# Knowledge Acquisition and Discovery for the Textual Case-Based Cooking system WIKITAAABLE

Fadi Badra[1], Julien Cojan[1], Amélie Cordier[2], Jean Lieber[1],
Thomas Meilender[1], Alain Mille[2], Pascal Molli[1], Emmanuel Nauer[1],
Amedeo Napoli[1], Hala Skaf-Molli[1], Yannick Toussaint[1]

[1]LORIA UMR 7503 CNRS, INRIA, Nancy Universities BP 239
54506 Vandœuvre-lès-Nancy, France, First-Name.Last-Name@loria.fr
[2]LIRIS CNRS, UMR 5202, Université Lyon 1, INSA Lyon, Université Lyon 2, ECL
43, bd du 11 Novembre 1918, Villeurbanne Cedex, France,
First-Name.Last-Name@liris.cnrs.fr

**Abstract.** The textual case-based cooking system WIKITAAABLE participates to the second Computer cooking contest (CCC). It is an extension of the TAAABLE system that has participated to the first CCC. WIKITAAABLE's architecture is composed of a semantic wiki used for the collaborative acquisition of knowledge (recipe, ontology, adaptation knowledge) and of a case-based inference engine using this knowledge for retrieving and adapting recipes. This architecture allows various modes of knowledge acquisition for case-based reasoning that are studied within the TAAABLE project. In particular, opportunistic adaptation knowledge discovery is an approach for interactive and semi-automatic learning of adaptation knowledge triggered by a feedback from the user.

**Keywords:** textual case-based reasoning, case-based cooking, semantic wiki, opportunistic knowledge acquisition
**URL of the system:** `http://taaable.fr` is the URL of a Web page with a hyperlink on the WIKITAAABLE system (and also a hyperlink on the TAAABLE system that has participated to the first CCC).

## 1 Introduction

As final result from last year did not make us good cooks, we decided to keep on doing research. Hence, for the second edition of the Computer Cooking Contest (CCC), the TAAABLE system has evolved towards a new architecture called WIKITAAABLE. This year, we focused our efforts on two intertwined aspects: knowledge and reasoning. Concerning reasoning, we worked on the inference engine improvement to enhance the adaptation ability of the system. Concerning knowledge we set up advanced knowledge acquisition facilities to support the evolution of the system during its life-cycle.

This paper describes the innovations developed in WIKITAAABLE, whose architecture is described in section 2 and discusses current research issues. One innovation this year is that the system is embedded within a semantic wiki and that the collaborative aspects are also of main concern mainly for document and knowledge edition and update.

The remainder of the paper shows how knowledge is manipulated within the system. Section 3 presents knowledge acquisition and representation within the semantic wiki. Section 4 illustrates how knowledge is used by the inference engine. Section 5 describes an opportunistic acquisition strategy guiding the evolution of the system knowledge. Finally, section 6 draws some conclusions and points out ongoing and future work. For qualification purpose, a restricted interface of the system is available online. This interface only allows users to ask queries to the system. The full application with embedded knowledge acquisition features will be available for the contest.

## 2 Architecture of WIKITAAABLE

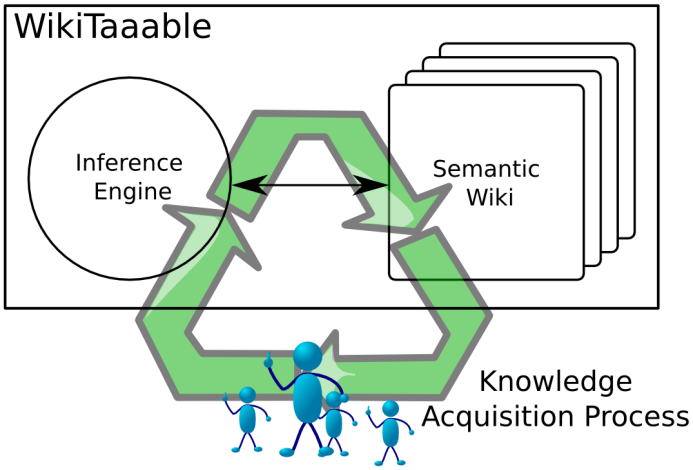


**Fig. 1.** Overview of the WIKITAAABLE architecture.

In a CBR system, results strongly depend on the quality of available knowledge. As a consequence, continuous improvement of knowledge is required to progressively enhance the obtained results.

