Building and Sharing Large Knowledge Bases in Molecular Genetics

François Rechenmann
INRIA Rhône-Alpes & IMAG/LIFIA
46, avenue Félix Viallet
38031 Grenoble Cedex 1, FRANCE
Tel. (+33) 76 57 45 63 Fax (+33) 76 57 46 95
Francois.Rechenmann@inria.fr

ABSTRACT

Large volumes of genomic sequences are being produced at increasing rates for many living organisms. The computer analysis of these data, using numerical and symbolic methods, generates various and numerous objects (such as genes, signals, repetitive sequences, etc.), the status of which is often hypothetical.

Object-based knowledge models allow to describe these objects which are inter-related and sometimes organised in higher level structures, such as operons in the case of bacterial genomes. Annotations under the form of hypertext can be associated to these descriptions. Methodological knowledge on the analysis methods themselves can also be described.

These capabilities are presented on the example of two operational object-oriented knowledge bases which have been designed in the context of a tight collaboration with the "Laboratoire de Biométrie, Génétique et Biologie des Populations" of Claude Bernard University in Lyon.

It is argued that such knowledge bases can become a very efficient medium for collaborative research. A set of laboratories or research groups, working together on a same living organism, or studying a same set of phenomena across several organisms, can incrementally build, maintain and share a consensual knowledge base. Relying on the metaphor of paper submission, protocols of interaction between the various contributors and the consensual knowledge base are presented and discussed.

INTRODUCTION

Genes are the physical supports of heredity. Coded as long textual sequences in a four letters alphabet, they form, together with several other types of objects, the genomes of the living organisms. Their controlled transcription and translation produce the proteins required to develop and to maintain the organism. Several large scale sequencing projects have been initiated these last five years. The Human Genome Project is the most famous one, but a lot of efforts are being devoted to get the full sequence of small genomes, such as the bacteria *E. coli* or *B. subtilis*, the baker's yeast *S. cerevisiae*, or the *C. elegans* worm, to name only some of them.

The volume of data, under the form of long genomic sequences, is therefore steadily increasing, but the investigation process to understand the organisation and the functions of these sequences is a very long and tedious one. Sequence data bases have been developed which allow to store sequences produced by the different sequencing projects, together with some annotations, under the form of keywords and plain text. Biologists make use of various advanced computer methods to analyse these sequences (see e.g. [1,2]), thus producing secondary data, and through mental inductive processes, new knowledge. Paper is clearly not an adequate support any more for storing, manipulating and storing these increasing volumes of knowledge; nor are classical data models. In order to manage knowledge on the sequences and their interpretation, more powerful data and knowledge models are thus required.

Object-based models appear to be quite adapted to these requirements. Developed and used in several areas of computer science, such as programming languages, knowledge representation and database management systems, object models share two basic characteristics [3] : the hierarchical structuration into classes and subclasses and the internal organisation of objects as a set of variables or attributes which can make reference to other object definitions. These descriptive capabilities are highly appreciated in the context of knowledge representation and object-based models are being experimented for the development of several knowledge bases in molecular biology (see for instance [4, 5, 6, 7]).

1. OBJECT-BASED KNOWLEDGE REPRESENTATION

The requirements imposed on knowledge models by the very definition of knowledge-based systems lead however to clearly distinguish object-based knowledge models from object-oriented programming languages. The absence of user-defined methods describing instance behaviour, the complete visibility of the internal

© JIPDEC 1993

structure of classes and the dynamic classification mechanism of incomplete objects, are the major differences which are explained hereafter.

1.1. Knowledge-based systems

Knowledge-based systems are characterised by the attempt to clearly separate knowledge from its exploitation mechanisms. The advantages of this separation are well known. Knowledge can be expressed without anticipating on the ways it will be used, within the limits of the semantics of the model. The resulting knowledge bases are therefore easier to read and thus to maintain. Moreover, explanations on the behaviour of the system, together with justifications of the results, can be generated. It must be underlined however that these advantages are more potential than real. The generation of user-understandable explanations in particular is a goal still to be attained.

The design of a knowledge base can rely on different knowledge models. Rules have been extensively used for the development of the so-called "expert systems". They support well-known and well-understood reasoning modes. Object-based models offer however an alternative for large knowledge bases which need to be highly structured. The object model indeed offers several advantages. It first allows for the development of both data and knowledge bases. It structures knowledge into units organised in a hierarchical structure which is easy to maintain : adding a new piece of knowledge is to search for its correct location in the structure. The object model supports inference mechanisms, such as inheritance, and more specifically inheritance of default values, and procedural attachment. However, these mechanisms are often considered as insufficient and, most of the time, objects are used in conjunction with rules. In such a combination, rules express decisional knowledge, while objects are used to model static descriptive knowledge. The set of objects is thus considered as a structured factual base, which can be made reactive with procedural attachments.

Although such hybrid knowledge representations offer good practical modelling facilities, they do possess all the well-known limitations and drawbacks which derive from mixing up different formalisms together. We therefore advocate the use of "pure" object-based knowledge models, in conjunction with powerful classification mechanisms.

1.2. Classes, sub-classes and instances

An object-based knowledge model allows to represent knowledge in descriptive units (figure 1) which are made up of attributes (often called "slots" after the famous paper of Minsky [8], which is often considered as the starting point of numerous studies on frames [9]), to which constraints on their values are attached, as types and/or lists and intervals of admissible values, or arbitrary complex predicates.

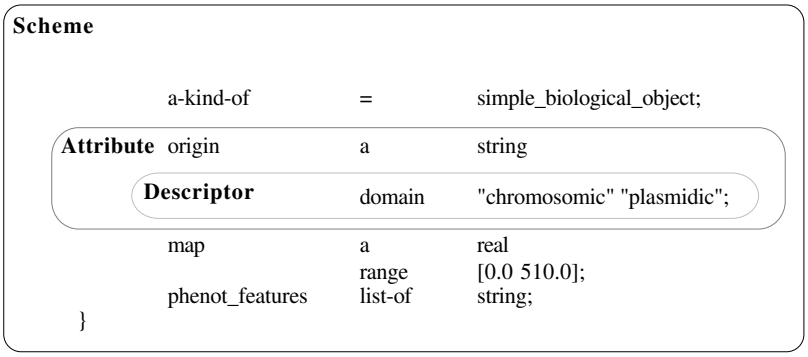

Figure 1 — Structure of a class as a knowledge description unit. Every class has a name and is composed by a list of attributes. Each attribute has itself a name and is described by a list of descriptors with their associated values.

