Man-Machine Collaboration to Acquire Cooking Adaptation Knowledge for the TAAABLE Case-Based Reasoning System

Amélie Cordier
Université de Lyon, CNRS
Université Lyon 1, LIRIS,
UMR 5205, F-69622, Lyon,
France
Amelie.Cordier@liris.cnrs.fr

Emmanuelle Gaillard
LORIA, Université de Lorraine
BP 239, F-54506
Vandœuvre-lès-Nancy,
CEDEX, France
Emmanuelle.Gaillard@loria.fr

Emmanuel Nauer
Université de Lorraine, LORIA
BP 239, F-54506
Vandœuvre-lès-Nancy
CEDEX, France
Emmanuel.Nauer@loria.fr

ABSTRACT

This paper shows how humans and machines can better collaborate to acquire adaptation knowledge (AK) in the framework of a case-based reasoning (CBR) system whose knowledge is encoded in a semantic wiki. Automatic processes like the CBR reasoning process itself, or specific tools for acquiring AK are integrated as wiki extensions. These tools and processes are combined on purpose to collect AK. Users are at the center of our approach, as they are in a classical wiki, but they will now benefit from automatic tools for helping them to feed the wiki. In particular, the CBR system, which is currently only a consumer for the knowledge encoded in the semantic wiki, will also be used for producing knowledge for the wiki. A use case in the domain of cooking is given to exemplify the man-machine collaboration.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*knowledge acquisition*; H.2.8 [**Database Management**]: Database Applications—*Data mining*; H.3.3 [**Information Storage and Retrieval**]: Information Storage and Retrieval—*relevance feedback*

Keywords

Man-machine collaboration, adaptation knowledge acquisition, semantic wiki, case-based reasoning, knowledge discovery

1. INTRODUCTION

This paper presents an architecture for improving the collaboration between humans and machines in the framework of the system TAAABLE[1]. TAAABLE is a case-based reasoning (CBR) system which adapts cooking recipes to users constraints [1], using cooking knowledge encoded in a semantic wiki, called WIKITAAABLE[2] [7]. Storing the cooking knowledge into a wiki allows to better manage knowledge: users are fully involved in the knowledge acquisition part of the system instead of only using the TAAABLE system. The idea is that if users increase the quantity and the quality of the knowledge in the wiki, the results of the TAAABLE adaptation process improves. For this, automated processes, like the TAAABLE adaptation process and other knowledge extraction processes, can be used to feed the wiki with new knowledge. Our objective, in this paper, is to consider (1) the "knowledge" part (WIKITAAABLE) and the "reasoning" part (TAAABLE) of a CBR system as a whole, (2) that these two parts can mutually and continuously benefits one from the other, and (3) that users play a key role for managing the interactions between these two parts.

This paper proposes an architecture that improves the acquisition of *adaptation knowledge* (AK), a specific kind of knowledge a CBR system can use to improve its results. The TAAABLE/WIKITAAABLE cooking application is chosen as concrete use case. Section 2 gives the minimal requirements about this application context and the motivations for acquiring AK. Section 3 gives the architecture for the collaboration between automatic processes and users for improving the AK acquisition and shows how this architecture has been instantiate in our use case. Section 4 presents related works and section 5 concludes the paper and discusses ongoing works.

2. CONTEXT AND MOTIVATION

TAAABLE is a CBR system that has been designed for participating to the Computer Cooking Contest[3], an international contest which aims at comparing CBR system results on a common domain: cooking. Several challenges are proposed in this contest. Among them, two challenges (won by TAAABLE in 2010):

- the *main challenge*, asking CBR systems to return recipes satisfying a set of constraints given by the user, such as inclusion or rejection of ingredients, the type or the origin of the dish, the compatibility with some diets (vegetarian, nut-free, etc.). For example: a user may ask for "a dessert, with rice and fig", as illustrated in Figure 1. Systems have to search into a set of limited (approximately 1500) recipes for recipes satisfying the constraints, and if there is no recipe satisfying all the constraints, the systems have to adapt existing recipes into new ones.

[1] http://taaable.fr/
[2] http://wikitaaable.loria.fr/
[3] http://computercookingcontest.net/

WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

- the *adaptation challenge*, asking CBR systems to adapt a given recipe to specific constraints. For example, "adapt the *My strawberry pie* recipe because I do not have strawberry".

2.1 TAAABLE principles

Figure 1: The TAAABLE interface. Queried for a dessert dish, with rice and fig, TAAABLE proposes to replace mango by fig in the "Glutinous rice with mangoes" recipe. After viewing the adapted recipe, the user can give feedback about the substitution ("OK" or "not OK").