The previous version of TAAABLE [1] suffered from different problems making maintenance and evolution of the system knowledge difficult. For example, there was no way to capture user feedback and to reuse it for maintenance. Besides, knowledge was organized in several files of heterogeneous formats that were difficult to update. As a consequence, the evolutions of the system knowledge were tedious tasks, even for the designers of TAAABLE.

In WIKITAAABLE [3] we decided to use a semantic wiki (Semantic Media Wiki [5]) as a central module to manage all data and knowledge used in the system. Making use of a semantic wiki has two major advantages: it enables humans and machines to rely on the same tool for representing and reasoning on shared knowledge and it provides users with user-friendly interfaces for browsing and editing knowledge.

Figure 1 gives an overview of the WIKITAAABLE architecture. Each component has been designed to work with the others and the components are strongly intertwined. For example, knowledge has not been represented at a general level but for reasoning purpose. The *semantic wiki* module manages knowledge of the system. The wiki is accessible for the users through a graphical interface and for the system through bots[1] that automates several tasks. Section 3 details this module. The *inference engine* includes the CBR core and is able to reason on the knowledge available in the wiki. It is described in section 4. In order to facilitate knowledge acquisition, the architecture of WIKITAAABLE is designed in such a way that it enables as much interactions as possible. *Opportunistic knowledge acquisition process* developed in WIKITAAABLE is discussed in section 5.

## 3 A Semantic Wiki for Collaborative Acquisition of Cooking Knowledge

In [3], Semantic Media Wiki (SMW [5]) is used as a blackboard for WIKITAAABLE knowledge management. WIKITAAABLE gathers the whole knowledge body required by the application.

To import knowledge of the first version of the TAAABLE system [1] into WIKITAAABLE, we wrote several bots that use mediawiki API. Recipes, ontologies, and specific adaptation knowledge are now represented as a graph of semantic wiki pages. Each page can be freely read an updated by humans and by bots. Hence, TAAABLE is now maintained and improved by a collaboration between users and machines.

### 3.1 Domain Ontology

The domain ontology contains four hierarchies: *ingredients, dish moments, dish types, and dish origins*. For adapting a recipe by substituting some ingredients by other ingredients, the CBR engine requires knowledge stating similarity between ingredients. Therefore, ingredients are organized in the *ingredient hierarchy*. This hierarchy is used by the CBR engine to compute the cost of a substitution: the closer the ingredients, the lower the cost; e.g., *orange* is closer to *lemon* than *apple*.[2] In order to characterize recipes, three other hierarchies define and organize dish moments (*appetizer*, *dessert*), dish types (*cake*, *pizza*), and dish origins (*Mediterranean*, *Chinese*). The original acquisition of the hierarchies is described in [1].

The four hierarchies are imported into WIKITAAABLE by using the *Category/SubCategory* relation of Semantic MediaWiki [5]. For example, there is a page for orange and another page for citrus and the two pages are linked by this relation. For instance, the figure 2 shows the imported ingredient hierarchy.

[1] A bot is a piece of software for doing automated repetitive tasks.

[2] This closeness can be measured by a weighted length of the shortest path between ingredients in the hierarchy.