In class-oriented models, a clear separation is made between descriptions of classes and descriptions of elements of these classes, called instances. An instance is only made up of attribute values, while classes incorporate the knowledge on these attributes. A class can be refined into sub-classes, while instances cannot be specialised and appear as leaves in the hierarchy. A class-oriented model makes thus the distinction between the AKO (a kind of) link between a class and its super-classes and the ISA link between an instance and the class to which it belongs (figure 2). The specialisation link between a class and its sub-classes has a strict interpretation : attribute descriptions can be added and refined, but neither extended nor hidden. In such a way, any instance belongs to all its super-classes and its attribute values satisfy all of their constraints.

Classes are also inter-related via their attributes. The type of an attribute can indeed be specified as the name of another class (figure 2). The value of this attribute in an instance must therefore be a reference to an instance of the class. These "horizontal" links between classes can support a navigation process inside the knowledge base.

The object-based knowledge model supports various inference mechanisms, such as default value inheritance, procedural attachment, and classification which determines the position of a given object in the specialisation hierarchy (§1.3).

```
{ protein-gene
  a-kind-of     =         gene;
  ref-num       var-name          nseq;
  origin        var-name          nsrc;
  type          a                 string
                domain    "CDS" "ORF" "URF"
                default   "CDS";
  rbs           a                 RBS
                if-needed
                  { cr-rbs
                  ref               var<-   nseq;
                  source            var<-   nsrc;
                  new-rbs           var->   rbs } }

{ aceE
  is-a          =         protein-gene;
  ref-num       =         ECOACEX;
  origin        =         GenBank;
  type          =         CDS;
  rbs           =         rbs-aceE }
```

Figure 2 — The **AKO** and **IS-A** attributes. The class **protein-gene** has been defined as a specialisation of the class **gene**. **aceE** is one of its instances. It is said to be a complex instance since the value of its attribute **rbs** (ribosome binding site) is a reference to an instance of the class **RBS**, as specified in the description of the class. The value can be obtained through the execution of the attached method **cr-rbs** .

Objects in declarative knowledge models present several important differences with objects in imperative programming languages :

- Since they have to be interpreted, both by human readers and by the computer, their contents must be readily available ; attribute descriptions and values are not hidden inside the objects.
- Incomplete instances may be created, i.e. some of their attributes may not possess values. If required, these values can be determined using the knowledge of the classes.
- In object-oriented programming languages, the methods attached to a class describe the behaviour of its instances. In a knowledge model, the behaviour of the instances of a class is dictated by the description of this class which is interpreted according to the (fixed) semantics of the model.

1.3. Classification of incomplete objects

Class-based models efficiently and correctly support classification mechanisms. An instance, which is known to belong to a class, is matched against the sub-classes. If the constraints attached to the attributes in a sub-class are satisfied by the attribute values of the instance, the sub-class is retained and its sub-classes are recursively tested (figure 3).

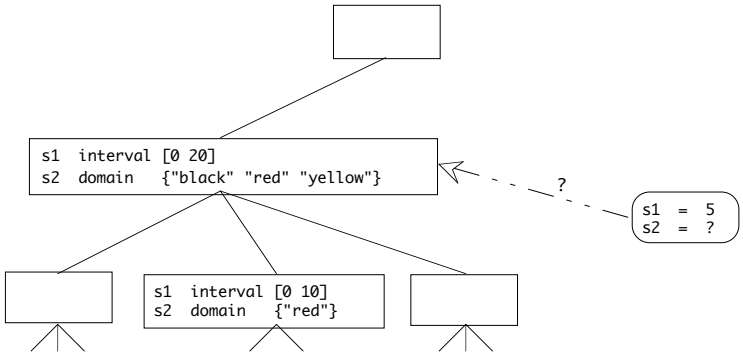

Figure 3 — Instance classification. The values of the attributes in the instance are matched against the constraints in the current candidate class. The value of the attribute **s1** is thus compared with the interval **[0 20]**. The value of attribute **s2** is unknown in the instance ; matching will be partial. The class will only be **possible** for the instance, but the classification will nevertheless proceed on the sub-classes.

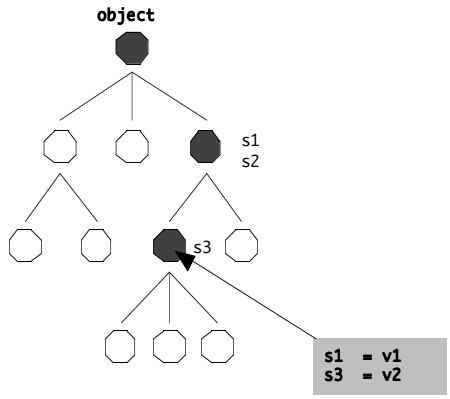

Figure 4 — Once an object has been created by instantiating a class of the knowledge base, it inherits the knowledge, i.e. the attribute descriptions, associated with the instantiation class and with its super-classes, up to the root of the specialisation graph. The object can thus be considered as an instance of any of these classes, including the class `object`.

The purpose of the classification process is thus to determine classes of the knowledge base to which the object could belong according to its characteristics, i.e. its attribute values. The aim of the process is of course to increase the amount of knowledge about the object. Indeed, knowing that an object can belong to a class means that its missing attribute values have to satisfy the constraints attached to the attribute descriptions in the class and that the knowledge in the class can be used to complete the instance (figure 4).

Since every instance belongs to the class `object`, i.e. the root of the specialisation graph, the classification process can start from this class. In this case, all the classes of the knowledge base are candidates to be examined. This leads to a problem of efficiency and to a problem of semantics. It is clear that attempting to match the instance with all the classes of the knowledge base is highly time-consuming. Moreover, all these classes are certainly not significant for the object under consideration. Even if the classes are sub-classes of `object` and inherit some basic knowledge from it, they do model different concepts. There is certainly no sense to test if an instance of `person` could belong to the class `motorcycle`, for instance. It is thus necessary to restrict the classification search space.

The first solution is to consider that, in the knowledge model, immediate sub-classes of `object` are roots of independent class families (figure 5), called concepts. An instance of a class inside a concept may then be only matched against the classes inside the same concept.

