Comparative Analysis of Case Retrieval Implementation for Knowledge Intensive CBR Application

Nadjette Dendani-Hadiby and Med Tarek Khadir
LabGED Laboratory, Computer Science Department, Badji Mokhtar Annaba University, Annaba, Algeria
Email: {hadibi, khadir} @labged.net

Souad Guessoum and Hayet Djillali
LRS Laboratory, Computer Science Department, Badji Mokhtar Annaba University, Annaba, Algeria
Email: {souadguessoum, hayetdjellali} @yahoo.fr

***Abstract*—The Case-Based Reasoning (CBR) is a problem solving approach based on reuse by analogy from past experiences. A CBR system is a combination of processes and knowledge called "knowledge containers", its reasoning power can be improved through the use of domain knowledge. CBR systems combining case specific knowledge with general domain knowledge models are called Knowledge Intensive CBR (KI-CBR). Although CBR claims to reduce the effort required for developing knowledge-based systems substantially compared with more traditional Artificial Intelligence approaches, the implementation of a CBR application from scratch is still a time consuming task. The present work aims to develop a CBR application for fault diagnosis of steam turbines that integrates a domain knowledge modeling in an ontological form and focuses on the similarity-based retrieval step. This system is view as a KI-CBR system based on domain ontology, built around jCOLIBRI and myCBR a well-known frameworks to design KI-CBR systems. During prototyping, we examine the use and functionality of the focused frameworks. A comparative study is performed with results of presenting a successful retrieval task and demonstrating that jCOLIBRI and myCBR are well adapted to design KI-CBR system.**

***Index Terms*—Cases based reasoning, domain ontology, fault diagnosis, KI-CBR, jCOLIBRI, myCBR**

I. Introduction

The diagnosis is an intelligent act which is hardly programmable with classic techniques. Several studies have been conducted for the development of fault diagnosis methods based on Artificial Intelligence (AI) methods and techniques. One of these techniques is Case-Based Reasoning (CBR) [1].

This choice is motivated by the idea that an industrial expert intervenes to diagnose a fault, he tries to remember past experiences of fault observed in similar situations which can lead to similar results [2], thus CBR techniques allow to simulate the expert reasoning. It also offers the reuse of past experience facilitating knowledge acquisition and process sharing, creating the opportunity of learning from experiences. A CBR relies on several containers of knowledge. The source cases are, obviously, among those containers of knowledge, but a lot of systems also use additional knowledge sources as the "domain knowledge". The more correct and accurate the domain knowledge is, the better the CBR system's inferences will be. This additional knowledge is based on knowledge modeling which is given by a representation model in the form of domain ontology.

In CBR, the design integrated systems that combine case specific knowledge with general domain knowledge models are called Knowledge Intensive CBR (KI-CBR).

The development of a quite simple Case-Based Reasoning application already involves a number of steps, Compared with other AI approaches, CBR allows to reduce the effort required for knowledge acquisition and representation significantly, which is certainly one of the major reasons for the commercial success of CBR applications. Nevertheless, implementing a CBR application from scratch remains a time consuming software engineering process and requires a lot of specific experience beyond pure programming skills.

However, more easily available and less complex CBR tools are required. Two frameworks are examined in building the diagnosis task. One is the open source jCOLIBRI [3] system developed by GAIA group and provides a framework for building CBR systems based on state-of-the art software engineering techniques. The other is the novel open source CBR tool myCBR [4] developed at the German Research Center for Artificial Intelligence (DFKI).

The present work aims to develop a CBR application for fault diagnosis of steam turbines that integrates a domain knowledge modeling in an ontological form and focuses on the similarity-based retrieval step. During prototyping, we examine the use and functionality of the focused frameworks and a comparative study is performed with results of presenting a successful retrieval task and demonstrating that jCOLIBRI and myCBR are well adapted to design KI-CBR system.

Manuscript received October 21, 2015; revised January 12, 2016.

© 2016 J. Adv. Inf. Technol.
doi: 10.12720/jait.7.2.105-112

The paper is organized as follows: Section 2 gives a theoretical background about CBR, ontology, application domain. The description of the proposed architecture is given in section 3. Section 4 presents the used paradigms and the system architecture implementation. We discuss our work in Section 5 and conclude it in the last one.

II. THEORETICAL BACKGROUND

The proposed approach combines AI paradigms previously cited, for instance Ontologies and CBR. Both paradigms contribution in the context OF knowledge capitalization and diagnosis for industrial purposes are explained in what follows

A. Ontology

Knowledge capitalization process consist in marking the crucial knowledge (know and know-how) that are necessary to the processes of decision. So it's important to identify; then to formalize and model the explicit knowledge in order to memorize them. One of the proposed methods is the construction of the ontology [5]. The following definition has been given to the ontology in [5] "to make ontology, is to decide of the individuals who exist, the concepts and properties that characterize them and the relations that link them".