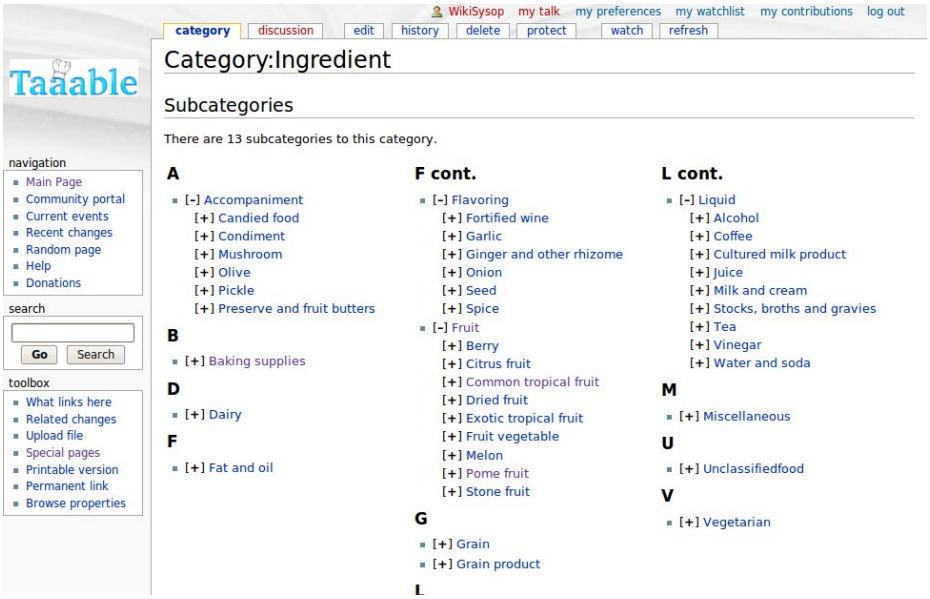


**Fig. 2.** WIKITAAABLE ingredient ontology.

### 3.2 Adaptation knowledge

To adapt a recipe, the CBR engine uses the ontology and a set `AK` of substitutions. A substitution $\sigma \in$ `AK` states that, in a given context, some ingredients may be substituted by other ones. For instance, the following substitution states that, for the context of a salad without potato, vinegar may be substituted by lemon juice and salt.

$$\sigma = \begin{array}{|l|l|l|} \hline \textbf{context} \text{ salad} \\ \quad\quad\quad\quad \textbf{no} \text{ potato} & \textbf{from} \text{ vinegar} & \textbf{by} \text{ lemon juice} \\ & & \quad\quad \text{salt} \\ \hline \end{array} \quad (1)$$

Each substitution is represented in a semantic wiki page. For instance, figure 3 shows the wiki page of the above substitution. The acquisition of substitutions is detailed in section 5.

### 3.3 Recipes

The recipes are also imported into WIKITAAABLE, where a wiki page is created for each recipe. Then, a bot crawls all the recipe pages, parses ingredient information, sets dish types, moments, and origins. It updates recipe pages with this information encoded as semantic annotations. Figure 4 shows a recipe page in WIKITAAABLE. We used the $n$-ary relationship of Semantic Media Wiki to represent an ingredient line, for example, $(1, \mathrm{c}, \texttt{Category:rice})$ represents `1 c rice`, the first ingredient line in figure 4. The parsing of ingredient information and setting types is described in [1].


**Fig. 3.** A substitution semantic wiki page.

## 4 Principles of the CBR Inference

### 4.1 Knowledge containers

Knowledge in WIKITAAABLE is mainly expressed in propositional logic. The TAAABLE knowledge base is a set of containers $\texttt{KB} = \{\mathcal{O}, \texttt{Recipes}, \mathcal{H}_{idx}, \texttt{AK}, \texttt{cost}\}$. KB is encoded in wiki pages and the CBR engine has access to these pages through SPARQL queries.

$\mathcal{O}$ is the domain ontology represented by a set of axioms of the form $a \Rightarrow b$ where $a$ and $b$ are two variables representing recipe classes. For example, `lemon` (resp., `citrus`) represents the class of recipes with lemon (resp., with citrus) and the axiom $\texttt{lemon} \Rightarrow \texttt{citrus}$ states that any recipe with lemon is a recipe with citrus. In fact, every ingredient name $X$ such as `lemon` is interpreted here as "class of the recipes with ingredient $X$".

`Recipes` is the set of recipes given by the CCC organizers, and consequently the case base of the CBR system TAAABLE. A recipe $R \in \texttt{Recipes}$ cannot be directly handled by the CBR inference engine: the engine requires a formal representation whereas $R$ is for the largest part written in natural language. Therefore, only the formalized part of the recipe $R$ is used: its index $idx(R)$, which is expressed by a conjunction of literals (the indexing process of the recipes coincides with the annotation process mentioned in section 3.3). For example

$$idx(R) = \texttt{lettuce} \wedge \texttt{vinegar} \wedge \texttt{olive\_oil} \wedge \texttt{tomato} \wedge \textit{Nothing else} \qquad (2)$$

is a formal representation of a recipe $R$ having ingredients lettuce, vinegar, olive oil, and tomato. A closed world assumption is associated to $idx(R)$ stating that if a prop-