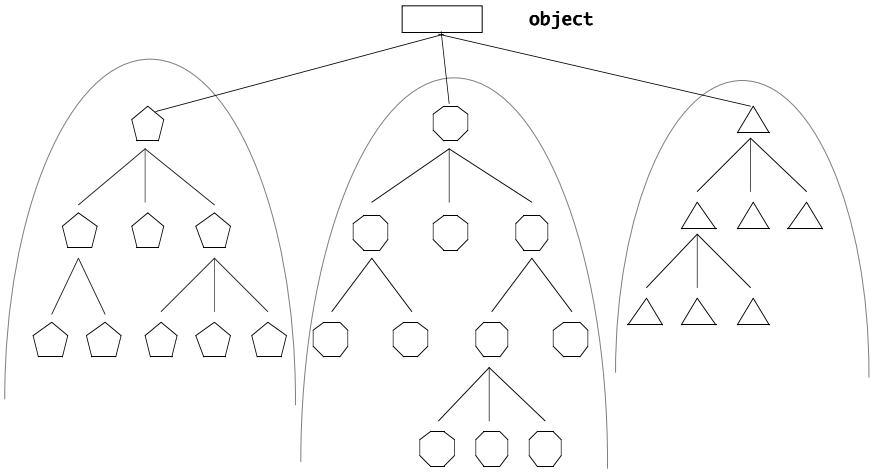

Figure 5 — In order to limit the search space of the classification process to meaningful classes, the knowledge base is organised in families of classes called concepts. Any direct class of **`object`** initiates a new concept.

As presented before (figure 2), a complex instance can make references to other objects through the value of some of its attributes. The classification of such an instance thus leads recursively to the partial classification of several other instances in different concepts of the knowledge base. Explanation on a complex instance classification can thus be generated at various depth levels.

2. MODELLING BIOLOGICAL KNOWLEDGE

Systematic genome sequencing projects are producing data at an ever increasing rate. As an example, the complete sequence of the chromosome III of *Saccharomices cerevisiae*, the baker's yeast, has been recently obtained by a large consortium of European laboratories [10]. But these texts, written in a 4 letters alphabet, are still to be semantically interpreted : subsequences coding for proteins, together with their regulatory signals, have to be identified ; repetitive sequences have to be looked for ; secondary and tertiary structures of tRNA are to be correctly predicted. These operations, and many others, are coded as executable methods which are applied by the biologists on the sequences they are interested in. The executions of these methods produce a lot of results which can be encoded as instances of pre-defined classes. Defining these classes, i.e. specifying a knowledge base conceptual schema, is essentially a modelling activity. Building such a knowledge base is indeed equivalent to providing a model of the entities involved in the organisation and the functions of genomes.

In the framework of a tight collaboration between our computer science research group and the Laboratoire de Biométrie, Génétique et Biologie des Populations in Lyon, two knowledge bases have been developed. The first one, ColiGene [11], describes the regulation mechanism of gene expression in the *E. coli* bacterial genome. The second one, MultiMap, allows to describe and to manipulate mammalian genomic maps at the cytogenetic, genetic and physical levels.

The design of these two knowledge bases has led to the identification of three types of knowledge to be represented : *descriptive* knowledge on the biological entities involved, *methodological* knowledge on the ways to select and link up methods for a given task, and textual knowledge as *annotations* on the various objects. A fourth type of knowledge, which has not been introduced in these two knowledge bases, concerns the modelling of dynamic phenomena, such as the dynamics of gene inhibition or activation. These four types of knowledge are illustrated in the following four sections. A complete prototypical knowledge base management system has been designed and implemented along these ideas. It has been used, in various stages of achievement, for the development of ColiGene and MultiMap, and is now being used for the implementation of a complete co-operative computer system for genome sequence analysis.

2.1. Descriptive knowledge

Descriptive knowledge is knowledge on the various entities involved in the structure and the function of the genomes : genes, signals, repetitive sequences, etc. These entities are described as object classes and organised in hierarchies. For instance, figure 6 presents the main concepts and their hierarchies in the ColiGene knowledge base.

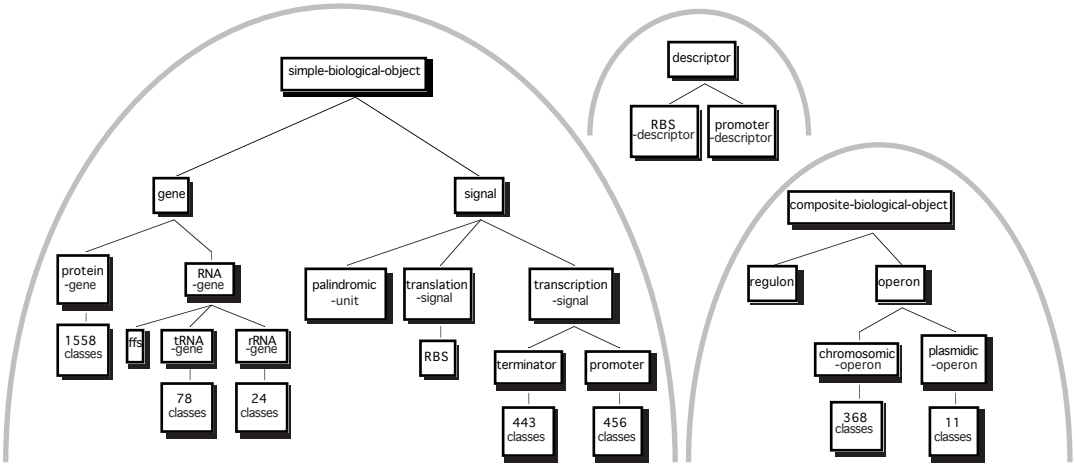

Figure 6 — Hierarchical structure of the knowledge base ColiGene. Individual subclasses and instances corresponding to individual structures are derived from the classes of upper levels. In such a way, instances take into consideration the allelic polymorphism. Three biological concepts are modelled in the base: the **`simple-biological-object`** concept gathers elementary information units like protein and tRNA genes and the signals involved in their regulation (promoters, operators, terminators, ...) ; the **`composite-biological-object`** concept contains complex regulatory systems like operons and regulons (this enables the linking of elementary objects which are completely independent in sequence databases) ; the **`descriptor`** concept describes DNA features corresponding to specific regulatory signals.

But the complexity of the biological entities involved makes a single hierarchy for the whole base very limiting. Genomic structures could be represented according to two perspectives, at least : a functional one where the hierarchy is based on the functions of the biological objects (this is the only perspective in the present state of ColiGene), and an evolutionary one describing the repetitive character of a sequence (i.e. whether a sequence is present in one or multiple copies in the genome and what kind of repetitive structure it contains) (figure 7).