Like many CBR systems [16], TAAABLE uses an ontology to retrieve the source cases that are the most similar to a target case (i.e. the query). TAAABLE retrieves and creates cooking recipes by adaptation. According to the user constraints, the system looks up, in the recipe base (which is a case base), whether some recipes satisfy these constraints. Recipes, if they exist, are returned to the user; otherwise the system is able to retrieve similar recipes (i.e. recipes that match the target query partially) and adapts these recipes, creating new ones. Searching similar recipes is guided by several ontologies, i.e. hierarchies of classes (ingredient hierarchy, dish type hierarchy, etc.), in order to relax constraints by generalising the user query. The goal is to find the most specific generalisation (with the minimal cost) for which recipes exist in the case base. Adaptation consists of substituting some ingredients of the source cases by the ones required by the user. It is important to notice that TAAABLE is based on an hypothetical reasoning and not deductive one. So, it is possible that some adaptations proposed by the system may be irrelevant (i.e. not cookable).

For dealing with the adaptation of a specific recipe (which is the *adaptation challenge* problem), TAAABLE uses the same hierarchy based generalisation/specialisation mechanism on a recipe base containing only the recipe that has to adapted. For example, when adapting the *"My Strawberry Pie"* recipe (in which strawberries are required) to the constraint "no strawberry", `Strawberry` is generalised on `Berry`, which is then specialised in another berry (`Raspberry`, `Blueberry`, `Blackberry`, etc.). Substitutions (e.g substitute `Strawberry` by `Raspberry`) are proposed to the user.

Again, this hierarchy-based adaptation can produce bad substitutions. Indeed, adding a new ingredient in a recipe may be incompatible with some other ingredients(s) of the recipe and/or may require the addition of new ingredient(s). Another problem, when adapting a given recipe is that TAAABLE is not able to choose, during the specialisation step, which ingredient is the best for substituting another one. In the previous example, TAAABLE is not able to determine whether it is better to replace `Raspberry` by `Strawberry`, by `Blueberry` or by any other berry. This lack has been corrected by building a special knowledge discovery process which is detailed in section 2.4.

2.2 WIKITAAABLE

WIKITAAABLE is a semantic wiki that uses Semantic MediaWiki [12] as support for encoding knowledge associated to wiki pages. WIKITAAABLE contains the set of resources required by the TAAABLE reasoning system, in particular: an ontology of the domain of cooking, and recipes.

Figure 2: The Berry concept in WIKITAAABLE.

The cooking ontology is composed of 6 hierarchies: a *food* hierarchy (related to ingredients used in recipes, e.g. `Berry`, `Meat`, etc.), a *dish type* hierarchy (related to the types of recipes, e.g. `PieDish`, `Salad`, etc.), a *dish moment* hierarchy (related to when eating a dish, e.g. `Snack`, `Starter`, `Dessert`, etc.), a *location* hierarchy (related to the origins of recipes, e.g. `France`, `Asia`, etc.), a *diet* hierarchy (related to food allowed or not for a specific diet, e.g `Vegetarian`, `NutFree`, etc.), an *action* hierarchy (related to cooking ac-

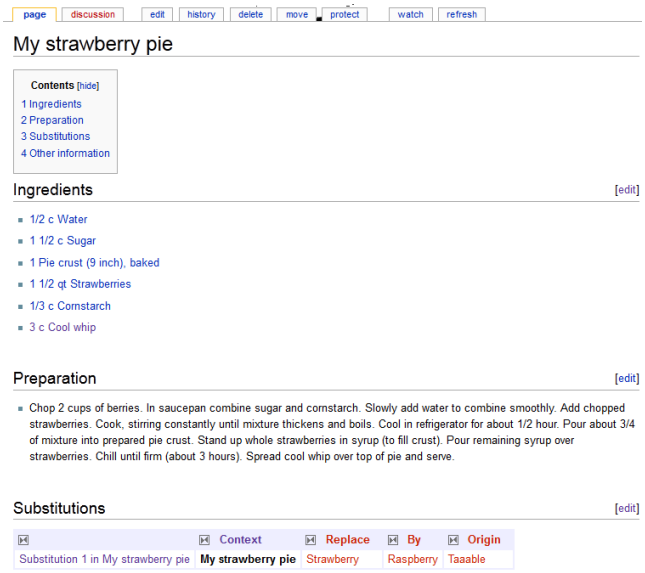

Figure 3: Example of a WIKITAAABLE recipe.

tions used for preparing ingredients, `toCut`, `toPeel`, etc.). In the semantic wiki, each concept of a hierarchy is encoded as a category page $Category : < conceptname >$. For example, the concept of berry is encoded in the `Category:Berry` page, as shown in Figure 2. Each concept is described by a short description, lexical variants (used by the annotation bot, for searching concepts in the full text of recipes), his sub-categories and super-categories. For berries, the wiki page indicates that `Berry` is a sub-concept of `Fruit` (corresponding to the $Category : Fruit$ page), and sub-categories of `Berry` (e.g. `Raspberry`, `Blueberry`, etc.) are listed.