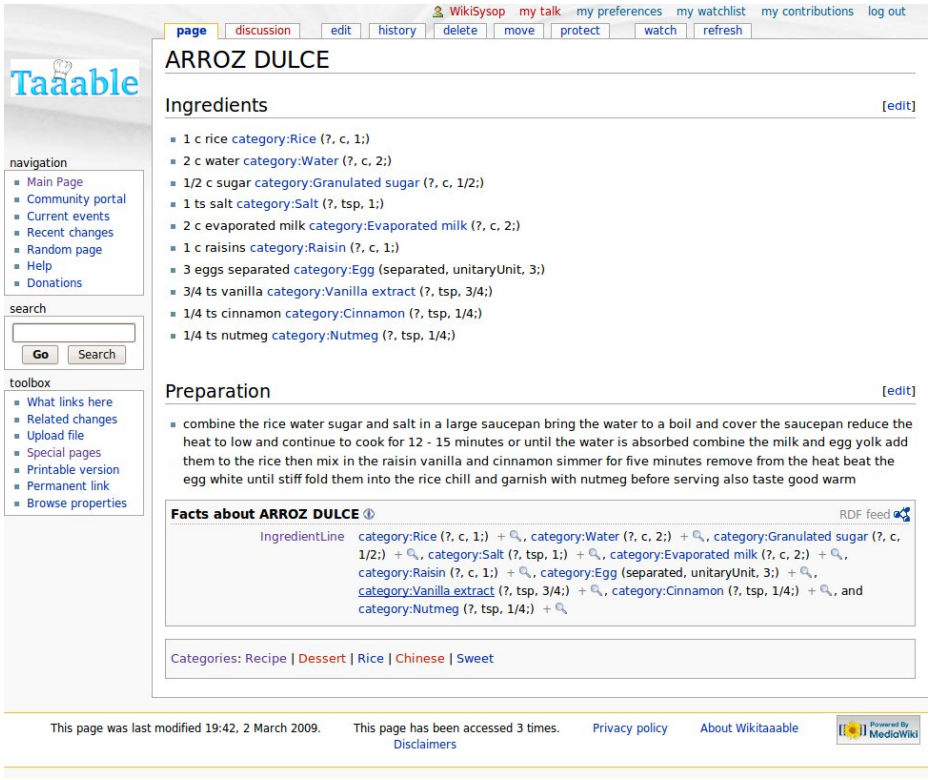


**Fig. 4.** indexed recipe of "Arroz dulce".

erty cannot be deduced from the recipe description and from the ontology then it is considered as false. Formally, if $idx(R) \not\models_{\mathcal{O}} a$ then "*Nothing else*" contains the conjunct $\neg a$.[3] For example, this closed world assumption enables to deduce that $idx(R) \models_{\mathcal{O}} \neg\texttt{meat} \wedge \neg\texttt{fish}$, i.e., that $R$ is a vegetarian recipe.

The indexes $idx(R)$ are used to access recipes through a hierarchy $\mathcal{H}_{idx}$, according to the partial ordering $\models_{\mathcal{O}}$: for $C, D \in \mathcal{H}_{idx}$, $C \models_{\mathcal{O}} D$ iff there is a directed path in $\mathcal{H}_{idx}$ from $C$ to $D$. The indexes $idx(R)$ are leaves of the $\mathcal{H}_{idx}$.

Adaptation knowledge has two parts. The first part is included in ontology $\mathcal{O}$. The second part is the set $\texttt{AK}$ of substitutions (cf. section 3.2). Any $\sigma \in \texttt{AK}$ may be considered as a domain specific inference rule $\dfrac{R \text{ is a good recipe}}{\sigma(R) \text{ is a good recipe}}$. The substitutions have the form $C \rightsquigarrow D$ where $C$ and $D$ are conjunctions of literals. Applying $C \rightsquigarrow D$ to a conjunction of literals (such as an index or a query) consists in replacing the literals of $C$ by literals of $D$. For example, the substitution $\sigma$ described in figure 3 is written as

[3] If $f$ and $g$ are two propositional formulas, $f \models_{\mathcal{O}} g$ means that $f$ implies $g$, given the ontology $\mathcal{O}$. More precisely: if $\mathcal{I}$ satisfies both $\mathcal{O}$ and $f$ then $\mathcal{I}$ satisfies $g$.

follows

$$\sigma = \texttt{salad} \wedge \neg\texttt{potato} \wedge \texttt{vinegar} \rightsquigarrow \texttt{salad} \wedge \neg\texttt{potato} \wedge \texttt{lemon\_juice} \wedge \texttt{salt} \quad (3)$$

It can be noticed that the substitution given by a triple $(\texttt{context}, \texttt{from}, \texttt{by})$ in the wiki pages (cf. equation (1)) are rewritten $\texttt{context} \wedge \texttt{from} \rightsquigarrow \texttt{context} \wedge \texttt{by}$ to suit the CBR engine formalism.