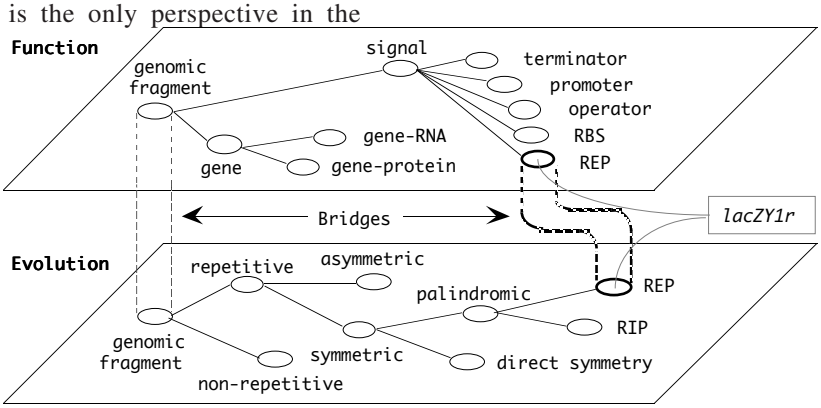

Figure 7 — Possible organisation of a part of ColiGene conceptual scheme. Upper levels of both hierarchies are defined here. A genomic fragment may be considered from the functional point of view as well as from the evolutionary one. The classification of the **`lacZY1r`** instance in the functional perspective leads to attach it to the **REP** class ; then, by using the bridge, the instance is attached to the **REP** class in the evolutionary perspective thus acquiring more information on the existing attributes of this class.

Thus, an extension of the classical object-based knowledge model consists in allowing the designer to describe different class/subclass structures according to the viewpoint retained [12]. On each perspective, a specific hierarchy of classes can be developed. An instance belongs to one class in each hierarchy and can be viewed along the desired hierarchies by showing the relevant attributes. A user can choose to look at a knowledge base along specific perspectives and dismiss information he is not interested in.

The classification process can take place on a single viewpoint or on a set of viewpoints. Using links between viewpoints, the result of classification on a viewpoint can be propagated to other viewpoints using bridges.

Figure 8 illustrates the navigation facilities in an example of a sample session under the knowledge base MultiMap which integrates different levels of mapping data with graphical tools allowing to switch between these levels. More precisely, MultiMap is dedicated to comparative mapping studies between man and mouse. It describes all known chromosomes and *loci* for both organisms (available in GENATLAS and EMG databanks) and allows the location study using cytogenetic, genetic and physical maps, i.e. three different levels of description. Links between the biological objects in the base and the different maps have been carefully modelled. When analysing a human genome, the related cytogenetic maps may thus be accessed, and in turn, from a cytogenetic map, any of its *loci* may be accessed. A special effort has been made to develop graphic browsing and manipulation facilities, because of the amount of objects (over 1000 *loci* for each of the two organisms).

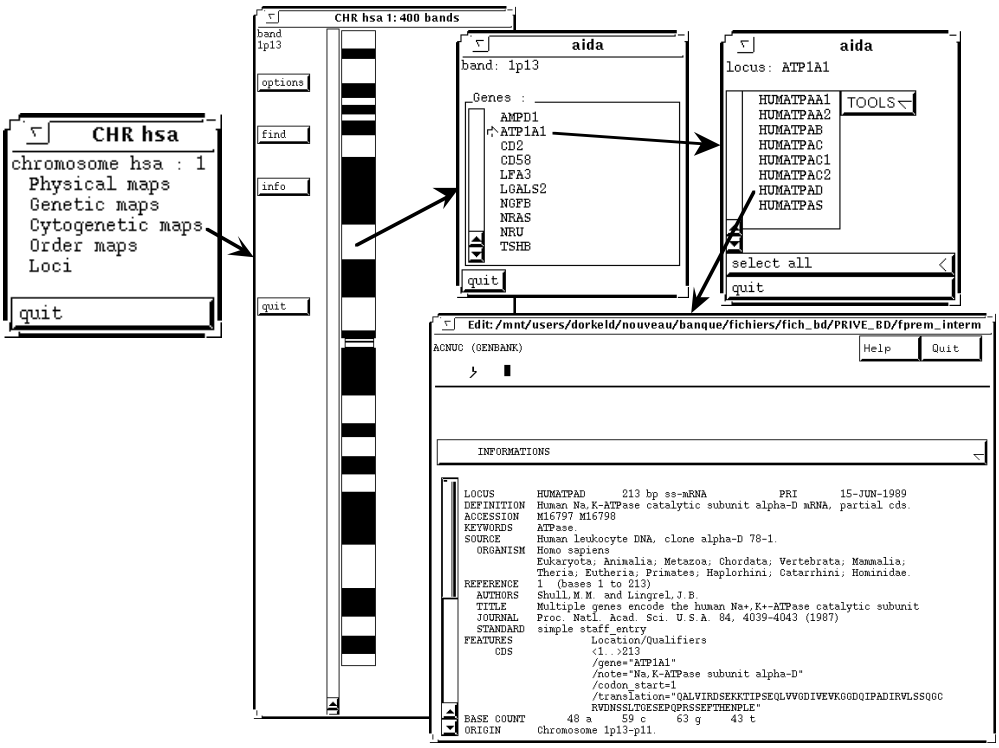

Figure 8 — Navigation facilities in the knowledge base MultiMap. Several windows obtained from successive user selections are represented here. The first one (on the left) shows the different levels of description of a human chromosome (n°1). In the second one, the cytogenetic chromosome bands obtained with G staining method are visualised. From one selected cytogenetic band, the list of its *loci* may be accessed and then, the list of the GenBank entries that describe a particular *locus* (in this example the **ATP1A1** gene has been selected). Given a GenBank entry, one can also obtain the information stored in the data bank (**HUMATPAD** in this example).

In order to add content search, an object matching mechanism has been implemented as a first step solution. Queries are built via a graphical editor, making the definition of the set of criteria rather easy. This query mechanism allows the retrieving of instances which satisfy the set of criteria. "What are the protein genes located between minute 0 and 5, in *E. coli* chromosome?" is a typical example of content search, in ColiGene. Queries may be saved and modified, if needed. The resulting sets of instances may be saved too : they may, in turn, be used as input in order to refine a query.