Gruber [6] defines the ontology as: "An ontology is an explicit specification of a conceptualization." Since then, a number of definitions for an ontological construction has been given. In 1997 Borst [7] added the terms shared and formal to Gruber's definition giving: "An ontology is a formal specification of a shared conceptualization." One year later both definitions were merged into one [8], giving: "An ontology is a formal, explicit specification of a shared conceptualization." The type of an ontology is closely related to its conceptualization objects such as: knowledge representation, high level, generic, domain and application. In our case the developed ontology is of domain type, as it contains a number of concepts and a certain vocabulary that defines a targeted domain i.e., the steam turbine and its maintenance aspects

B. Case Based Resoning (CBR)

The processes that make up case-based reasoning can be seen as a reflection of a particular type of human reasoning. In many situations, the problems that human beings encounter are solved with a human equivalent of CBR. When a person encounters a new situation or problem, he or she will often refer to a past experience of a similar problem. This previous experience may be one that they have had or one that another person has experienced. If the experience originates from another person, the case will have been added to the (human) memory through either an oral or a written account of that experience. The idea of CBR is intuitively appealing because it is similar to human problem solving behavior. Therefore, CBR involves reasoning from prior examples [1][9]: retaining a memory of previous problems and their solutions and solving new problems by reference to that knowledge. Descended of the research in artificial intelligence on the problems resolution, this principle of resolution can be described as follows [10]: Generally, a case-based reasoner will be presented with a problem, either by a user or by a program or system. The case-based reasoner then searches its memory of past cases (called the case base) and attempts to find a case that has the same problem specification as the case under analysis. If the reasoner cannot find an identical case in its case base, it will attempt to find a case or multiple cases that most closely match the current case. In situations where a previous identical case is retrieved, assuming that its solution was successful, it can be offered as a solution to the current problem. In the more likely situation that the case retrieved is not identical to the current case, an adaptation phase occurs. During adaptation, differences between the current and retrieved cases are first identified and then the solution associated with the case retrieved is modified, taking these differences into account. The solution returned in response to the current problem specification may then be tried in the appropriate domain setting.

At the highest level of abstraction, a case-based reasoning system can be viewed as a black box represented often by a cycle [10], [11] (Fig. 1) that incorporates the reasoning mechanism and the following external facets:

- The input specification or problem case.
- The output that defines a suggested solution to the problem.
- The memory of past cases, the case base, that are referenced by the reasoning mechanism.

In many practical applications, the reuse and revise stages (Fig. 1) are sometimes difficult to distinguish, and several researchers use a single adaptation stage that replaces and combines them. However, adaptation in CBR systems is still an open question because it is a complicated process that tries to manipulate case solutions [11]. Usually, this requires the development of a causal model between the problem space (i.e., the problem specification) and the solution space (i.e., the solution features) of the related case.

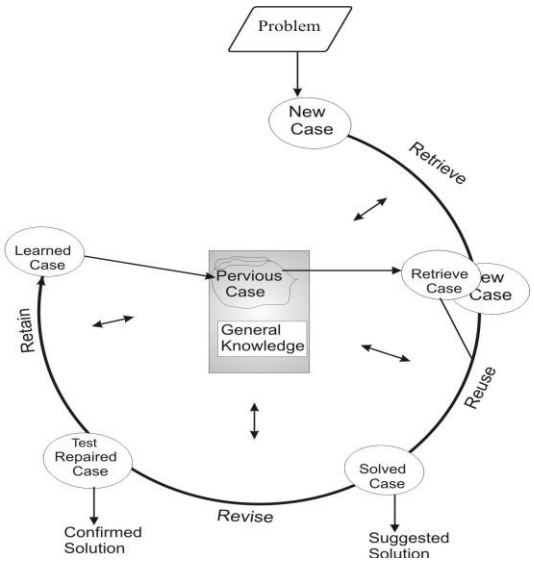

Figure 1. Case based reasoning phases [1]

The feasibility of the CBR for the decision support where the experience of past situations is reused to manage new situations, has been shown in the survey of the decision making process [11]. The deepening of this mechanism (CBR) brings us to see behind a knowledge management process. In fact, the CBR and the knowledge management follow the same objective of acquisition and reuse of experience or knowledge.

