World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:1, No:10, 2007

A Knowledge Engineering Workshop: Application for Choise Car

Touahria Mohamed, Khababa Abdallah, and Frécon Louis

***Abstract*—**This paper proposes a declarative language for knowledge representation (Ibn Rochd), and its environment of exploitation (DeGSE). This DeGSE system was designed and developed to facilitate Ibn Rochd writing applications. The system was tested on several knowledge bases by ascending complexity, culminating in a system for recognition of a plant or a tree, and advisors to purchase a car, for pedagogical and academic guidance, or for bank savings and credit. Finally, the limits of the language and research perspectives are stated.

***Keywords*—**Knowledge representation, declarative language, Ibn Rochd, DeGSE, facets, cognitive approach.

I. Introduction

A major field of the Artificial Intelligence (AI) is to design a declarative systems, commonly called Knowledge Based Systems (KBS) or Expert Systems (ES) in the second generation. These systems are characterized by a separation between knowledge necessary to describe a problem and the mechanisms exploiting this knowledge [1]. This separation describes knowledge independently of their later use. It facilitates on the one hand the modification and the addition of new knowledge at the base ; in addition, one can provide justifications of the system behavior.

Such systems are above all the software which simulates the reasoning of an expert. Thus it draws its motivation and its justification in the human experiment [2], and must validate its semantic.

Thus the artificial intelligence is at the crossroads of data processing and the "cognitive sciences", who aim understanding how knowledge is born, used, evolved and transmitted. If the purpose of AI program is to imitate an intelligent behavior. It is necessary for it as well as an individual to have access to knowledge.

However, the efficiency depends on the Knowledge representation [3]. For reason, the knowledge representation is a key question of the AI. It effectively consists in formalizing the concepts of the real world in exploiting form by "intelligent" programs [4].

November 17, 2005.

Touahria Mohamed, Ferhat Abbas University of Setif, Computer Science Department 19000, Algeria, (e-mail : touahriam@lexpress.net).

Khababa Abdallah, Ferhat Abbas University of Setif, Computer Science Department 19000, Algeria, (e-mail : khababa_abdlh@yahoo.fr).

Frecon Louis, Computer Science Department, INSA of Lyon F69621 Villeurbanne, France, (e-mail :louis.frecon@insa-lyon.fr).

Knowledge results from a *correspondence* (translation) of the real world in a symbolic system ; system makes it to compute [5]. Then the *cognitive step* consists in ensuring this correspondence by known processes.

According to Vogel [6], knowledge relates to the still impregnated information of the enonciator vision. In artificial intelligence field, expertise knowledge is usually divided in various categories. Among those, most usual present themselves according to various criteria : *scientific knowledge* and *empirical knowledge*, *major knowledge* and *surface knowledge*, *factual knowledge* and *procedural knowledge*, *expert knowledge* and *fortuitous knowledge*.

The definition of the formalism in the form of production rules exploited by a general mechanism the inference-engine-uses a representation paradigm and knowledge exploitation [7], true incarnation of the declarative programming.

The expertise collection process is based on a distinction between axis of the models and axis of the paradigms [8]. The problems of the knowledge acquisition are centred on the level of knowledge, knowledge which it is necessary well to study for better formalizing and structuring.

We can postulate that the treatment of knowledge includes :

- *a factual comprehension*, inductive product rules starting from the observed facts;
- *a deductive*, cognitive phase of decisions, according to a logic guided by a strategy to control the validity of the obtained conclusions.

Our main purpose is to show the interest and the contributions of Knowledge Representation Language (KRL) Ibn Rochd [9] to resolve problems of representation, and in second time achieving a Cognitive Engeneering Workshop.

The work presented in this paper is situated in the context of an object approach, in the Sygemor style [10] that uses the *Networks of Worlds*, (see Shirka [11]). We find two essential notions here : the *classes* and the *instances* or the *objects.*

II. Knowledge Representation Language

A Knowledge Representation Language is a support of a reasoning : his power ia same than a programming language, but it is less directive than its form [9].
The in demands aspects are :

- *descriptive capacity*, domain knowledge, laws and system states…;
- *capacity heuristics*, knowledge useful for the problems resolution, operational concepts, strategies : focalisation, planning ;

- *granularity*, assured by very autonomous and elementary constructions, facilitating incrémental development and evolutive maintenance [9].