The set of recipes contained in WIKITAAABLE are those provided by the contest, that have been semantically annotated according to the domain ontology. Each recipe, as the one given in example in Figure 3, is encoded as a wiki page, composed of several sections: a title, which is the name of the recipe, an "*Ingredients*" section containing the list of ingredients used in the recipe, each ingredient being linked to its corresponding *Category* page in the food hierarchy, a "*Textual Preparation*" section describing the preparation process, some possible "*substitutions*" which are adaptation knowledge, and "*other information*" like the dish type, for example.

2.3 Adaptation knowledge

Improving the current results of TAAABLE could be done by acquiring *adaptation knowledge* (AK). In CBR systems, using AK is a classical approach for producing more fine grain adaptations [15]. In the TAAABLE/WIKITAAABLE context, an AK is a substitution of some ingredients by other ones (e.g. in *"My Strawberry Pie"* recipe, `Strawberry` could be replaced by `Raspberry`). Formally, an adaptation knowledge is a 4-tuple $(context, replace, by, origin)$, where:

- *context* represents the recipe or the class of recipes on which the substitution can be applied. An AK is specific if its *context* is a single recipe and generic if its *context* is a class of recipes (a specific type of dish, for example). In this paper, only specific AK are considered.
- *replace* and *by* are respectively the set of ingredients that must be replaced and the set of replacing ingredients.
- *origin* is the source the AK comes from. Currently, four sources have been identified:
  1. TAAABLE, when AK results from a proposition of adaptation given by the reasoning process of TAAABLE.
  2. *AK extractor*, when AK results from a specific knowledge discovery system called "AK EXTRACTOR" (which is detailed in the next section).
  3. *user*, when AK is given by a user by editing the wiki, as it is usually done in cooking web site, when users add comments about ingredient substitution on a recipe. See, for example, `http://en.wikibooks.org/wiki/Cookbook:substitutions`.
  4. *recipe*, when the AK is directly given by the original recipe when a choice between ingredients is mentioned (e.g. "100g butter or margarine"). This particular substitutions are taken into account by a wiki bot which runs through the wiki for automatically extracting them.

Figure 4: Example of a substitution page.

According to this definition, (*"My Strawberry Pie"*, *Strawberry*, *Raspberry*, TAAABLE), is an AK obtained from TAAABLE, meaning that strawberries can be replaced by raspberries in the *"My Strawberry Pie"* recipe. In WIKITAAABLE, each substitution is encoded as a wiki page like the one given in Figure 4. A semantic query is used to feed automatically the *Substitutions* section of a recipe page, as visible in Figure 3.

2.4 Adaptation knowledge extractor

Semi-automatic extraction of AK for adapting cooking recipe has been studied in [10]. An operational system that proposes ingredient substitutions for adapting

a given recipe to additional constraints has been implemented. Figure 5 shows an example of results for adapting the *"My Strawberry Pie"* recipe without strawberry. AK Extractor proposes a set of substitutions like replacing `Strawberry` and `CoolWhip` by `Raspberry` and `FoodColor`, or by `Peach` and `LemonJuice`, etc.

Figure 5: An illustration of the AK extractor interface

AK Extractor was built in order to acquire AK, taking into account incompatibly of ingredients or ingredients often associated in recipes, overcoming the weakness of Taaable. The AK Extractor is founded on a typical knowledge discovery in databases (KDD) process. The main steps of a KDD process are data selection and preparation, datamining, and interpretation of the extracted units of information [9]. The step of interpretation requires usually an expert of the domain for validating the extracted units for giving them a status of knowledge (for cooking, many web users could be considered as experts). In traditional KDD approaches, the KDD process produces a huge amount of information units which can be turned into knowledge. Experts have then to process these information units in order to validate the "good" ones, which become knowledge. This is a tedious task for experts. In addition, they face situations where they cannot decide if an information unit is appropriate or not because they lack context information to decide. By contrast, the AK approach we describe in this paper is interactive and opportunistic [5]. Users use the same interface for querying the system and for validating knowledge. In addition, they validate knowledge on the fly, in context, which is more convenient than dealing with a huge amount of candidate knowledge (information units) out of any context. AK Extractor is based on a comparison of ingredients between the recipe to adapt, and a set of similar recipes. The system selects first recipes which have a minimal number of ingredients in common and a minimal number of different ingredients with the recipe that has to be adapted. Closed itemsets [11] are extracted from variations between recipes, starting from variations between each selected recipe of the previous step and the recipe to adapt. The closed itemsets are then filtered and ranked with specific rules (which are detailed in [10]). The system displays the propositions for ingredient substitutions coming from the first five better ranked closed itemsets. Finally, user can give feedback for validating or not some of these propositions.