C. Steam Turbines

Steam turbines are mechanical devices using superheated steam power, and convert it into useful mechanical work. In the studied case, the mechanical work produced is used for electrical production. The steam is created by a boiler, where pure water passes through a series of tubes and then boils under high pressure to become superheated steam. The heat in the firebox is normally provided by burning fossil fuel (*e.g.* coal, fuel oil, or natural gas as in the studied case). The superheated steam leaving the boiler then enters the steam turbine throttle, where it powers the turbine and connected generator to make electricity. After the steam expands through the turbine, it exits the back end where it is cooled and condensed back to water in the surface condenser. This condensate is then returned to the boiler through high-pressure feed pumps for reuse. Heat from the condensing steam is normally rejected from the condenser to a body of water; in the studied case sea water is used. Because of the importance of the steam turbines in the process of the economic development, maintenance operation of these equipments is of a fundamental importance. It permits to reduce the inactivity time of equipments that is *very expensive.*

III. System Architecture

The proposed approach includes the use of ontologies to build models of general domain knowledge. Although in a CBR system the main source of knowledge is the set of previous experiences, our approach to CBR is towards integrated applications that combine case specific knowledge with models of general domain knowledge. The more knowledge is embedded into the system, the more effective it is expected to be. Semantic CBR processes can take advantage of this domain knowledge and obtain more accurate results. In KI-CBR systems, ontologies play an important role [12], as a vocabulary to describe cases, as knowledge structure where cases are located, and as knowledge source allowing the semantic reasoning in the methods of similarity calculation, adaptation and learning.

Based on the latter, a novel ontological form of implementing reasoning process. The proposed architecture [13] is composed of two functional components, domain ontology and CBR application.

A. Domain Knowledge

The domain ontology is implemented to build general knowledge models and which includes vocabularies, concepts and relations for representing all knowledge concerning fault diagnosis of steam turbines equipment and its maintenance aspects.

B. CBR Application

This component is used for solving a diagnosis problem and dedicated to the case structure and to the four task of CBR process (Retrieve the most similar case/s, Reuse its/their knowledge to solve the problem, Revise the proposed solution and Retain the experience). The aforementioned tasks are mapped respectively to the following processes: a. Case representation, b. Case matching and retrieval, c. Case adaptation, and, d. Case-base maintenance. The focus is put only on the first two tasks.

IV. Implementation

Protege editor [14] is chosen for the manual generation and modeling of the domain ontology and jCOLIBRI, myCBR for building CBR application

A. Domain Ontology

Protege is a Java-based open source software tool that can be used to develop knowledge based systems and to create ontologies [15].

We use a domain ontology "Onto-turb" developed in our research laboratory by Djeddi and khadir [16] built with Protege, used to store the diagnosis cases of the steam turbines equipments. This ontology was constructed from database of a central system at SONALGAZ Company and domain experts were asked to validate it. (Fig. 2) illustrates the obtained ontology that can contain description of classes, properties and instances.

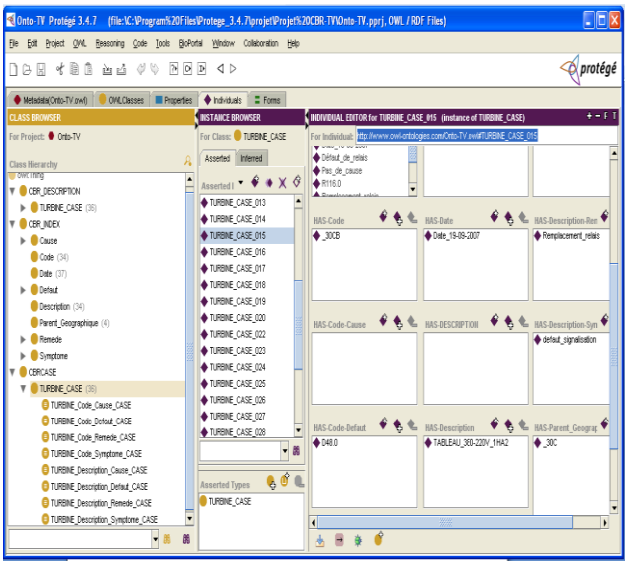

Figure 2. The classes hierarchy in "Onto-Turb"

B. CBR application CBR Application

Developing a CBR system is a complex task where many decisions have to be taken, the system designer has to decide among a range of different methods for organizing, retrieving and reusing the knowledge retained in past cases. This process would greatly benefit from the reuse of previously developed CBR systems.

© 2016 J. Adv. Inf. Technol.

Two object-oriented ontology based CBR frameworks jCOLIBRI and myCBR are used for developing a CBR application.

1) Using jCOLIBRI

jCOLIBRI is an object-oriented framework in Java for building CBR systems, developed by GAIA group, is used to develop our CBR application. Expert programmers can use java to instantiate the framework, although the easiest way of using its graphical configuration tools. The underlying ideas of CBR can be applied consistently across application domains. However, developing a CBR system is a complex task where many decisions have to be taken.

The system designer has to decide among a range of different methods for organizing, retrieving and reusing the knowledge retained in past cases. This process would greatly benefit from the reuse of previously developed CBR systems.

