



## RAPPORT DE RECHERCHE

# Introducing Graph-based Reasoning into a Knowledge Management Tool: an Industrial Case Study

Olivier Carloni, Michel Leclère, Marie-Laure Mugnier

24/11/2005

R.R.LIRMM 06-004

# Introducing Graph-based Reasoning into a Knowledge Management Tool: an Industrial Case Study

Olivier Carloni<sup>1,2</sup>, Michel Leclère<sup>1</sup>, Marie-Laure Mugnier<sup>1</sup>

<sup>1</sup> LIRMM, CNRS - Université Montpellier 2  
161 rue Ada, F-34392 Montpellier cedex 5 - France  
{carloni, leclere, mugnier}@lirmm.fr

<sup>2</sup> Mondeca,  
3 cité Nollez, F-75018 Paris - France  
<http://www.mondeca.com>

**Abstract.** This paper is devoted to an industrial case study focused on the issue of how to enhance ITM, a knowledge management tool, with reasoning capabilities, primarily by introducing a semantic query mechanism. ITM knowledge representation language is based on topic maps. We show that these topic maps (and especially those describing the domain ontology and the annotation base) can be naturally mapped to the *SG* family, a sublanguage of conceptual graphs. As this mapping is reversible, ITM can be equipped with a graph-based query language equivalent to conjunctive queries and it can be enriched with inference rules.

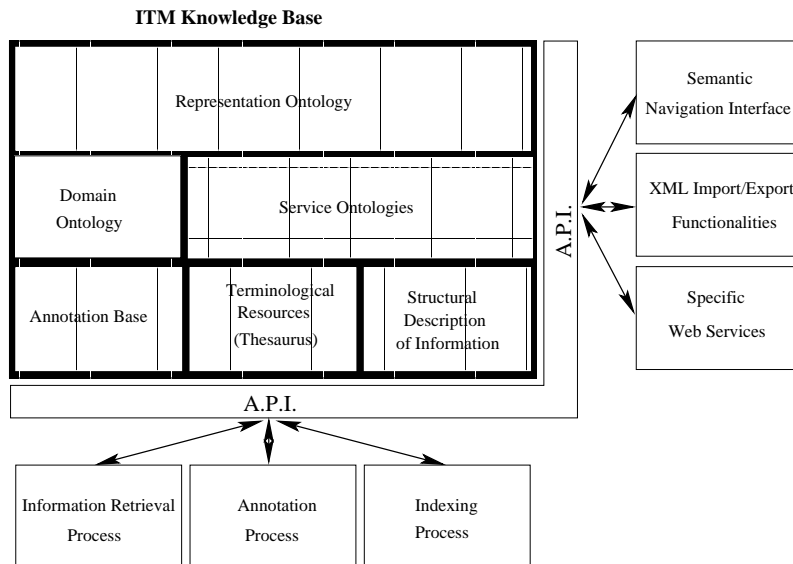
## 1 Introduction

In the last decade, there has been a massive flow of knowledge into organizations to meet with requirements of (1) capitalization of the knowledge and know-how of their members in order to build up a corporate memory (i.e. a disembodied representation of their expertise) and (2) implementation of their information systems at the “knowledge level”, i.e. exploiting semantics of exchanged information. This evolution from information systems to knowledge systems has given rise to true knowledge engineering, which aims at eliciting knowledge (e.g. the semantics of exchanged information content). This elicitation is performed by way of a knowledge representation language and is controlled by ontologies (domain ontologies for the conceptual vocabulary and representation ontologies for the language constructs) which are semantic referentials that delineate the meaning of symbols used by these languages. This issue arises again within the framework of the semantic web, which endeavors to describe the content of web resources in order to facilitate their access and use.

Mondeca is a software publisher that is developing ITM (Intelligent Topic Manager), a knowledge management tool based on these principles. The core of this software is a three-level knowledge base (cf. fig. 1):

- the highest level is a meta-model of all ITM knowledge bases; it consists of a representation ontology reflexively specifying the semantics of all representations used by ITM;
- the intermediate level is composed of models that specify the vocabulary used to describe customer data managed by ITM; it generally comprises a domain ontology describing the conceptual vocabulary used to annotate the content of the customer’s resources, some (representation) ontologies defining primitives related to specific data structures enabling ITM services (primitives for representing the thesaurus or the logical organization of documents);
- the lowest level contains the instances, which are clustered into workspaces (each workspace being “controlled” by an ontology of the intermediate level): annotations describing the content of information managed by ITM (data, documents), terminological resources (thesaurus) used to index this information, description of the logical organization of documents, etc.

ITM provides many services based on its knowledge base: indexing of documents from the thesaurus, creation of annotations controlled by the domain ontology, semi-automatic building of document annotations, semantic navigation through the annotation base, searching the annotation base by a querying mechanism.



**Fig. 1.** Workspaces and modules in ITM

The representation language in ITM is based on the Topic Map (TM) paradigm in which knowledge is described by *topics*, representing entities of the modeled domain, and *associations* connecting topics and identifying the *role*

played by each topic in the association [?]. Topics can be identified by *names* and characterized by *occurrences* (kinds of attribute-value pairs). Two specific binary associations are standardized for any TM: *class-instance* association and *superclass-subclass* association [?]. TMs can be formalized as labeled graphs (or hypergraphs) [?] where the vertices come from topics and associations and an edge between a topic and an association indicates the role played by the topic in this association (cf. section 2).