The CBR inference is based on substitutions, either taken from AK or built with the help of ontology $\mathcal{O}$ (see below for details). The choice of substitutions is made according to the problem-solving context and to a cost function $\texttt{cost} : \sigma \mapsto \texttt{cost}(\sigma) > 0$; substitution $\sigma$ is preferred to substitution $\tau$ when $\texttt{cost}(\sigma) < \texttt{cost}(\tau)$. Therefore, the cost function (and its parameters) constitutes an additional knowledge container.

### 4.2 CBR Inference

Let $Q$ be a query. For example

$$Q = \texttt{endive} \wedge \texttt{lemon\_juice} \wedge \neg\texttt{onion} \quad (4)$$

is a query for a recipe with endive and lemon juice and without onion. The CBR inference consists in (1) retrieving recipes matching exactly or approximately $Q$ and (2) adapting the retrieved recipes.

Retrieval aims at choosing indexes $\mathit{idx}(R)$ matching the query $Q$. An exact match corresponds to $\mathit{idx}(R) \vDash_{\mathcal{O}} Q$. If no index exactly matches $Q$, the query $Q$ is progressively relaxed into $\Gamma(Q)$ such that $\mathit{idx}(R) \vDash_{\mathcal{O}} \Gamma(Q)$, for some $\mathit{idx}(R)$. The relaxation of $Q$ is obtained by applying generalizations $g_k$ according to $\mathcal{O}$: $g_k = a_k \rightsquigarrow b_k$ is a substitution such that $(a_k \Rightarrow b_k) \in \mathcal{O}$. Thus $\Gamma(Q) = g_n(\ldots(g_1(Q))\ldots)$. Therefore, retrieval provides a similarity path

$$\mathit{idx}(R) \vDash_{\mathcal{O}} \Gamma(Q) \xleftarrow{g_n} \ldots \xleftarrow{g_1} Q \quad (5)$$

This similarity path is built according to a best-first search minimizing $\sum_k \texttt{cost}(g_k)$. For example, retrieval can give the following result

$$\begin{aligned} Q &= \texttt{endive} \wedge \texttt{lemon\_juice} \wedge \neg\texttt{onion} \\ \Gamma(Q) &= \texttt{green\_salad} \wedge \top \wedge \neg\texttt{onion} \equiv \texttt{green\_salad} \wedge \neg\texttt{onion} \\ \mathit{idx}(R) &= \texttt{lettuce} \wedge \texttt{vinegar} \wedge \texttt{olive\_oil} \wedge \texttt{tomato} \wedge \textit{Nothing else} \end{aligned}$$

($\Gamma$ consists in generalizing endive into green_salad and in removing lemon_juice from the query by generalizing it in several steps to $\top$, the hierarchy top).

Adaptation is composed of two sub-steps. Let $R$ be a retrieved recipe, with index $\mathit{idx}(R)$. The first subset of adaptation is matching, that aims at building an adaptation path from $\mathit{idx}(R)$ to $Q$ of the form

$$\mathit{idx}(R) \xmapsto{\sigma_1} \ldots \xmapsto{\sigma_p} \Sigma(\mathit{idx}(R)) \vDash_{\mathcal{O}} \Gamma(Q) \xleftarrow{\gamma_q} \ldots \xleftarrow{\gamma_1} Q \quad (6)$$

where $\sigma_i \in \texttt{AK}$ $(i = 1 \ldots p)$ and substitutions $\gamma_j$ $(j = 1 \ldots q)$ correspond to axioms of the ontology: $\gamma_j = a_j \rightsquigarrow b_j$ with $(a_j \Rightarrow b_j) \in \mathcal{O}$. Such an adaptation path is built according to a best-first search in a state space minimizing $\sum_i \texttt{cost}(\sigma_i) + \sum_j \texttt{cost}(\gamma_j)$.