2.5 Objectives

In a semantic wiki, knowledge is in continuous evolution. Usually, this evolution results from users that feed the wiki. The main objective of this work is to increase the acquisition of AK by establishing a collaborative environment between users and automatic processes supported by machines. An interesting point is that the reasoning system (Taaable) is not only seen as the consumer of the wiki knowledge but also as the producer of new knowledge that will be added to the wiki.

Automatic processes and collective intelligence must be coordinated. The better machines and users will collaborate, the better the acquisition process succeed. Indeed, machines are able to compute lot of things (for example, substitution propositions for Taaable and AK Extractor) but the results cannot be considered as knowledge before they are validated as such by a human expert. Indeed, the crucial step of the KDD process [9] is that the results produced by the machine must be interpreted and validated by a (human) expert. So, users will be asked to play the role of the expert for validating automatic results produced by machine and a proposition of substitution becomes an AK only when it is validated by a user.

The next section details our approach for enhancing the collaboration between humans and machines, by plugging together automatic systems (Taaable, WikiTaaable and AK Extractor) under the control of users.

Increasing the quantity and quality of AK in the wiki will improve the results of the reasoning system. However, this requires a smooth modification of the reasoning process for taking into account the AK for computing adaptation (this is a short-term work). Besides, to ensure the non-regression of the system, the impact of the continuously new AK produced on the system results must be evaluated by test sets. In this paper, we focus on AK acquisition. A way for evaluating how knowledge evolution can be managed for improving the results of a reasoning process is studied in [18].

3. COLLABORATIVE ENVIRONMENT FOR ACQUIRING AK FROM VARIOUS SOURCES

3.1 The approach

Currently, users are already involved in Taaable, WikiTaaable and AK Extractor systems when they use them, but these three systems are separated. If they were integrated, each could benefit from the others. Some interactions between these systems have to be developed. For example, Taaable and AK Extractor return substitution propositions, and users can validate these propositions as being relevant or not. Each system has is own storage for taking into account the user feedbacks, but there is no common space for storing together all the AK that can be produced. The architecture proposed in Figure 6 aims at plugging together the three systems (WikiTaaable, Taaable and AK Extractor) and WikiTaaable will be used to store the results of AK acquisition produced by humans as well as produced by machines.

Machines and humans will work together for a common objective which is AK acquisition. In our approach, users are essential for controlling and integrating AK produced by

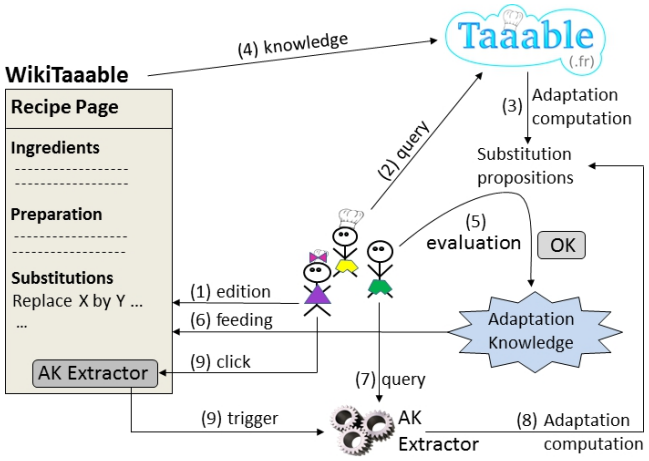

Figure 6: Architecture for a man-machine AK acquisition in the cooking domain.

automatic processes in WikiTaaable. Indeed, additionally to their wiki editing, users will trigger automatic processes and validate their results. They play the role of cooking expert when validating substitution propositions as AK being stored in the wiki. Besides, different users can, in parallel, start Taaable, AK Extractor on a specific recipe page, or edit new substitution pages in the wiki. The new AK integrated in the wiki will be used, at short term, by the reasoning process of Taaable. Thus, this loop ensures the continuous knowledge building and the improvement of the results in Taaable.

Figure 6 shows the interaction between the different resources and their integration into a global system. In this system, a user can (1) directly edit a wiki substitution page for adding, removing, modifying AK.