2.2. Methods and tasks

Methods are software modules which implement numerical, symbolic or graphic display algorithms. Domain-dependent methods, e.g. in molecular biology, give rise to various difficulties : their number and/or complexity grow regularly, they have been written in programming languages independently from the knowledge base design, they are available in some repository(ies) elsewhere, etc. In order to make them easier to use, our solution consists in the definition of problem solving strategies or *tasks*, i.e. the identification of subproblems and their proper ordering on the one hand, and in the selection of one method among a set of convenient ones for each (sub-)problem, on the other hand. Tasks as well as methods are thus described explicitly in the knowledge base as classes [13].

Most object-oriented languages do not allow methods to be defined outside a class (some languages, as CLOS, allow it, but with restrictions), because of the so-called "encapsulation" principle. Method selection according to multiple input arguments is therefore not possible. Object-oriented programming languages are thus not adapted to the description of methodological knowledge.

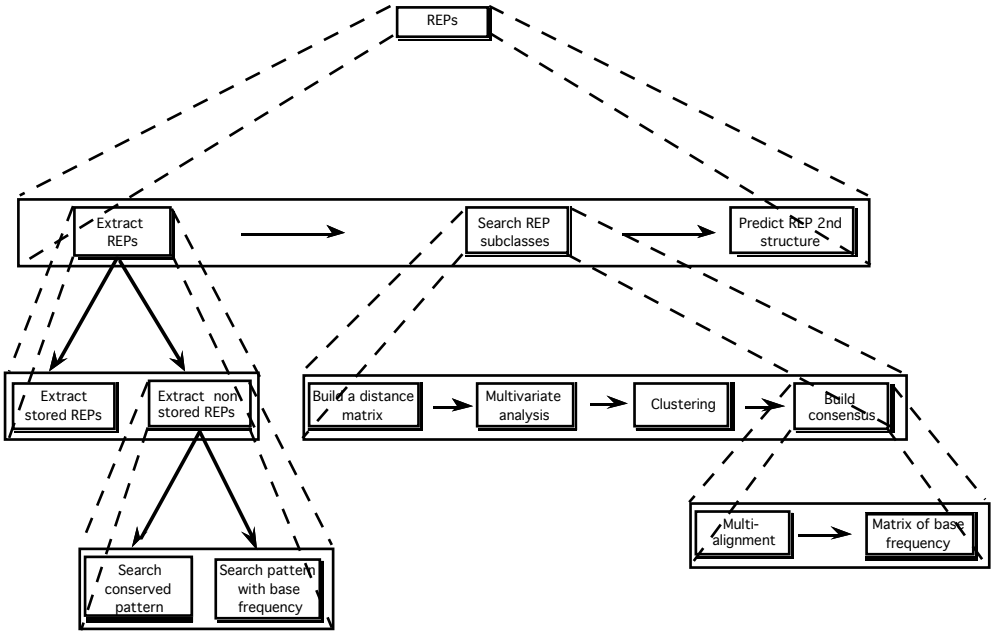

Figure 9 — Task decomposition hierarchy for Repetitive Extragenic Palindromic (REP) units identification. REPs are repetitive structures characterised by their complex secondary structures, which make them able to prevent mRNA degradation. To study REPs, a sequence of three sub-tasks has to be executed. First the REPs have to be extracted from the data bases. This sub-task is a choice sub-task: the extraction of REPs is different whether it concerns stored or non-stored REPs. In the case of the extraction of non-stored REPs, another choice is made : searching for a conserved pattern or searching a pattern with base frequency. When the complex task **`extract REPs`** is done, the REPs subclasses have to be identified. The complex task **`Search REP subclasses`** is a sequential task : first each REP sequence is aligned with the others, and the alignment score is used to build a distance matrix ; then multivariate analysis can be made followed by a clustering of the REPs. Obviously, the complex tasks **`Multivariate analysis`** and **`Clustering`** are tasks to be specialised, and a classification process must be done again. For each group defined in the clustering task, a consensus matrix has to be constructed : the task **`Build consensus`** is in turn a sequential task (multiple alignment in a way to establish a "consensus" sequence). Finally, when the complex task **`Search REPs subclasses`** is achieved, secondary structure prediction could be under study.

In our approach, tasks, as well as methods, are described as classes with input and output attributes. The set of constraints attached to input or output attributes describes respectively pre- and post-conditions. A task provides additional facilities : to each complex task, a decomposition into sub-tasks is attached, unless the task is elementary ; in that case, a set of methods is directly attached to the task. A complex task can be either a selection among sub-tasks (e.g. `extract-REPs` in figure 9), or a sequence of sub-tasks (e.g. `Search REP subclasses`).

Since tasks are described as classes, it seems very natural to organise them into hierarchies. A task class may be a sub-class of an other task class (e.g. the task class `REPs` is a sub-class of the more general class `Palindromic-units` in figure 10).

The decomposition of a task into sub-tasks can then be different according to the sub-class retained, if required. Therefore, each time a task has to be decomposed, a classification process is executed in order to select, among the possible sub-classes, the class which matches the current context and which will provide the adequate decomposition. Thus, the overall problem solving process consists in mixing classification stages, where a task is characterised according to its input, and planning stages, during which the task is split up in sub-tasks according to the decomposition attached to the sub-class chosen during the classification stage.

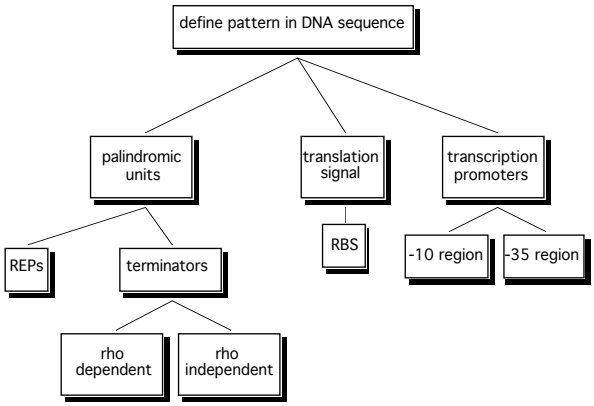

Figure 10 — Let us consider a high level task for pattern analysis in DNA sequences. The upper level task describes a general objective (**`Define pattern in DNA sequences`**). No specific solving strategy is associated with this task description: then using the classification process, this kind of task has to be decomposed into sub-tasks according to the control flow. At a lower abstraction level, the problem defined by a task is sufficiently precise so that a decomposition scheme can be associated to the task description. It is the case of the tasks **`rho-independent`**, **`rho-dependent`**, **`REPs`**, **`RBS`**, **`-10 region`** and **`-35 region`** .