The second sub-step of adaptation consists in "following" the adaptation path: first $R$ is adapted successively in $\sigma_1(R)$, $\sigma_2(\sigma_1(R))$, $\ldots \sigma_p(\ldots(\sigma_2(\sigma_1(R))\ldots) = \Sigma(R)$. Then, ingredients of $\Sigma(R)$ are substituted by other ingredients according to a generalization-specialization process (generalization corresponds to the relation $\models_{\mathcal{O}}$ and specialization to the substitutions $\gamma_q^{-1}, \ldots, \gamma_1^{-1}$).

For example, let $idx(R)$ and $Q$ be the example presented above. Matching may provide the following adaptation path:

$$idx(R) \overset{\sigma}{\longmapsto} \Sigma(idx(R)) \models_{\mathcal{O}} \Gamma(Q) \overset{\gamma}{\longleftarrow} Q$$

where $\sigma$ is defined by (3) and $\gamma = \texttt{endive} \rightsquigarrow \texttt{green\_salad}$. Thus, the adaptation of $R$ consists in replacing vinegar by lemon juice and salt (cf. $\sigma$) and by substituting `lettuce` by `endive` (cf. $\models_{\mathcal{O}}$ and $\gamma^{-1}$).

It can be noticed that retrieval provides a first matching: a similarity path is a kind of adaptation path involving no $\sigma \in \texttt{AK}$. Thus, the retrieved recipe $R$ can be adapted following this similarity/adaptation path. However, during adaptation, some substitutions $\sigma \in \texttt{AK}$ may be used and, if they do, the resulting adaptation requires less effort.[4]

## 5 Opportunistic Knowledge Acquisition and Discovery

WIKITAAABLE has been designed to facilitate continuous knowledge acquisition through interactions with its users: it is a reflexive and perpetually evolving system. However, due to the heterogeneity of knowledge acquisition situations that can be envisioned, setting up such a process is a complex task. This diversity of situations is explained by several factors:

- The various types of knowledge (ontology, adaptation knowledge, substitutions costs, recipes) that can be acquired.
- The different interaction modalities such as simple user feedback, direct modification on wiki pages, interaction through dedicated interfaces, use of external knowledge discovery methods, etc.
- The provenance of knowledge that is acquired: single users, community of users, experts, other sources of data (web sites), or local knowledge from which new knowledge in inferred.

In the following, a particular scenario of opportunistic knowledge discovery is detailed. In WIKITAAABLE, substitutions $\sigma \in \texttt{AK}$ are acquired at problem-solving time through interactions with the user. The knowledge discovery process CABAMAKA [4] is used to assist the user in the formulation of new pieces of knowledge. Its role is to generate a set of substitutions $\sigma \in \texttt{AK}$ "on the fly" from the comparison of two sets of recipes of the case base. The generated substitutions can be used by the system to repair a failed adaptation. Each time a substitution is validated by the user, it is stored

[4] If the cost function is an estimation of the adaptation effort, then the adapted recipe should be better by following (6) then by following (5). Indeed, since adding new substitutions (the ones of AK) only adds new ways to connect indexes to queries, it comes that $\sum_i \texttt{cost}(\sigma_i) + \sum_j \texttt{cost}(\gamma_j) \leq \sum_k \texttt{cost}(g_k)$.

for future reuse. In the following, the main principles of the approach are illustrated on an example. More details on the proposed approach can be found in [2].

In section 4, the user wanted a salad recipe with lemon juice but without onion, which was modeled by the query $Q$ defined by (4). The substitution $\sigma \in \texttt{AK}$ defined by (3) was used to adapt the retrieved recipe $R$ by replacing vinegar by lemon juice and salt. Such a substitution cannot be obtained from the ontology $\mathcal{O}$, so let us assume in this scenario that $\sigma$ is not available to the system. Thus, to perform adaptation, the system relies solely on the ontology $\mathcal{O}$ from which it generates the substitution $\texttt{vinegar} \rightsquigarrow \texttt{lemon\_juice}$. Now, in our scenario, the user is not satisfied with the proposed solution and gives this feedback to the system. Therefore, the knowledge discovery process is triggered: a set of substitutions is learned from the case base by comparing salad recipes with vinegar and salad recipes with lemon juice. Among the learned substitutions is the substitution $\sigma_{learned} = \texttt{vinegar} \rightsquigarrow \texttt{lemon\_juice} \wedge \texttt{salt}$, which suggests to replace vinegar by lemon juice in the retrieved recipe $R$ and to add salt. The user is satisfied with the adaptation resulting from the application of this substitution, so the latter is stored for future reuse. At this point, the user is encouraged to specify the condition of application of the substitution $\sigma_{learned}$. The user states that vinegar can be replaced by lemon juice and salt in salad recipes that do not contain potato, which is modeled by the substitution context $\texttt{salad} \wedge \neg \texttt{potato}$. Combining the learned substitution $\sigma_{learned}$ and its application context gives the substitution $\sigma$ defined by (3).