When a user (2) queries Taaable, (3) Taaable computes adaptation propositions thanks to (4) knowledge stored in WikiTaaable. These propositions are presented to the user who has (5) to evaluate whether a proposition is relevant or not. If the proposition is relevant (i.e. "OK" has been activated), this proposition becomes an AK which will automatically (6) feed a new substitution page, which context is the recipe concerned by the adaptation.

A similar process holds when a user (7) queries the AK Extractor, which (8) computes substitution propositions, submitted to the user (5) evaluation and which will (6) feed the wiki with AK if relevant.

Finally, the possibility of (9) trigger the AK Extractor for a given recipe is added in all recipe pages. In that case, the sequence (8,5,6) takes place again.

The communication between Taaable and WikiTaaable is performed through the RDF store linked to Semantic Media Wiki. On one side, Taaable, as well as AK Extractor, read RDF data from this store when using knowledge of the wiki. On the other side, results provided by Taaable and AK Extractor are written as RDF triple, for feeding the RDF store, and, by extension, the wiki.

Even if the architecture is presented in a specific domain with specific systems, this kind of organisation can be reproduced in various domain, because it is a generic organisation. Indeed, inference engines, KDD engines and wikis can collaborate together under the control of users, who play a key role in the organisation.

3.2 Implementation of the architecture in Taaable/WikiTaaable

The proposed architecture has been instantiated to integrate Taaable feedback and the AK Extractor process in WikiTaaable. Some screenshots are given to show how it looks like in the wiki. Figures 7, 8, and 9 illustrate how the AK Extractor process has been integrated as a wiki extension. Figure 7 shows the first step of the AK Extractor process in which the user has to give constraints on the recipe to adapt. This interface is triggered by a special link that has been added on each recipe page. The interface given in example was obtained from the *"My Strawberry Pie"* recipe page. The user enters constraints (e.g. `−Strawberry` means that the user does not want strawberries).

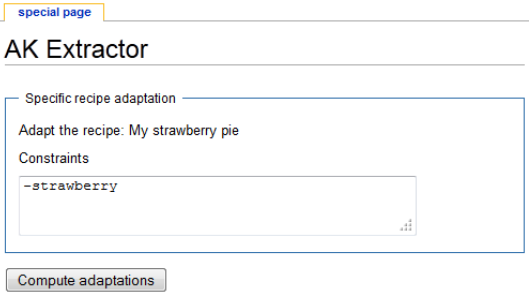

Figure 7: The AK Extractor query interface.

When clicking on "Compute adaptations", the AK discovery process is run. This process produces a wiki page with some AK propositions, as it is illustrated in Figure 8. The user may validate some proposition(s) as being relevant by clicking on "OK". Optionally, some adjustments could be made thanks to the checkboxes: if not checked, an ingredient will not take part of the substitution[4]. Figure 9 shows the page resulting from the validation of the first proposition of substitution. This page announces that a substitution page, like the one presented in Figure 4, has been created and that this substitution will automatically appear in the recipe page as an AK, as illustrated in Figure 10.

4. RELATED WORK

Previous works deal with man-machine collaboration where knowledge is obtained by a knowledge extraction process which is guided or/and validated by human. In this section, we present several systems based on KDD or on CBR or even both that demonstrate this collaboration.

4.1 KDD systems

The KDD process requires to be supervised by an expert, who could interact at various levels. One of the most usual problems in a KDD is to control the over-abundance of results generated by the KDD process. The expert could, for example, interact for better selecting the data that will be mined as it is described in [4] which proposes an approach

[4]the "not OK" button does not produce anything yet.

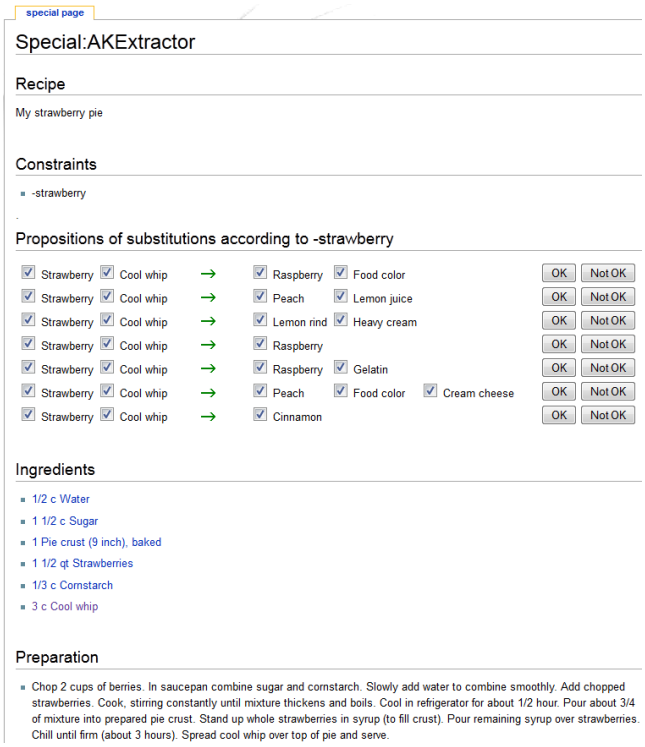

Figure 8: AK propositions computed by the AK Extractor.