Such a model interactively supports the user in choosing the methods necessary to solve his problem, and in chaining them in a proper way. It allows automatic solving as well as user-solving, if required (e.g. no available or known solving strategy, analysis of graphics, etc.) [14].

2.3. Dynamic models

Horizontal links between attributes in objects can be used to represent interactions between variables in a dynamic model. Obviously, some information must be added onto the links to specify the interactions. When minimal information is provided, only qualitative simulation is possible. Such an approach allows simulation at an intermediate level of precision, and makes it possible without requiring experimental data as for differential equations..Bobrow [15] identified different qualitative reasoning tasks ; among them are simulation, envisionment (determining all possible behaviours), diagnosis and verification. Even if this approach has limited utility for large models [16], mainly because of combinatorics, it tends to allow the modelling of phenomena for which the numerical accuracy is not required, or for which behaviour laws are not completely known.

Qualitative simulation has been already applied in biology. MOLGEN.[17] supports the qualitative simulation of the *trp*-operon system. Weld [18] has implemented a system to perform causal simulation in the molecular genetics domain : it allows the description of discrete processes to model possible enzymatic behaviours for a set of molecule types and it attempts to predict the behaviour of the system. A system for the design of metabolic pathways.[19] relies on the qualitative representation of the Boerhinger metabolic map. From constraints on inputs and reactions, it infers the complete set of all the possible pathways through the map.

We plan to extend our object-based model to support the description, selection and simulation of qualitative models. We will benefit from our experience on knowledge-based systems for dynamic modelling in population dynamics [20].

2.4. Hyper-textual annotations

While developing a knowledge base, many modelling choices have to be made. Every entity of the real world may indeed be represented by several distinct object descriptions. The decisions are made by the designer according to the expected uses of the knowledge base and to his own perception. The explicit reasons which support these decisions should be accessible to the users and therefore stored in the knowledge base itself. In order to avoid indefinite loops, these justifications cannot be expressed with the knowledge model, and natural language is certainly the most innate and easy to understand medium.

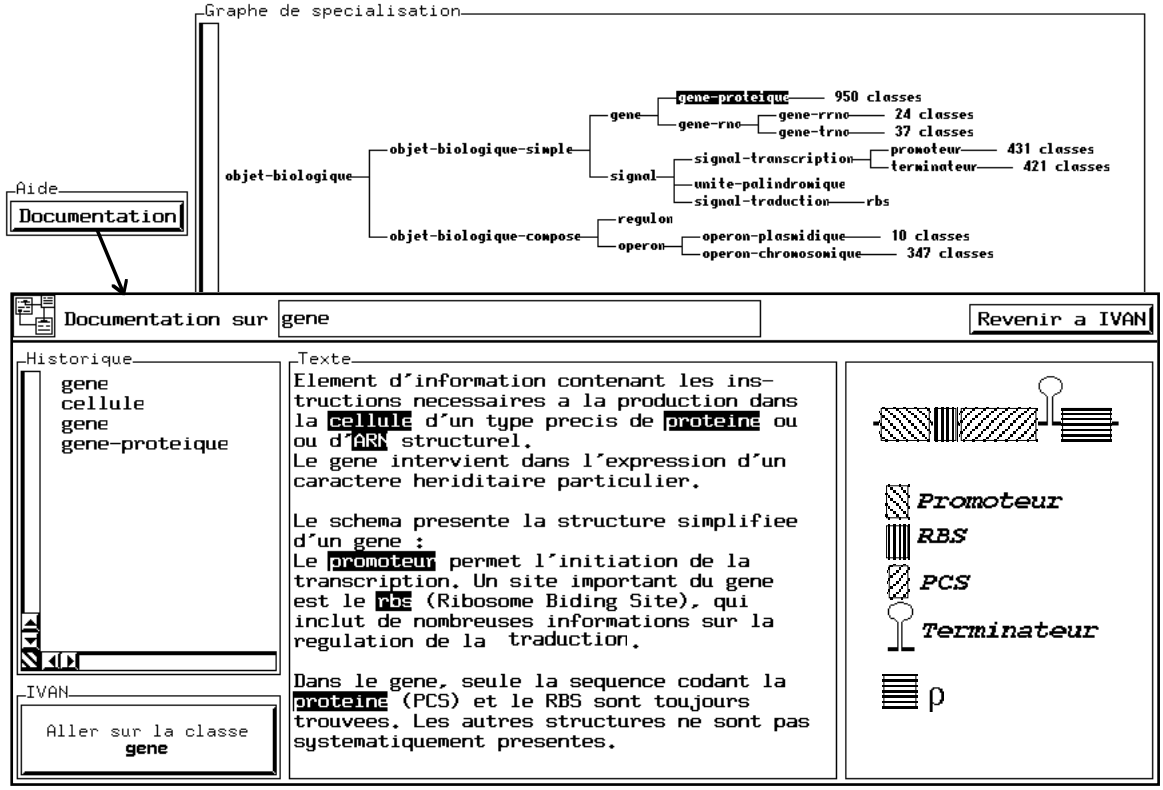

Figure 11 — The navigational interface using hypertext. The user has selected a class in the inheritance graph (upper window) and has consulted the associated node of the hypertext (lower window). Some of the words of the text are links to other nodes. From an hypertext node, it is possible to come back to the entity it is attached to. The example is drawn from the ColiGene knowledge base, in which only the highest levels of the specialisation graph have been annotated, essentially for pedagogical purposes.

Thus, pieces of text should be attached to the various object classes and to their components, such as attributes essentially. Navigating from a class to another, a user can read the associated text. But the text itself can be seen as a knowledge base, unable of course to support any inference mechanism. Arranging the various pieces of text into an hypertext [21, 22] allows a navigation process to take place between its nodes. Moreover, the user can choose to navigate among the objects or among the hypertext nodes alternatively.

We have implemented the connection between objects and hypertext in our system. Every new piece of text is turned into an hypertext node automatically: the text is searched for words which match class or attribute names. The node itself is named after the class or the attribute to which it is attached. Pictures can also be placed in the node and displayed together with the text at the user's request. Most of the classes and the attributes of the upper levels of the class hierarchy of ColiGene have been commented in that way (figure 11).