In jCOLIBRI, ontology is not represented as a new source; all concepts of CBR are mapped into classes and interfaces of framework. Classes that represent the ontology concept serve as templates where new CBR types should be added. They also provide the tasks and abstract interface of the methods. The design of the jCOLIBRI framework comprises a hierarchy of Java classes plus a number of XML files and is organized around the following elements [3]:

- Tasks and methods: The tasks supported by the framework and the methods that solve them are all stored in a set of XML files.
- Case-base: Different connectors are defined to support several types of case determination, from the file system to a database.
- Cases: A number of interfaces and classes are included in the framework to provide an abstract representation of cases that support any type of actual case structure.
- Problem solving methods: The actual code that supports the methods included in the framework.

jCOLIBRI includes a complete Graphical User Interface (GUI) that guides the user in the design steps of a CBR system, these steps are given in that follows:

2) Definition of case structures

A case is composed by three components: description (describes the problem), solution (represents a possible solution approach) and result (reveals if the proposed solution is able to solve the problem). Description and solution are collections of simple or compound attributes, permitting us to build a hierarchical case structure. By using jCOLIBRI GUI users are able to create the case structure defining simple and compound attributes that describe the cases together with their types, weights, similarity measure that is chosen from a library of existing similarity functions and parameters or others can easily be included. This generates a XML file with the structure information. When user has defined the case structure he configures a connector that uses that information for mapping the cases to the chosen persistence media. This mapping is also saved to a XML file. (Fig. 3) shows the definition of the diagnosis case parameters for instance description, symptom, cause, defect, geographical parent, date, remedy.

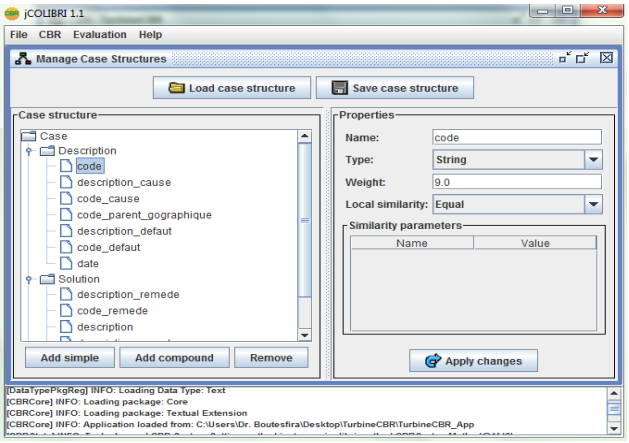

Figure 3. Creation of case structure

V. Creation of Case Structure

Building the case-base: The cases persistence concept is then built around those connectors representing objects that know how to access and retrieve cases from the storage media and return those cases to the CBR system in a uniform way. Therefore connectors provide an abstraction mechanism that allows users to load cases from different storage sources in a transparent way [17]. Defined connectors can work with plain text files, XML files, relational data bases or ontology. However, using ontologies as persistence media means that the slots of the case structure are defined by the concepts and properties of the ontology and the slots-filler are the instances of these concepts. This way, it makes no sense to do a representation of the case structure and then map it (using the connector) with the same case structure contained in the ontology. To solve this problem our pure ontological case structure represents directly the concepts and properties of the ontology using the Java classes exploited for reasoning. With this approach the connector for DLs does not need any configuration file and can load the cases from the ontology using only case structure information.(Fig. 4) shows how the case structure is mapped with ontology.

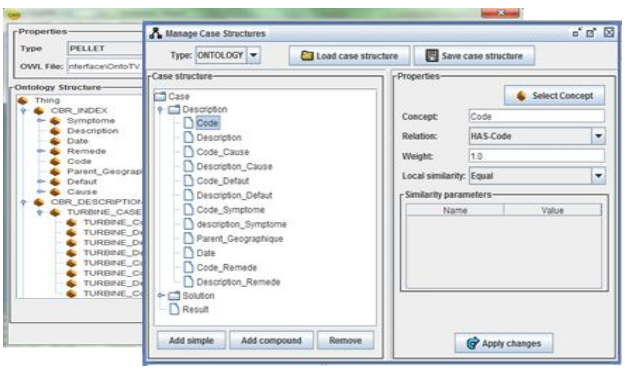

Figure 4. Case attributes mapping with ontology

© 2016 J. Adv. Inf. Technol.

VI. Case Attributes Mapping with Ontology

A. Managing Similarity Measures

Computational based retrieval approach is used where numerical similarity functions are used to assess and order cases regarding the query. When two cases are compared, the local similarity functions are used to compare simple attribute values. Global similarity functions are linked to compound attributes and are used to gather the similarities of the collected attributes in a unique similarity value. At last, the similarity value of two cases is computed as the similarity of their description concepts. The available similarity measures are listed in a configuration file, and can be managed using the GUI. These functions compute the similarity between the query and the case, and are used to choose the most similar case to the query. There are two types of similarity functions: Local functions that compute the similarity between simple attributes and global ones focusing on some sort of average over the local similarities. The similarity between query (q) and cases (c), Sim (q, c) is defined as follows:

$$Sim(q,c) = \frac{\sum_{s \mid CS} (Sim(q.s, q.c).w_s)}{|CS|} \quad (1)$$

Or w_s is the weight associated for each attribute s, CS is all the simple attributes in q and c, $|CS|$ its cardinality , q.s (or c.s) represent the simple attribute of q (or of c), and sim (q.s, c.s) is the similarity between these two attributes. Thus Sim (q.s, c.s) is defined as follows:

$$Sim(q.s, c.s) = \begin{cases} 1, & if\ v_{q.s} = v_{c.s} \\ 0, & else \end{cases} \quad (2)$$

where v***q.s (or* vc.s)** is the value of this attribute in q (or in c)