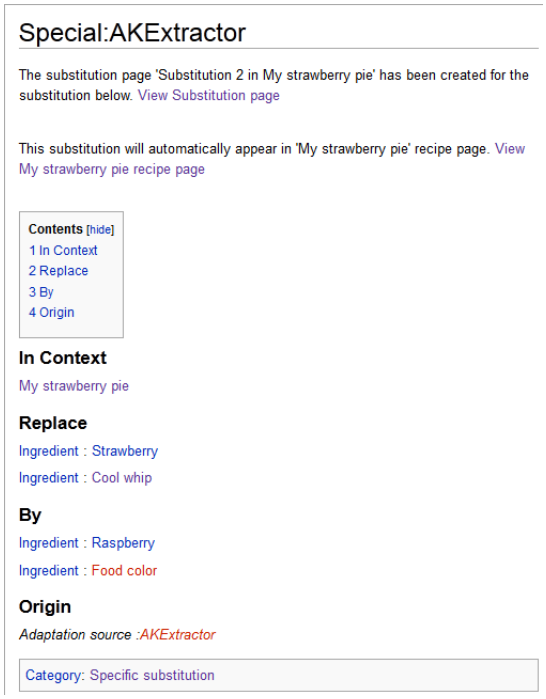

Figure 9: Generating the AK substitution page.

Substitutions

	Context	Replace	By	Origin
Substitution 1 in My strawberry pie	**My strawberry pie**	Strawberry	Raspberry	Taaable
Substitution 2 in My strawberry pie	**My strawberry pie**	Strawberry Cool whip	Raspberry Food color	AKExtractor

Figure 10: *"My Strawberry Pie"* substitutions section after the acquisition of a new substitution.

for optimising the formulation of the problem to solve. Another approach consists in filtering and ranking of numerous results obtained by the data mining algorithms. For example, [17] proposes subjective measures of interestingness for evaluating the datamining results. These measures depend on the user profile and a result is considered as interesting for a user according to two major reasons unexpectedness (if the user is "surprised" by the results) and actionability (if the user can exploit the results). The paper focuses on unexpected results which are results in contradiction with beliefs of the user. So, a result which may revise beliefs of a user is relevant.

Another usual problem of the KDD process is the selection of relevant information units among the large set of information units produced, for transforming them into knowledge. Many approaches for taking into account this step of the KDD process have been proposed. For example, [3] presents a methodology for KDD in the context of building a semi-automatic ontology from heterogeneous textual resources. [3] uses formal concept analysis (FCA) [11] for classifying objects according to their properties which are extracted from various textual resources (e.g thesaurus, full texts, dictionaries, etc.). The results of the FCA process is translated in description logics for representing the ontology concepts. Experts are involved at each step of the KDD process. For example, when the properties describing the objects are produced by an automatic extraction process, experts have to validate and filter the most representative properties, i.e. that described the best the objects. At the last step of the process, experts have to validate the formal concepts that have been produced, by selecting those which makes sense in their domain.

The integration of the AK Extractor follows the same principle. AK Extractor implements a KDD process which produces a set of substitution propositions. These substitutions have to be validated in order to be stored as AK.

4.2 CBR systems

CBR [16] is a method for solving new problems thanks to adaptation of previously solved problems. However, the step of adaptation may fail. Consequently, CBR systems integrating users are built in order to repair irrelevant adaptations. The systems presented in the following involve user in an opportunistic way, the goal is to repair adaptations that have failed, by acquiring AK or by increasing the domain knowledge (on which the reasoning is based) itself.

DIAL [14] focuses on an interactive acquisition of AK in the domain of disaster response planning. When the system returns a solution that is inconsistent, the response planning is returned with a description of the elements that need to be adjusted in the planning. For example, a response planning for an earthquake in Los Angeles indicates that National Guard must be called. When this plan is used for an earth-

quake in Indonesia, a problem arises because there is not National Guard in Indonesia. So, the response plan must be adapted. DIAL is composed of three kinds of adaptation process:

- case-based adaptation: which reuses adaptations which previously performed,
- rule-based adaptation: which determines elements that have to be adapted and the transformation to apply (e.g substitute an element) by the generation of a generic rule. The generic rule is then instantiated by searching relevant knowledge in the knowledge base. In the previous example, it consists in building a rule meaning that national guard must be substituted, and instantiate this rule by a substituting element searched in the knowledge base,
- manual adaptation: from an interface, a user may select a generic transformation to apply and navigate in the knowledge base to search relevant knowledge for instantiating the generic transformation. This third adaptation method is a man-machine collaboration for AK acquisition triggered opportunistically. Indeed, in case of failure of the case-based and rule-based adaptations, the system proposes the user to guide the adaptation process. After being solved, the AK is memorised. An AK corresponds to the transformation applied (e.g substitute an element) with the links in the knowledge base toward relevant knowledge uses to make the transformation.