What is the exact status of text in such knowledge bases? Text can be seen as simple comments as presented before. These comments can sustain pedagogical purposes: text attached to the highest levels in the class hierarchy can present fundamental notions of the scientific field the knowledge base is concerned with. Text can also be presented as the knowledge format which precedes formalisation. This is the view supported for instance in [23] where hypertext is used for knowledge elicitation and acquisition. Even after textual knowledge has been formalised as inter-related objects, it can be conserved to play the role of comments. But text can also be seen as the most advanced form of compiled knowledge. Under this interpretation, as parts of scientific knowledge bases are progressively becoming consensual, they could be rewritten as texts and figures only, hopefully arranged in hypertexts: (hyper-) textbooks are perhaps the ultimate state in the knowledge life-cycle.

3. CONCURRENT AND INCREMENTAL BUILDING OF KNOWLEDGE BASES

Knowledge bases have long been considered only as essential parts of the so-called "expert systems". Following this interpretation, knowledge bases are essentially used for problem solving, while the knowledge bases presented above are designed with a modelling objective in mind. They do help problem solving, through the use of inference mechanisms such as classification or procedural attachment, but their first interest is to be models which can be understood by human readers. The next step is therefore to use these knowledge bases as a support for sharing knowledge among a community of researchers and to help the emergence of common, or "consensual", scientific knowledge through a concurrent process of model building. The resulting consensual knowledge bases can be seen as "ontologies" [24]. But in our context, and in agreement with the ideas of Stefik [25], producing and maintaining these ontologies constitute ends in themselves.

3.1. Consensual and personal knowledge bases

In order to present the general scheme of concurrent scientific knowledge base building, the particular case of genome sequencing projects will be discussed. Large-scale sequencing projects involve several laboratories which are geographically distributed. The yeast European consortium is a very good example. Thirty-five laboratories in different member countries of the EEC have worked together towards the determination of the complete sequence of the *S. cerevisiae* chromosome III.

But getting the sequence, that is the "text", is one thing, interpreting the text is another. The discovery and the confirmation of objects and their roles in the function and the structure of the genome rely on the exchange of information between all the groups. Up to now, most of these exchanges take place either via informal ways such as personal communications, either through the submission of papers to scientific specialised journals.

Figure 12 presents an alternative to these interaction schemes. Each contributor, single user or group of users, maintain a "personal" knowledge base. When pieces of knowledge in a personal knowledge base appear to be mature, they may be sent to the "consensual" knowledge base, in order to be confronted with its contents. As a result of this confrontation, a report is sent back which explains how the submission has been evaluated. The submitter has then the possibility to react to this report, either by modifying its submission and submitting again, or by initiating discussions with his colleagues who are also contributing to the same consensual knowledge base.

3.2. Basic operations on knowledge pieces

Such an interaction scheme assumes that it is possible to compare two pieces of knowledge written as class or instance descriptions in an object-based knowledge model. Several comparison operations can be implemented.

The first one is widely used in the context of the so-called "terminological languages" or "descriptive logics" [26, 27] : the operation of subsumption which determines if one description subsumes another one. It is now well known that the complexity of the operation is very sensitive to the level of expressiveness of the descriptive language. Subsumption allows to decide if a piece of knowledge is a special case of another one, or if it is a generalisation of another one. It is also the basic operation which helps inserting a new class into an existing hierarchy of classes and sub-classes (see for instance [28]).

A second kind of operation attempts to decide if a description contradicts another one, i.e. the intersection of their extension (potential set of instances) is empty. This operation applies only when descriptions are supposed to describe the same concept.

A third class of operations is quite large, since it includes the operations which rely on some distance to search for similar descriptions. A set of similar descriptions may then be the input to automatic generalisation algorithms.

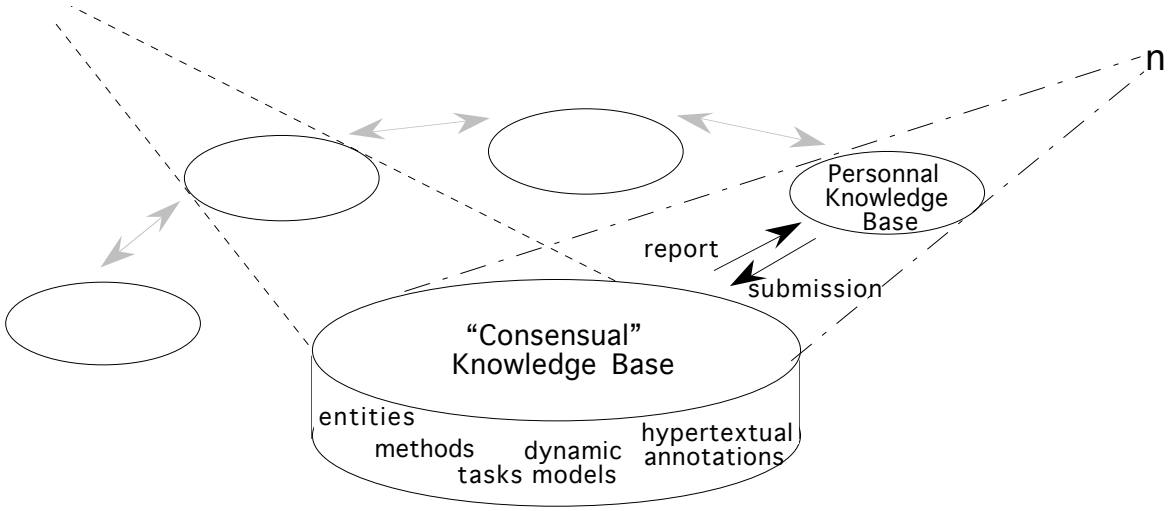

Figure 12 — The metaphor of paper submission is the basis for computer-based information exchange and consensual knowledge formation.

3.3. Interaction protocols

As explained before, the result of the confrontation between pieces of knowledge and the contents of the consensual knowledge bases is the production of a report which explains to the submitter how his submission relates to the consensual knowledge.

According to the report several actions are possible. If the submission does not contradict the knowledge base it may be accepted and inserted at the proper location, following the subsumption relationship. It will be tagged with the name of the submitter and is obviously subject to further revisions or deletion.

If the submission contradicts the knowledge base, the report explains what are the sources of contradiction. The submitter may then decide to modify his description and try another submission, and so on. But, he might also initiate a discussion with his colleagues, confronting his description with their own descriptions they maintain in their personal knowledge bases. The general idea of this scheme is that repeated confrontation of knowledge leads to consensus.