A Knowledge base is the expertise projection in a Knowledge representation language.

A. Knowledge Representation in Ibn Rochd

In Ibn Rochd, two types of knowledge exist : the *descriptive knowledge* and the *knowledge operating*.

1. Descriptive Knowledge

KRL describes the frame of the problem (knowledge), model (classes) and objects. When several objects, possess a structure and a common behavior, it is very useful to group together them in a common mould : a **class**.

♦ **The Classes**

A class is a general model from which *objects* can be genered : intances of class.

Principes : A class represents a group of attributes and behaviours. A subclass object inherits both properties and attributes.

The classes :

- offer a hierarchical organization of the knowledge ;
- federate a lot of information which serves for describing the abstract model, and using knowledge ;
- represent generic information, and realize the descriptive model of the knowledge.

The *inheritance of properties* allows to group together the similar objects in a subclass and the more general objects in *a super-class (or class-mother)*.

A class represents a general model with a set of *attributes.* One classe is build up :

- in a *autonomous* way, all the attributes being defined at its level ;
- by *refinement* of a class-mother, whose new class resumes the attributes, to add its appropriate characteristics : we call a *simple inheritance*.

♦ **The Attributes and the facets**

The *Attributes* possess a name, and a set of facets, statements (declarations) or characteristics (sometimes optional). *Facets* represent all the characteristics for every attribute, as : *type*, *Cardinality*, *Interval*, *Domain*, *Question*, *Default*, If Need, *Validity.*

Syntax :

CLASS : < class name>
[**INHERIT_OF** : {< class name>}*]
[**ATTRIBUTE** :
{< attribute name> :
TYPE :integer|real|string|[class-name]
[**DOMAIN** : (value_1, …, value_n)]
[**CARDINALITY** : (single|multiple)]
[**INTERVAL** :[limit_inf, limit_sup]]
[**IF_NEED** : [call of procedure]
[**VALUE**: expression |
QUESTION : "character string" |
DEFAULT : value_i] }*]

{}*: possibility to have zero or several cases (occurrences).

Semantic :

- DOMAIN and INTERVAL used to restreiendre the type ;
- VALUE introduces a calculable attribute, so assuring the coherence of the object ;
- QUESTION activates (starts) a dialogue ;
- DEFAULT introduces a defect value of known and specific value ;
- TYPE : indicates attribute type.
- Genericity : we can define macro, hyper-classes in the style below, where #object indicates a substitutable symbol.

CLASS LIST (#object)
first : [CONTAINER(#object)]
last : [CONTAINER(#object)]
cardinal : VALUE (if first = NOTHING then 0 else first.nb)
END LIST.

♦ **Objects**

Objects are instances or *objects* of this class. Objects have common properties defined by their class, in the form of lists of couples (**attribute, value**) where *attribute* indicates an class attribute or the super-class, and where *value* is compatible with the attribute facets.

A specific object already having a name can be directly appointed by this name, or indicated anonymously by $, followed by the name of a class of the object : example : $bird indicate *some* bird.

2. Knowledge Operating

They express the expertise in a declarative form (rules). The production rules represent the knowledge representation formalism used in expert systems. The rule consists of four components : Condition, Action, Explanation and Score number (priority).

The action part defines the actions to be taken when the conditions in the Condition part are satisfied. The explanation part contains explanation related to the firing of the rule. Score Number is a number, which reflect the relative importance of the rule with respect to the other. The rule :

- represents the dynamic part of the knowledge base (KB) ;
- offers legibility and ease of writing ;
- offers the possibility of balancing the knowledge.

Syntax :

R <number.number>
{**IF** <premise>}$^{\oplus}$
THEN
{<action>}$^{\oplus}$
BECAUSE <explanation>

{}$^{\oplus}$: expressed at least once.

♦ **Premises**

The *premises* compare *terms* : attribute, constant, class instance, specific or anonymous. They use for *comparative operators*, relational operators among " =, <,>, <=, >=, IN and EXCEPT (for the list), DIVIDE and MUTIPLE ".

♦ **Actions**

In this part we can make operations of adds, modifications or retracts information from the objects base.

- Affectation operation: = impose a new exclusive value.
- Addition operation + = adds a value to a multiple attribute.
- Retract operation (deletion) -= removes a value (or some) from a multiple attribute : the new value is the ensembliste difference between the previous value and modifying..