In DIAL, adaptation is manually transformed if automatic adaptations fail. Conversely, WebAdapt [13] proposes two independent adaptations modes: one automatic and one manual. WebAdapt is a system for personalising adaptations based on learned information about user preferences. WebAdapt is used in the domain of sightseeing itinerary planning. The system adapts a proposed itinerary, corresponding to a list of locations, with one of these two independent adaptation modes. Either, the user of WebAdapt indicates a location to substitute and if the location must be substitute by criteria of similarity or criteria of nearness. Then, WebAdapt computes an automatic adaptation without intervention of the user. Either, the itinerary proposed by the process is customized by user preferences. The user selects constraints (e.g. "I would like to visit building of the 16th century") in order to adapt locations proposed by the system.

The FRAKAS system [5] is also such an opportunistic system, which integrates interactions with an expert during the reasoning for acquiring missing domain knowledge and ensure a better adaptation. The adaptation proposed by the system is evaluated by an expert. From an interface, the expert may detect some inconsistencies between the proposed adaptation and its own knowledge. If an inconsistency is detected, the adaptation failed. At this moment, user completes the domain knowledge by selecting incompatible problem variables and/or solutions variables. Adapted cases and domain knowledge are acquired at the same time during the interactive part between the expert and the system.

Like [6], [2] uses an opportunistic approach for AK acquisition. In the second version of TAAABLE, which implements the approach proposed in [2], users may give some feedback on substitutions for adapting recipes. If a proposition of substitution is judged irrelevant by the user, an interface allows to guide the system for repairing the adaptation. The user may indicate that some ingredient(s) is/are missing when adding an ingredient, or that some ingredients of the recipe are not compatible with an ingredient that must be added. In the case where is ingredient(s) missing, a system of AK acquisition, called Cabamaka [8] based on KDD, is triggered. This last step allows to repair the bad adaptation and memorise the AK. The AK is a rule composed of ingredient(s) that have to be removed and ingredient(s) that have to be added, similar to a *substitution* AK used in WIKITAAABLE.

In each of the previous systems, knowledge acquisition is triggered when an adaptation failed. The originality of approach is that AK can be triggered in parallel of the adaptation process and that this knowledge can be acquired at any time in a semantic wiki collaborative space.

5. CONCLUSION

This paper proposes an architecture for improving the man-machine collaboration for collecting AK for a CBR system. Automatic processes are integrated under the control of users for feeding knowledge in a semantic wiki. In the specific context of a CBR system, this approach allows to better collect adaptation knowledge. The CBR engine, which was initially only a consumer of the wiki knowledge, has been turned as a producer as well. Other tools, and especially knowledge discovery ones, can be plugged as wiki extension for helping users to feed the wiki. A single interface is used for viewing, querying, enriching and correcting the knowledge. Moreover, as it is a collaborative wiki interface, everyone will benefit from the experience of the other members of the community.

As our approach produces specific AK (i.e. for adapting a specific recipe), an extension of this work concerns the study and the development of a KDD process for extracting generic adaptations (i.e. for a set of recipes, for a given dish type, for example). Another short-term work concerns the exploitation of irrelevant adaptation propositions. It was shown that when a user validates a adaptation proposition as being relevant, this adaptation becomes an AK and is stored in the wiki. However, when a proposition is irrelevant (i.e. the user disagrees with an automatic adaptation), nothing is currently done for collecting them nor for exploiting them. Some specific interactions with the user could be developed for integrating a process for dealing with rejected adaptations proposed by TAAABLE or AK EXTRACTOR.

6. ACKNOWLEDGEMENTS

This work is supported by French National Agency for Research (ANR), program Contint 2011. More information about Kolflow is available on the project website: `http://kolflow.univ-nantes.fr/`.

7. REFERENCES

[1] F. Badra, R. Bendaoud, R. Bentebitel, P.-A. Champin, J. Cojan, A. Cordier, S. Després, S. Jean-Daubias, J. Lieber, T. Meilender, A. Mille, E. Nauer, A. Napoli, and Y. Toussaint. Taaable: Text Mining, Ontology Engineering, and Hierarchical Classification for Textual Case-Based Cooking. In *ECCBR Workshops,*

Workshop of the First Computer Cooking Contest, pages 219–228, Trier, Germany, September 2008.

