limited to 2,000

[8] `wikipedia.org`, `wiktionary.org`
[9] `http://www.wolframalpha.com`

per month. It provides the data in several formats and provides an easy way to navigate to alternative interpretations of the query in our plugin.

Figure 5 depicts a collage of the three different concept explanations arriving from Wordnet, Wikipedia and Wolfram Alpha.

3.2 Explanation Provenance

To trust an explanation it is important to be aware of its provenance, where it came from. This is important for both the experts behind the system as well as the users of it. When the experts need to verify the explanations given, the system needs to be able to tell how the answers were derived such that the experts know that it was not based on a bug or some other anomaly. The same reason applies for the users of the system, they need to know that the knowledge used originates from a trusted knowledge source and that valid methods are used for computing answers from these sources.

To some degree this is the same as the cognitive and justification explanation, showing the steps gone through when computing the answer and why this method was used. In our plugin and *myCBR 3* it is possible to examine the whole tree of similarities resulting in the final similarity value, and the similarity metrics underlying the computation is described in the justification explanations. In addition to this the source giving each particular concept explanation is shown, as well as a link showing where to obtain a similar explanation of the concept in a browser.

In the case of Wikipedia, Wiktionary, and Wolfram Alpha these links will show exactly the same explanation as shown in our plugin. In the case of Wordnet we were not able to find a method for specifying which of the different meanings of a word is required.

A comment we can make on the three former knowledge sources is that they themselves have quite good provenance. Wolfram Alpha does not list specific references for the different items in their result, but provide information about the sources for each topic of knowledge as a link ("Source information"). For all Wikipedia entries it is possible to view the history of everyone that has contributed to the entry and what they have contributed. We will not go into the discussion of how reliable this information is since this is far outside the scope of the project.

4 Discussion

In evaluating our approach we followed a staged model for AI research as described by Cohen and Howe [21]. The model comprises five steps: Refine a topic to a task and a view of how to accomplish the task; refine the view into a specific method; implement the method; design experiments to test the implementation; and run the experiments. As the authors point out this an idealised approach. Nevertheless, the criteria presented are beneficial to keep in mind regardless of the type of project at hand. Since we focused on generating explanations we do

not have any formal tests that we have run in order to evaluate our results. We have created a system that is capable of generating two kinds of explanations: similarity explanations and concept explanations.

4.1 Similarity Explanations

The similarity explanations we have presented are a combination of *action* and *justification* explanations aiming at *transparency*. The final similarity value can always be decomposed into its local similarity measurements. The justification comes from the fact that each explanation contains information about what method was used to calculate the similarity value. In particular the name of the similarity function as well as a description of how it works and what configuration parameters has been set.

There are some elements missing regarding the content of the similarity explanation, in particular regarding justification. Missing are for the most part visualisations; if present they would make the explanations easier to understand for a novice user, increasing the confidence in the system. It would, for example, be easier to understand how the `IntegerFunction` works if a graph was presented such that the user could see how the similarity values were distributed.

The explanations are however highly *relevant* (as long as the description of a similarity function is done properly). Since the function description is exclusively text, there may be some problems in cases where it contains concepts that should have been linked such that they could be explained.

The explanations, since they were constructed from local to global similarity measurements have high *fidelity*, *verification* and *duplication*, exposing what knowledge has been used to generate the explanation.

4.2 Concept Explanations

The concept explanations presented are not really constructed from a knowledge source within our system, but rather an aggregate of other knowledge sources outside the system. We merely fetch descriptions from other sources based on the name of the concept for which an explanation has been requested. That an explanation should be constructed from a particular knowledge source is of course not a requirement or criterion. It does however mean that the explanation given is not necessarily coherent, since it comprises up to four different explanations of the concept. Here it becomes visible how much the quality of concept explanations depends on the quality of the attached knowledge sources and on the abilities of the knowledge engineer in providing the knowledge.

We are undecided to whether the transparency and justification aspects of the concept explanations should be evaluated as good or poor. The system does not show the user how the explanations were generated, it only gives it to the user along with a link indicating where the online version might exist. It is indicated to the user where the information is originating from, and that several alternatives are offered when there are several entries with the same title. Because of this, one could argue that the explanation is transparent and justified since it is fairly

clear that the explanation is gathered from this knowledge source, and that this is the entry the particular source had for the concept in question. Again, this is quite dependent on the knowledge engineer's work. The concept explanations are however in most cases good explanations of what the concept in question is, and how it is used. The four sources we have used are quite good and reliable.

5 Summary and Future Work

In this paper we presented our work on a first step of enhancing a case-based reasoning engine with explanatory capabilities. We demonstrated how the retrieval phase in *myCBR 3* can be made explanation-aware by implementing a plugin for Protégé 4.x that generates explanations for the similarity values found in the retrieval step of the case-based reasoning cycle. In addition to this, the presented system is able to explain concepts used in the domain model by consulting external knowledge sources.