B. Configuring the Behavior of the CBR Process

TABLE I. Target Case (Query) and Source Case

Attributes	Target case (Query)	Source case
Code	_30AP10X113	_30AP10X113
Description	Dispositif_de_mise_à_la_terre	Dispositif_de_mise_à_la_terre
Code Symptom	S3.0	S513.0
Description Symptom	Manque_signalisation	Fausse indication
Code Defect	D529.0	D16.0
Description Defect	Déclenchement_disjoncteur	Défaut nettoyage
Code Cause	C149.0	C8.0
Description Cause	Vibrations_et_vieillissement	Manque entretien
Geographical Parent	_30A	_30A
Date	/	07/06/2005
Code remedy	/	Changement de collies
Description remedy	/	167.0

jCOLIBRI formalizes the CBR knowledge using CBROnto, a knowledge level description of the CBR tasks and a library of reusable PSMs [17]

Configuration of tasks is done in an interactive approach by choosing from a library of reusable methods one that is suitable to solve the selected task.

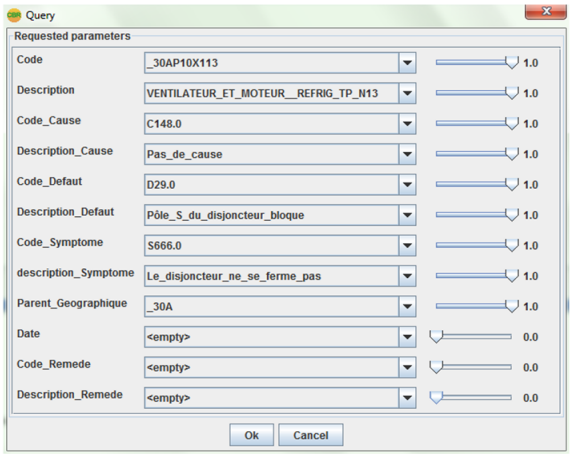

Figure 5. Query task

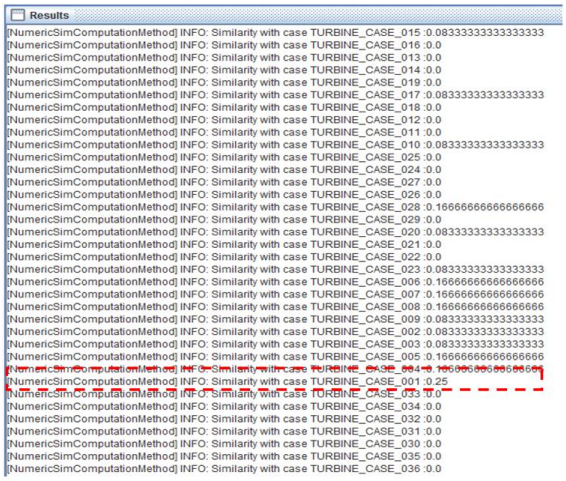

Figure 6. Result of retrieval task

Constraints of the selected task are being tracked during the configuration process so that only applicable methods in the given context are offered to users. In our work, we focus only on the representation and retrieval tasks. To achieve the retrieval task, similarity functions introduced in the previous point must be included after used. The CBR application is finished when all the tasks have been configured. Users can test the system from inside the graphical interface. The first task of the CBR system, obtains the query which contains the description of problem (fault), (Fig. 5) that is going to be used to retrieve the most similar cases, applying retrieval task. (Fig. 6) shows the result of retrieval task, the most similar case is case number 001 with highest similarity 0.25

Table I contain attributes values of the target case (query) and the most similar case source "CASE_001" returned by the system

© 2016 J. Adv. Inf. Technol.

We explain how the similarity value is calculated for a single case and the same principle is applied to all cases in the case base:

- The weights of attributes are set to 1 except the date that is not important for the research.
- Both attributes (code remedy, remedy description) of the solution part are unknown, they represent the solution to our problem.
- The target and source case share the same values for 3 attributes.
- The similarity measure cited above is calculated as follows

Sim (q, c) = 1 * 1 + 1 * 1 + 0 *1+ 0 * 1+ 0* 1 + 0 * 1 + 0 * 1 + 0 * 1 +1* 1 + 0 * 0 + 0 * 0 + 0 * 0 = 3/12 = 0.25

C. Using myCBR

myCBR is an open-source plug-in for the open-source ontology editor Protege [18]. This plug-in consists of the following modules:

- Modelling Tools: These tools extend the existing functionality of Protege for creating domain models and case instances and add the missing functionality for defining similarity measures.
- Retrieval GUI: The retrieval GUI provides powerful features for analyzing the quality of the defined similarity measures. Moreover, it can also serve as the user interface of first prototypical CBR applications.
- Retrieval Engines: For executing the similarity-based retrieval, different retrieval engines are provided.
- Explainer: A dedicated explanation component provides modelling support information as well as explanations of retrieval results for quicker roundtrips of designing and testing.