CONCLUSION

Object-based knowledge models allow the development of large knowledge bases as exemplified in the domain of molecular genetics. They provide excellent descriptive and structuring capabilities and support natural navigation processes and classification mechanisms. Classification allows to characterise incomplete objects, but also to correctly insert newly defined classes into existing specialisation hierarchies, thus making the knowledge base design easier.

The object-based model is used to model the tasks to be solved, the entities these tasks and methods work on, and the relationships between these entities as qualitative equations. Hypertext nodes can be attached to each object and to its components, so that a navigation process can take place among the objects and these nodes.

ColiGene, on the regulation of gene expression in *E. coli*, and MultiMap, on the different levels of mammalian genome mapping, are two operational examples of knowledge bases which rely on the object-based model to describe complex knowledge in molecular genetics.

The ability to compare two homologous knowledge description units and to decide whether they are complementary or contradictory leads to a possible scheme of concurrent and incremental construction of scientific knowledge bases. The traditional structure of scientific exchanges and scientific knowledge formation would be completely transformed.

ACKNOWLEDGEMENTS

This research is being supported by French Ministère de la Recherche et de l'Espace (grant 92.H.0904), by CNRS (Centre National de la Recherche Scientifique, GdR "Informatique et génomes") and Région Rhône-Alpes.

REFERENCES

[1] Von Heije G., "Sequence analysis in molecular biology : treasure trove or trivial pursuit", San Diego, CA, Academic Press, Inc., 1987

[2] Waterman M.S., "Mathematical methods for DNA sequences", Boca Raton, FL, CRC Press, Inc., 1989

[3] Stefik M., Bobrow D.G., "Object-Oriented Programming : Themes and Variations", The AI Magazine, 6(4), pp. 40-62, 1986

[4] Overton C.G., K. Koile, J.A. Pastor, "GeneSys : a knowledge management system for molecular biology", in Bell G. I., T. G. Marr, (eds), "Computers and DNA", Addison-Wesley, 1990

[5] Tanaka, H., "Integrated System for Protein Information Processing", Proc. of the International Conference on Fifth Generation Computer Systems, 1992, pp. 321-329

[6] Yasukawa, H., "Objects, Properties and Modules in Quixote", Proc. of the International Conference on Fifth Generation Computer Systems, 1992, pp. 307-319

[7] Yoshida, K., "Toward a Human Genome Encyclopedia", Proc. of the International Conference on Fifth Generation Computer Systems, 1992, pp. 307-320

[8] Minsky M., "A Framework for Representing Knowledge", in P.H. Winston (ed.), "The Psychology of Computer Vision", pp. 211-281, McGrawHill 1975

[9] Fikes R., Kehler T., "The Role of Frame-based Representation in Reasoning", Communications of the ACM, Vol. 28, n° 9, pp. 904-920, September 1985

[10] Oliver S. *et al.*, "The complete DNA sequence of yeast chromosome III", Nature, vol. 357, May 1992, pp. 38-46

[11] Perrière G., Gautier C., "ColiGene : object-centered representation for the study of *E. coli* gene expressivity by sequence analysis", Biochimie, Vol. 75, n°5, 1993, pp. 415-422

[12] Mariño O., Rechenmann F., Uvietta P., "Multiple Perspectives and Classification Mechanism in Object-Oriented Representation", Proc. 9th ECAI (August 8-10, 1990, Stockholm), Pitman Publishing, London, 1990

[13] Rechenmann F., Rousseau B., "A development shell for knowledge-based systems in scientific computing", in Houstis E.N., Rice J.R. (eds), "Expert Systems for Numerical Computing", Elsevier Science Publishers, Amsterdam, 1992

[14] Chevenet F., Jean-Marie F., Willamowski J., "A development shell for cooperative problem solving environments", Third International Conference on Expert Systems for Numerical Computing, Purdue (IN, USA), May 1993, Computer Sciences Department, Purdue University, Technical Report CSD-TR-93-028, CAPO Report CER-93-14, pp. 1-7

[15] Bobrow, D.G., "Qualitative Reasoning about Physical Systems : An Introduction", Artificial Intelligence, 24, pp. 1-5, 1984

[16] Forbus, K.D., Falkenhainer, B., "Self-Explanatory Simulations : Scaling up to large models", Proc. of the 10th National Conference on Artificial Intelligence AAAI-92, July 12-16, San Jose, Ca, pp. 685 -690, 1992

[17] Friedland P., Kedes L., "Discovering the secrets of DNA", Computer, November 1985, pp. 49-69, 1985

[18] Weld, D.S., "The Use of Aggregation in Causal Simulation", Artificial Intelligence, Vol. 30, pp. 1-34, 1986

[19] Mavrovouniotis, M., *et al.*, "Computer Aided Synthesis of Biochemical Pathways", Biotechnological Bioengineering, Vol. 36, pp. 1119-1132, 1990

[20] Rechenmann F., Rousseau B., "EDORA : an object-centered knowledge-based approach to dynamic modelling", in "Artificial Intelligence and Cognitive Sciences", Proceedings in Non-Linear Science, Demongeot J., Hervé T., Rialle V., Roche C. (eds), Manchester University Press, 1988

[21] Conklin J., "Hypertext : an introduction and survey", IEEE Computer, Vol. 20, n° 9, 1987

[22] Nielsen J., "Hypertext and Hypermedia", Academic Press, 1990

[23] Gaines B.R., Linster M., "Integrating a Knowledge Acquisition Tool, an Expert System Shell and a Hypermedia System", in International Journal of Expert Systems, Vol. 3, n° 2, pp.105-129, 1990

[24] Neches R. *et al.*, "Enabling Technology for Knowledge Sharing", AI Magazine, Vol. 12, n° 3, pp. 36-56, 1991

[25] Stefik M., "The Next Knowledge Medium", AI Magazine, Vol.. 7, n° 1, pp. 34-46, 1986

[26] MacGregor R., Burstein M.H., "Using a Description Classifier to Enhance Knowledge Representation", IEEE Expert, June 1991, pp. 41 - 46

[27] Patel-Schneider P.F., Owsnicki-Klewe B., Kobsa A., Guarino N., MacGregor R., Mark W.S., McGuinness D.L., Nebel B., Schmiedel A., Yen J., "Term Subsumption Languages in Knowledge Representation", AI Magazine, Vol. 11, n° 2, summer 1990, pp. 16-63

[28] Capponi C., "Interactive Class Classification", Proc. *IFCS'93*, Paris, France, September 1993