[2] F. Badra, A. Cordier, and J. Lieber. Opportunistic Adaptation Knowledge Discovery. In L. McGinty and D. C. Wilson, editors, *8th International Conference on Case-Based Reasoning - ICCBR 2009*, volume 5650 of *Lecture Notes in Computer Science*, pages 60–74, Seattle, United States, July 2009. Springer.

[3] R. Bendaoud, A. Napoli, and Y. Toussaint. Formal Concept Analysis: A unified framework for building and refining ontologies. In A. Gangemi and J. Euzenat, editors, *16th International Conference on Knowledge Engineering and Knowledge Management - EKAW 2008*, volume 5268 of *Lecture Notes in Artificial Intelligence*, pages 156–171, Acitrezza, Catania, Italie, 2008. Springer Berlin / Heidelberg.

[4] Z. Chen and Q. Zhu. Query construction for user-guided knowledge discovery in databases. *Inf. Sci.*, 109:49–64, August 1998.

[5] A. Cordier. *Interactive and Opportunistic Knowledge Acquisition in Case-Based Reasoning*. ThÃĺse de doctorat en informatique, UniversitÀl' Lyon 1, Nov. 2008.

[6] A. Cordier, B. Fuchs, J. Lieber, and A. Mille. Interactive Knowledge Acquisition in Case Based Reasoning. In D. Wilson and D. Khemani, editors, *Workshop on Knowledge Discovery and Similarity, a workshop of the seventh International Conference on Case-Based Reasoning (ICCBR-07), D. C. Wilson and D. Khemani (volume editors)*, Belfast, United Kingdom, August 2007. D. Wilson and D. Khemani.

[7] A. Cordier, J. Lieber, P. Molli, E. Nauer, H. Skaf-Molli, and Y. Toussaint. Wikitaaable: A semantic wiki as a blackboard for a textual case-based reasoning system. In *SemWiki 2009 – 4th Semantic Wiki Workshop*, Heraklion, Greece, May 2009.

[8] M. d'Aquin, F. Badra, S. Lafrogne, J. Lieber, A. Napoli, and L. Szathmary. Adaptation Knowledge Discovery from a Case Base. In Traverso, editor, *17th European Conference on Artificial Intelligence ECAI06*, Trento, Italy, August 2006. IOS Press.

[9] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, Menlo Park, CA, USA, 1996.

[10] E. Gaillard, J. Lieber, and E. Nauer. Adaptation knowledge discovery for cooking using closed itemset extraction. In *The Eighth International Conference on Concept Lattices and their Applications - CLA 2011*, pages 87–99, Nancy, France, Oct. 2011.

[11] B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, 1999.

[12] M. Krötzsch, D. Vrandecic, and M. Völkel. Semantic mediawiki. In I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *International Semantic Web Conference*, volume 4273 of *Lecture Notes in Computer Science*, pages 935–942. Springer, 2006.

[13] D. Leake and J. Powell. Mining Large-Scale Knowledge Sources for Case Adaptation Knowledge. In R. Weber and M. Richter, editors, *Proceedings of the Seventh International Conference on Case-Based Reasoning*, pages 209–223, Belfast, United-Kingdom, August 2007. Springer Verlag.

[14] D. B. Leake, A. Kinley, and D. C. Wilson. Acquiring Case Adaptation Knowledge: A Hybrid Approach. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 684–689, Belfast,Portland, Oregon, United States, August 1996. AAAI Press.

[15] R. Lopez De Mantaras, D. McSherry, D. Bridge, D. Leake, B. Smyth, S. Craw, B. Faltings, M. L. Maher, M. T. Cox, K. Forbus, M. Keane, A. Aamodt, and I. Watson. Retrieval, reuse, revision and retention in case-based reasoning. *Knowl. Eng. Rev.*, 20:215–240, September 2005.

[16] C. K. Riesbeck and R. C. Schank. *Inside Case-Based Reasoning*. Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey, 1989.

[17] A. Silberschatz and A. Tuzhilin. On subjective measures of interestingness in knowledge discovery. In *In Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 275–281, Montreal, Quebec, Canada, August 1995.

[18] H. Skaf-Molli, P. Molli, E. Desmontils, E. Nauer, Y. Toussaint, G. canals, A. Cordier, and M. Lefevre. Knowledge Continuous Integration Process (K-CIP). In *WWW2012 Workshop on Semantic Web Collaborative Spaces (SWCS2012)*, 2012.