A next step will be to analyse requirements from the experience with this integration and to further enhance the explanation capabilities of the *myCBR 3* SDK and the *myCBR 3* GUI.

Acknowledgements

This work has partially been supported by the RAE-funded project "Explanation-aware myCBR development" at the University of West London.

References

1. Sørmo, F., Cassens, J., Aamodt, A.: Explanation in case-based reasoning – perspectives and goals. Artificial Intelligence Review **24**(2) (October 2005) 109–143
2. Leake, D.: Goal-based explanation evaluation. In: Goal-Driven Learning. MIT Press, Cambridge, MA (1995) 251–285
3. Roth-Berghofer, T., Adrian, B.: From provenance-awareness to explanation-awareness—when linked data is used for case acquisition from texts. In Marling, C., ed.: ICCBR 2010 Workshop Proceedings, Viale Teresa Michel 11, 15121 Alessandria, Italy, Dipartimento di Informatica Università del Piemonte Orientale "A. Avogadro" (July 2010) 103–106 TECHNICAL REPORT TR-INF-2010-06-03-UNIPMN.
4. Roth-Berghofer, T.: ExaCt manifesto: Explanation-aware computing (July 2009) `http://on-explanation.net/content/ExaCt_Manifesto.html`.
5. Lillehaug, M.B.: Explanation-aware case-based reasoning. Master's thesis, Department of Computer and Information Science, Norwegian University of Science and Technology (NTNU) (2011)
6. Stahl, A., Roth-Berghofer, T.R.: Rapid prototyping of CBR applications with the open source tool myCBR. In Bergmann, R., Althoff, K.D., eds.: Advances in Case-Based Reasoning, Springer Verlag (2008)
7. Ye, L.R., Johnson, P.E.: The impact of explanation facilities on user acceptance of expert systems advice. MIS Quarterly **19**(2) (June 1995) 157–172

8. Wahlster, W., Kobsa, A., eds.: User models in dialog systems. Springer Verlag (1989)
9. Kobsa, A.: User modeling in dialog systems: Potentials and hazards. AI & Society **4** (1990) 214–231
10. Schank, R.: Dynamic memory; a theory of reminding and learning in computers and people. Cambridge University Press (1982)
11. Schank, R.C.: Explanation Patterns – Understanding Mechanically and Creatively. Lawrence Erlbaum (1986)
12. Leake, D.B.: Evaluating Explanations: A Content Theory. Lawrence Erlbaum Associates (1992)
13. Roth-Berghofer, T.: Explanations and case-based reasoning: Foundational issues. In Funk, P., Gonzalez-Calero, P.A., eds.: Advances in Case-Based Reasoning, Springer-Verlag (2004) 389–403
14. Roth-Berghofer, T.R., Cassens, J.: Mapping goals and kinds of explanations to the knowledge containers of case-based reasoning systems. In Muñoz-Avila, H., Ricci, F., eds.: Case-Based Reasoning Research and Development, ICCBR 2005, Proceedings. Number 3620 in LNAI, Heidelberg, Springer Verlag (2005) 451–464
15. Kofod-Petersen, A., Aamodt, A.: Contextualised ambient intelligence through case-based reasoning. In Roth-Berghofer, T.R., Göker, M.H., Güvenir, H.A., eds.: Proceedings of the Eighth European Conference on Case-Based Reasoning (ECCBR 2006). Volume 4106 of Lecture Notes in Computer Science., Ölüdeniz, Turkey, Springer Verlag (September 2006) 211–225
16. Cassens, J., Kofod-Petersen, A.: Explanations and case-based reasoning in ambient intelligent systems. In Coyle, L., Schwarz, S., eds.: Case-Based Reasoning and Context-Awareness, The Seventh International Conference on Case-Based Reasoning (ICCBR '07), Workshop Proceedings, Belfast, Northern Ireland, University of Ulster (August 2007) 167–176
17. Aamodt, A.: Knowledge-intensive case-based reasoning in CREEK. In: Advances in Case-Based Reasoning: Proceedings ECCBR 2004. (2004) 1–15
18. Kofod-Petersen, A., Cassens, J., Aamodt, A.: Explanatory capabilities in the creek knowledge-intensive case-based reasoner. In Holst, A., Kreuger, P., Funk, P., eds.: Tenth Scandinavian Conference on Artificial Intelligence (SCAI 2008). Volume 173 of Frontiers in Artificial Intelligence and Applications., Stockholm, Sweden, IOS Press (May 2008) 28–35
19. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications **7**(1) (March 1994) 39–59
20. Gennari, J.H., Musen, M.A., Fergerson, R.W., Grosso, W.E., Crubézy, M., Eriksson, H., Noy, N.F., Tu, S.W.: The evolution of Protégé an environment for knowledge-based systems development. Int. J. Hum.-Comput. Stud. **58**(1) (2003) 89–123
21. Cohen, P.R., Howe, A.E.: How evauation guides AI research. AI Magazine **9**(4) (1988) 35–43