In WIKITAAABLE, the wiki is used as a gateway enabling to centralize knowledge used in the system. It provides functionalities to facilitate acquisition and maintenance of knowledge and enables to progressively add new acquisition features, allowing the evolution of the whole system. However, setting up a complex knowledge acquisition process raises several issues. For example, tools for ensuring consistency of knowledge used in the system must be developed. Another issue is to handle updates from multiple users. What happens when one believes that an avocado is to be eaten as a starter whereas someone else reckon that it has to be eaten as a dessert? Is the system supposed to converge towards a "commonly accepted" knowledge base or should it be able to deal with user's preferences?

A strength of the architecture of WIKITAAABLE is that it will enable to progressively address these issues. A future work is to allow users to add their own recipes to the system. This functionality requires to be able to dynamically annotate new recipes within WIKITAAABLE in order to make them usable by the CBR inference engine. One of the advantages of such a functionality, combined with the benefits of a wiki, is that communities of users will be able to share their recipes and to collaborate in order to improve the global knowledge of the system. Next, we would like to tackle the multi-user issue which is a prerequisite for envisioning a collaborative building of the knowledge base of TAAABLE.

## 6 Conclusion, Ongoing Work, and Future Work

The textual case-based cooking system WIKITAAABLE participates to the second CCC. It is an extension of the TAAABLE system that has participated to the first CCC. WIKITAAABLE's architecture is composed of a semantic wiki used for the collaborative

acquisition of knowledge (recipe, ontology, adaptation knowledge) and of a CBR inference engine using this knowledge for retrieving and adapting recipes. This architecture allows various modes of knowledge acquisition for CBR that are studied within the TAAABLE project. In particular, opportunistic adaptation knowledge discovery is an approach for interactive and semi-automatic learning of adaptation knowledge triggered by a feedback from the users.

The first ongoing work is the improvement of the WIKITAAABLE system (user interface, inference engine, knowledge base encoded in wiki pages, and links between these components). Another work planned for the next weeks is the development of tools within WIKITAAABLE for knowledge acquisition triggered by user feedbacks. Such a knowledge acquisition leads to a continuous evolution of the knowledge base and thus, of the behavior of the system. It is important to ensure that these evolutions are improvements and that the integrity of the knowledge is preserved. We plan to use non regression and consistency tests for this purpose. Today, only the compulsory task of the CCC is addressed by WIKITAAABLE but we plan to have also the two challenges addressed by the system for the day of the contest.

Currently, wiki pages are accessed and maintained by a limited community: the TAAABLE project members. These pages encode the knowledge that have been acquired on the basis of a consensus. A long term objective is to have several open semantic wikis with cooking knowledge, each of them corresponding to a user community, the consensus being only realized at the level of a community.

## Acknowledgments

The participants of the TAAABLE project wish to thank the organizers of the CCC for providing this benchmark, that entails many interesting problems, and the need to collaborate with researchers in various domains on knowledge engineering.

## References

1. F. Badra, R. Bendaoud, R. Bentebitel, P.-A. Champin, J. Cojan, A. Cordier, S. Desprès, 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, 2008.
2. F. Badra, A. Cordier, and J. Lieber. Opportunistic adaptation knowledge discovery. In Lorraine McGinty and David C. Wilson, editors, *Case-Based Reasoning Research and Development / ICCBR 2009*, 2009. To appear.
3. 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.
4. M. d'Aquin, F. Badra, S. Lafrogne, J. Lieber, A. Napoli, and L. Szathmary. Case base mining for adaptation knowledge acquisition. In *Proceedings of the International Conference on Artificial Intelligence, IJCAI'07*, pages 750–756, 2007.
5. M. Völkel, M. Krötzsch, D. Vrandecic, H. Haller, and R. Studer. Semantic Wikipedia. *Journal of Web Semantics*, 5(4), 2007.