Four steps are required to develop a CBR application:

1) Generation of case representations

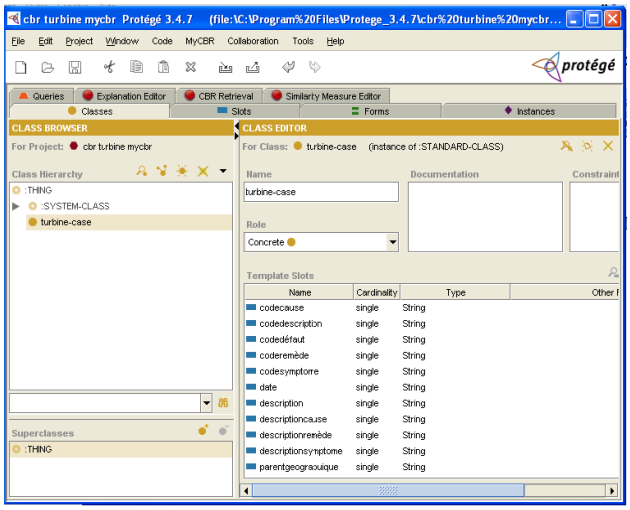

Figure 7. Turbine case representation in myCBR

In myCBR cases and their attributes can be created manually or automatically. A powerful feature provided by myCBR is the easiness of the case representation by CSV data import module [4]. Users have the choice to import data instances in an existing Protege class or to create a new class that is suitable for their raw data. myCBR allows also slots to be added manually using Protege.

Fig. 7 shows our turbine-case class created by exploiting instances of cited ontology in Section 4.A which will be used as query and case values for the retrieval step.

2) Modeling of similarity measure

myCBR follows the local-global approach which divides the similarity definition into a set of local similarity measures for each attribute, a set of attribute weights, and a global similarity measure for calculating the final similarity value. This means, for an attribute value based case representation consisting of n attributes, the similarity between a query q and a case c may be calculated as follows:

$$sim(q,c) = \sum_{i=1}^{n} w_i \times sim_i(q_i, c_i) \qquad (3)$$

here, $sim_i(q_i, c_i)$ and w_i denote the local similarity measure and the weight of attribute i, and represents the global similarity measure [4]. The dataset used in this experiment is simple so we leave the similarity measure definition as the default of myCBR. We only change the weight values of the Id and Class slots from one to zero.

3) Testing of retrieval functionality

myCBR includes an easy to use GUI for performing retrievals and for analyzing the corresponding results. By providing similarity highlighting and explanation functionality myCBR supports the efficient analysis of the outcome of the similarity computation. We tested the 36 records (case) that are excluded from the dataset. Fig. 8 shows one query of these records after retrieving the most similar cases. Another alternative of performing case retrieval is to use a query from cas

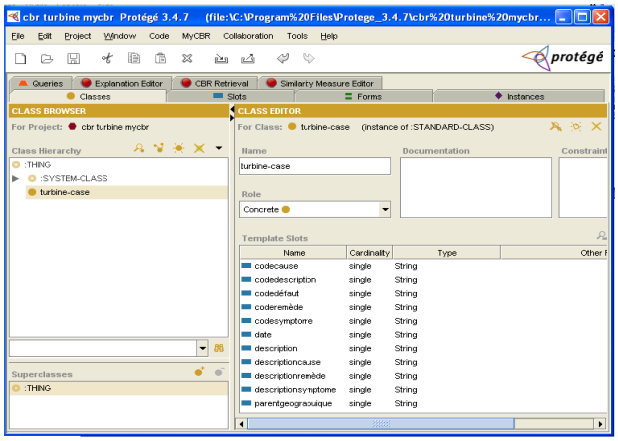

Figure 8. Result of retrieval task

Result of retrieval task Implementation of stand-alone application: myCBR can also be used as a stand-alone Java module, to be integrated in arbitrary applications.

In this application phase, the retrieval engines of myCBR just read the XML files of the created project

© 2016 J. Adv. Inf. Technol.

generated using the plug-in interface and perform the similarity-based retrieval.

VII. Results and Discussion

In this paper, the construction and the design of a CBR system of which the knowledge within is described on ontological form has been presented. Two object-oriented ontology based CBR frameworks jCOLIBRI and myCBR are examined.