Ibn Rochd offers a dynamic management of the operations on the instances of class and allows to :

- create an object : the action CREATE creates a new instance of the indicated class ;
- remove an object : the action KILL (name of object) inverse operation of CREATE, removes all values from this object.

3. Communication Operation

They aim at acquiring data or to post (show) results.

- **READ**

This operation allows to read base entity (attribute value).

Examples:

```
READ<object-designation>.<attribute-name>
READ "characters"
+$<class-name>.<attribute-name>
CLASS  MENU
ATTRIBUTES :
 POST : LIST (CHAIN)
 SHOW :  METHOD
 {     SCREEN.CLEAN
       POST.WRITE
  }
 CHOICE : TYPE  INTEGER
       VALUE :
   {   *.SHOW
         WRITE « YOUR CHOICE ? »
         READ CHOICE
    }
END  MENU
```

- **WRITE**

This action displays texts, information and/or some entity and/or one (or some) relation values associated to a given entity.

- **Screen management**

For a bigger legibility of the applications, Ibn Rochd offers the possibility to the knowledge engineer to manage its screen. Before every operation of writing, the construction IN(line_number, column_number) positions the cursor in the place specified by the indicated coordinates. RAZ SCREEN : Clean screen.

4. Piloting and Meta-Operations

Ibn Rochd offers the knowledge engineer the possibilities of intervention on rules examination order.

- INSPECT {number of rule}*), imposes the priority evaluation of the mentioned rules.
- FREEZE ({number of rule}*): inverse operation INSPECT, makes inactive rules, whatever is the order to their place in the " conflicts list ".

B. Comments

Ibn Rochd use three types of comments :

- comments without effect, beginning with the symbol '/* ' ;
- comments with effect used by the engine in the writing of the track of the reasoning (explanation), beginning with the symbol '!' ;
- BECAUSE : justification.

III. The Environment / Workshop DeGSE

DeGSE workshop is an environment of Expert Systems Development.

This workshop :

- Supports the declarative language Ibn Rochd (rules base and objects) ;
- analyses, codes, archives and manages knowledge bases described in this language ;
- proposes a Knowledge Editor (KE) to write them and modify them;
- works in deductive mode (natural deduction), with possibility of track of its inferences.

This workshop is dedicated to :

- the Development : experts and knowledge engineer use KE and Inference Engine to produce and finalize a KB ;
- the Exploitation : the end users use the Inference engine coupled with the KB (specific of the application).

In the state, DeGSE is operational under Windows; it contains 4 modules including approximately 24 thousand lines of C++.

IV. Validation : Case Study

The validation is a check of the product with regard to the initial specifications. At present tests train is organized by increasing complexity.

This benchmark validates the language and the first model of DeGSE. This task frequently situates on the most critical road of the project and constitutes one of the main elements for the definitive work.

The validation of expert systems differs substantially from that of the other computer programs as far as :

- the specifications of expert systems are called to evolve, sometimes in a very sensitive way, throughout the cycle of development ; therefore, the validation of the expert systems cannot consist only of a check of the adequacy of the product to its specifications ;
- there often are no objective criteria to decide if the found result is indisputably the best.

Prolem

The goal is to Develop an Expert System for car Choise.

Analysis

Seen the important number of **countries** and **groups** (Psa, Vag, Bmw, Fiat) builders (manufacturers) of cars and *type* of cars that every country can produce or groups ; we limit ourselves to four big producing countries represented in Algeria : France for Renault, Germany for the Group Volkswagen (VW, Audi, Skoda, Seat), Japan for Nissan and the USA for General Motors (Opel, Daewoo).Modelling a knowledge base for such application revealed a delicate task.

Realization

Was structured and coded according to the agreements of writing of the KRL Ibn Rochd, at present, this knowledge base contains :

- 5 classes : *car*, *technique*, *custumer*, *performance*, *equipment* ;
- more than 700 *objects* : 256 for Renault, 205 for Volkswagen and 140 for Nissan ;
- 175 *rules*.

After interpretation of the base above, the system detects and indicates all errors in the writing of the base to the user. After correction of the base, the system creates the data structures necessary for the evaluation of the base. The class *custumer* (friendly user) allows for one person to formulate a choice. The user answers a series of questions to start the reasoning, by exploiting each answer as well as possible. The system generates relevant solutions. The next Fig. 1 presents the Knowledge Editor.