A fault diagnosis of steam turbine system is built by using the two selected framework. For the CBR process we centered our work on two phases of the cycle for instance the elaboration, and retrieve of the cases based on the domain ontology. A part of results provided by the retrieval phase shown in the Fig. 6 and Fig. 8 are synthesized in Table II.

- During the implantation of the fault diagnosis application, it was found that jCOLIBRI is user friendly and efficient to quickly develop application. Our motivation for choosing this framework is based on a comparative analysis between it and other frameworks, designed to facilitate the development CBR applications. jCOLIBRI enhances the other CBR shells:

CBR*Tools [19], CATCBR [20], IUCBRF [21], Orenge [22]. jCOLIBRI is an open source framework which uses CBROnto a task / method ontology of which supplies the necessary vocabulary to describe elements implied in the CBR process and their interface layer provides several graphical tools that help users in the configuration of a new CBR system. Another decision criterion for our choice is the relative ease of ontologies integration. jCOLIBRI offers the opportunity to incorporate ontology in the CBR application to use it for case representation and content-based reasoning methods to assess the similarity between them.

TABLE II. The Retrieval Phase Results

Id	1	2	3	4	5	6	7	8	9	10
Similarity Results whith jCOLIBRI	0.25	0.083	0.083	0.166	0.166	0.166	0.166	0.166	0.083	0.083
Similarity Results whith myCBR	0.33	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11

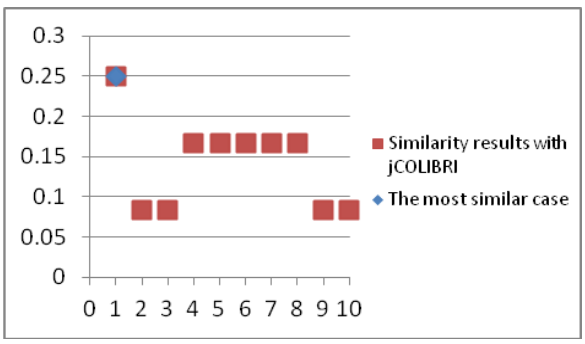

Figure 9. Similarity results with jCOLIBRI

The retrieval phase was successful for the selected dataset. Among 36 cases in our case base, represented diagrammatically by Fig. 9, the system returned the most similar case to the target problem with the highest similarity of 0.25, for instance case number 1.

During the implantation of the fault diagnosis application using myCBR, it was noticed that myCBR is a tool for rapid prototyping of a new CBR application. Quickly, users may have a running standalone CBR application just using the CSV importing feature. myCBR allows to build the case structure and the case base by parsing the provided CSV file. It also saves time and efforts, making the development of a new CBR application done inside Protege possible. The retrieval phase was successful for the selected data set. Among 36 cases contained in the case base, represented diagrammatically by Fig. 10; it returned the most similar case to the target problem with the highest similarity of 0.33, for instance case number 1. We notice that case number 1 is considered to be the best case for both system with different similarity value 0.25 and 0.33 respectively. The result confirms the search approach for these tools.

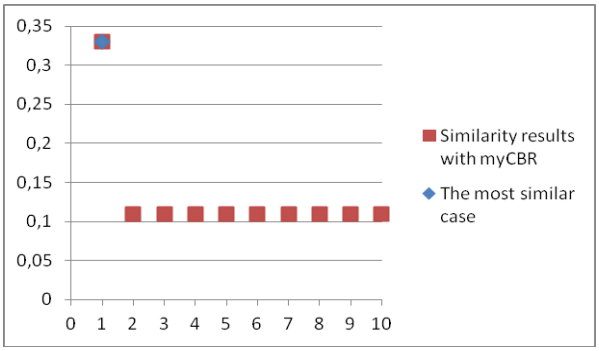

Figure 10. Similarity results with myCBR

VIII. Conclusion

This paper presents the advantages that ontologies provide when used as part of a CBR system. In the presented work, ontologies are used as:

- The cases/queries definition languages independently of the persistence media we use for cases;
- The knowledge structure where the cases are embedded, so that the persistence media for the case base uses the ontology formalization language; and
- The knowledge source to get semantic reasoning processes, like retrieval, similarity and adaptation.

The main innovation is that the domain ontology is used to represent all this knowledge and DLs-like formalism to reason.

The usage of such KI-CBR for diagnostic purposes on an industrial application, demonstrates the power of integrating conceptualized domain knowledge in a decision solving problem approach such as CBR [23]. The obtained results are promising and deserve to be tested on a larger case base.

The paper also describes how rapid prototyping of 9CBR applications and easy integration of ontologies is

© 2016 J. Adv. Inf. Technol.

well supported by jCOLIBRI. During the implementation of our fault diagnosis application we have found the both CBR frameworks very useful and efficient to develop applications. It proves the advantages of using frameworks for development of new applications which helps improve software quality.