V. Conclusion

In this paper, we have :

- conceived Ibn Rochd as a declarative centred language object for the knowledge representation;
- conceived his environment (DeGSE), which codes bases, lists the classes, the attributes, the objects, the rules and in the execution, supplies a formed track.
- modeled and exploited a knowledg base for choise car.

Ours experiments confirme the interest of this choise.

In the future, it is necessary to :

- introduce a revisable priority (heuristics) by meta-rules according to the obtained results (learning) ;

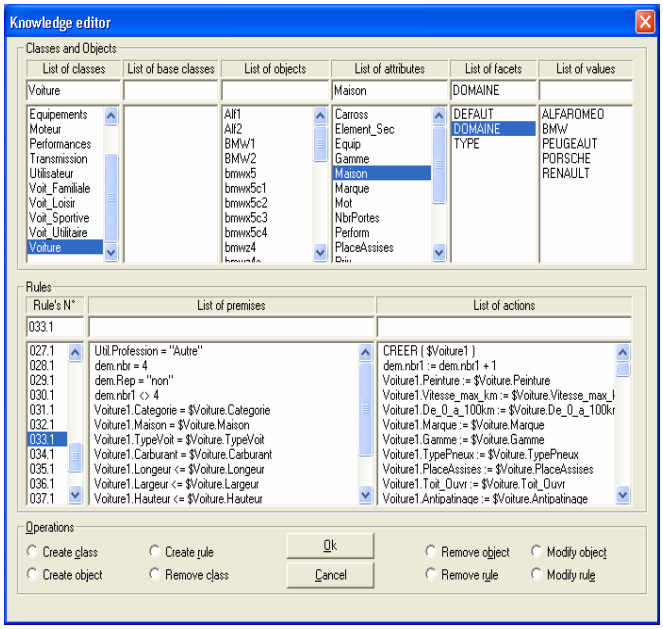

Fig. 1 Copy of screen

- allow a coupling of DeGSE with the relational data bases: consider DeGSE as a pit cooperating with data bases ;
- integrate features of coherence control, to pass from a editor to a publisher / tax auditor ;
- integrate an online help ;
- introduce a function of compilation (translation C++ or in Java) applications focusing.

The behavior of the new system shows itself rather supple. In order to express, to represent and finally to exploit easily its knowledge by the user.

References

[1] N. Tayar, "Management of the versions for the construction incrémentale and shared by knowledge bases", Doctoral thesis of state, Joseph Fourier university, Grenoble, 1995.
[2] M. Touahria , L. Frécon, "Workshop of Cognitive Genius for Ibn-Rochd 1/2", I.N.S.A of Lyon, Computer Science Department, 2002.
[3] Y. Hamon, "Integration of Techniques of Artificial intelligence in the Workshops of Software engineering, application to the workshop Softpen: SAIGL", Doctoral thesis in Data processing Lyon 1, 1993.
[4] G. Massini, A. Napoli, D. Colnet, D. Léonard, K. Tombre, "object-oriented languages, languages of classes, languages of frames, actors languages" Inter-Editions, Paris, 1989.
[5] B. Decourbe, "factual Understanding and cognitive decisions, Seen again(Revised) by Artificial intelligence", Flight 1, n°2, 1987.
[6] C. Vogel, "Cognitive Genius", Masson, 1988.
[7] G. Caplat, "Cognitive Modelling and resolution of problems", PPUR, Lausanne, 2002.
[8] C. Vogel, "KOD : the implemented", Masson, 1990.
[9] M. Touahria M, L. Frécon., "Workshop of Cognitive Genius for Ibn-Rochd 1", I.N.S.A of Lyon, Computer Science Department, 1999.
[10] Y. Sohbi, "Study and Realization of SYGEMOR: environment for Systems Multi Experts.Centred Objet", Doctoral thesis of the INSA of Lyon, 1990.
[11] F. Reichenmann, "SHIRKA : management system of centred knowledge bases objec"t, Manuel of reference, INRIA/ARTEMIS, Grenoble, 1988.
[12] A. Khababa, M. Touahria, L. Frécon, "Cognitive Modelling a system for pedagogical and academic guidance CORUS", Computer Science Department, Sétif, (DZ), 2004.