REFERENCES

[1] A. Aamodt and E. Plaza. "Case Based Reasoning: Foundational issues, methodological variations and system approaches," *AI Communications*, IOS Press, vol. 7, no. 1, pp. 39-59, 1994.

[2] J. Lieber, "Contribution à la conception de systèmes de raisonnement à partir de cas," HàDR de l'université Henri Poincaré–Nancy1, janvier 2008.

[3] J. B. Tomás, J. A. G. Calero, and B. D. Agudo, "JCOLIBRI: An object-oriented framework for building CBR systems," in *Advances in Case-Based Reasoning, Lecture Notes in Computer Science*, Springer Berlin/ Heidelberg, vol. 3155, 2004, pp. 32–46.

[4] A. Stahl and T. R. Roth-Berghofer, "Rapid prototyping of CBR applications with the open source tool myCBR," in *Advances in Case-Based Reasoning*, R. Bergmann and K. D. Altho, ed. Springer Verlag, 2008.

[5] J. Charlet. (2004). L'ingénierie des connaissances entre science de l'information et science de gestion. Rapport de recherche SIC 805. [Online]. Available: http://archivesic.ccsd.cnrs.fr.

[6] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisit*, vol. 5, no. 2, 1993.

[7] W. N. Borst, "Construction of engineering ontologies for knowledge sharing and reuse," CTIT Ph.D. dissertation, Universiteit Twente, 1997.

[8] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge engineering: Principles and methods," *Data and Knowledge Engineering*, vol. 25, no. 1-2, pp. 161-197, 1998.

[9] J. Aha, "The omnipresence of case-based reasoning," in *Science and Application. Knowledge Based Systems*, vol. 11, no. 5-6, pp 261-273, 1998.

[10] B. Fuchs, J. Lieber, A. Mille, and A. Napoli, "A general strategy for adaptation in case based reasoning," Technical report RR-Liris-2006-016 , LIRIS /CNRS/INSA (2006).

[11] A. Mille, "From case-based reasoning to traces-based reasoning," *Annual Reviews in Control*, Elsevier, vol. 30, pp. 223-232, 2006.

[12] J. R. Garcia, B. D. Agudo, P. G. Calero, and A. Sanchez, "Ontology based CBR with jCOLIBRI," in *Proc. 26th SGAI Int. Conf. (AI-2006)*, UK: Springer, 2006, pp. 149–162.

[13] N. Dendani and M. T. Khadir, "A case based reasoning system based on domain ontology for fault diagnosis of steam turbines," *International Journal of Hybrid Information Technology*, vol. 5, no. 3, pp. 89-103, July 2012.

[14] Protégé (2009). Ontology Editor and Knowledge Acquisition System. [Online]. Available: http://protege.stanford.edu/

[15] H. Knublauch, R. W. Fergerson, N. F. Noy, and M. A. Musen, "The protege owl plugin: An open development environment for semantic web applications," in *Proc. Semantic Web-Iswc*, vol. 3298, pp. 229-243, 2004.

[16] O. Djeddi and M. T. Khadir, "Degnin and realisation of aligning ontology algorithm in OWL," M.S. thesis, Badji Mokhtar University Annaba Algeria, 2009.

[17] J. A. R-Garcïá, A. Sïánchez, B. D-Agudo, and P. A. G. Calero, "jCOLIBRI 1.0 in a nutshell. A software tool for designing CBR systems," in *Proc.10th UK Workshop on Case Based Reasoning*, M Petridis, ed. CMS Press, University of Greenwich, 2005, pp. 20–28

[18] J. H. Gennari, M. A. Musen, R. W. Fergerson, W. E. Grosso, M. Crubezy, H. Eriksson, N. F. Noy, and S. W. Tu. "The evolution of protege: An environment for knowledge-based systems development," *International Journal of Human-Computer Studies*, vol. 58, no. 1, pp. 89-123, 2003.

[19] M. Jaczynski and B. Trousse, "CBR*Tools: An object-oriented framework for the design and the implementation of case-based reasoners," in *Proc. the 6th German Workshop on Case-Based Reasoning*, Berlin, 1998.

[20] C. Abasolo, E. Plaza, and J. L.Arcos, "Components for case-based reasoning systems," *Lecture Notes in Computer Science*, vol. 2504, 2002.

[21] S. Bogaerts and D. Leake, *IUCBRF: A Framework For Rapid And Modular Case-Based Reasoning System Development*, November, 2004.

[22] Orenge. [Online]. Available: http://www.ovitas.com/PDF/orengeWhitepaper.pdf

[23] N. Dendani and M. T. Khadir, "A fault diagnosis application based on a combination case-based reasoning and ontology approach," *International Journal of Knowledge-Based and Intelligent Engineering Systems (KES)*, vol. 17, no. 4, pp. 305–317, Novembre, 2013.

© 2016 J. Adv. Inf. Technol.