LIRMM and Mondeca are collaborating in a research project which aims at introducing reasoning on the knowledge represented in ITM knowledge bases. Formal semantics of representations used by ITM have thus to be defined in order to specify the desired reasonings. Despite the lack of formal semantics within the TM paradigm, its closeness to the conceptual graph model [?] and especially to the  $\mathcal{SG}$ -family [?] – a TM is naturally transformed into a simple conceptual graph (SG) – has led us to map ITM representations into this family. By assimilating TMs to SGs it is possible: to provide ITM with formal semantics since the  $\mathcal{SG}$  family is logically based; to incorporate inference rules in ITM bases since the  $\mathcal{SG}$  family manages this kind of knowledge (as well as constraints, type definitions, etc); to use the reasoning schemes of the  $\mathcal{SG}$  family, which are graph-based (thus operate directly on the knowledge defined by the user) while being sound and complete with respect to the predicate logic; to benefit from efficient algorithms developed for these graph operations and based on combinatorial techniques; and finally to tap the GPL library CoGITaNT, which is dedicated to developing applications based on conceptual graphs and implements the  $\mathcal{SG}$  family [?].

In this paper, we present mapping of ITM topic maps to SGs and the query language - as well as the answering mechanism - which is obtained via this mapping, thus equipping ITM with a formally based reasoning module. We then show that these initial results are easily extended to take inference rules into account, thus enriching the reasoning capabilities.

The sequel of this paper is organized as follows. In section 2 the TM language and the three-level architecture of ITM are presented. Section 3 introduces SGs. Section 4 is devoted to the mapping, the kind of reasoning it enables, and the gains for ITM. Finally, Section 5 outlines future extensions.

## 2 ITM - A knowledge management tool

ITM knowledge is represented by Topic Maps (TMs). A TM is a network of topics linked by associations. Topics and associations are typed elements. Each topic involved in an association plays a role in this association. Formally, we define a TM as a bipartite labeled graph  $tm = (T, A, E, type, name)$  where  $T$  is the set of topic nodes,  $A$  is the set of association nodes and  $E \subseteq A \times T$  is the set of edges. *type* and *name* are labeling functions from  $T \cup A \cup E$  into  $L_T$ , where  $L_T$  is the set of labels identifying topics. *type* assigns to each node and edge a label from  $L_T$ , which denotes the class for a topic, the type for an association

and the role name for an edge. The partial function *name* identifies some topics with a name. See figure 2.

Each topic can be further described by a set of occurrences expressing relevant information about the topic. Each occurrence is composed of a value, a data type (called physical type in ITM) and a topic representing the type of the occurrence (called logical type in ITM). We denote by *occ* the mapping from  $T$  into  $2^{V \times D \times L_T}$  which assigns to a topic its occurrence set, where  $D$  is the data type set and  $V$  is the set of values for types in  $D$ . Finally, we call *ref* the mapping which assigns to a TM the subset of  $L_T$  labels returned by the functions *type* and *occ*. A TM *tm* is said to be self-content if each label  $l$  from *ref(tm)* is assigned to a topic by the *name* function. An ITM knowledge base is a self-content TM structured in three levels, namely the meta level TM, the model level TM and the instance level TM. These three levels fulfill the following constraints:

- the meta level TM is self-content;
- the set of labels obtained by *ref* to the model level TM (resp. instance level TM) is included in the set of labels assigned by *name* on the meta level topics (resp. model level TM).
- all meta level topics and model level topics have a name.

The meta level TM defines modeling primitives. These primitives are interpreted by ITM and provide semantics of knowledge in all levels:

- an association of type **allowed association type** linking three topics  $t_c$ ,  $t_a$  and  $t_r$  specifies that every topic of class  $t_c$  is allowed to play a role  $t_r$  in an association of type  $t_a$ . For example, the model level in Figure 2 authorizes a topic of class **Company** to play the role **Acquiring** in an association of type **Acquisition** at instance level;
- an association of type **allowed occurrence type** linking  $t_c$  and  $t_o$  specifies that a topic of class  $t_c$  may have an occurrence with logical type  $t_o$ ;
- an association of type **has physical type** linking  $t_o$  and  $t_p$  specifies that all occurrences with logical type  $t_o$  have the physical type  $t_p$ . ITM uses the physical type to correctly interpret the value of the occurrence (as an integer, a date, a string or a pointer).
- the association of type **class-subclass** defines a partial order on topics; the **allowed association type** and **allowed occurrence type** associated with classes of topics are inherited according to this order.

### 3 Simple Conceptual Graphs

This section is devoted to an informal presentation of basic conceptual graphs. For further details the reader is referred to [?] or [?] (this latter paper studying the whole *SG* family).

A *simple conceptual graph (SG)* is a bipartite labeled graph: one class of nodes, called *concept* nodes, represents entities and the other, called *relations* nodes represents relationships between these entities or properties of them. E.g.

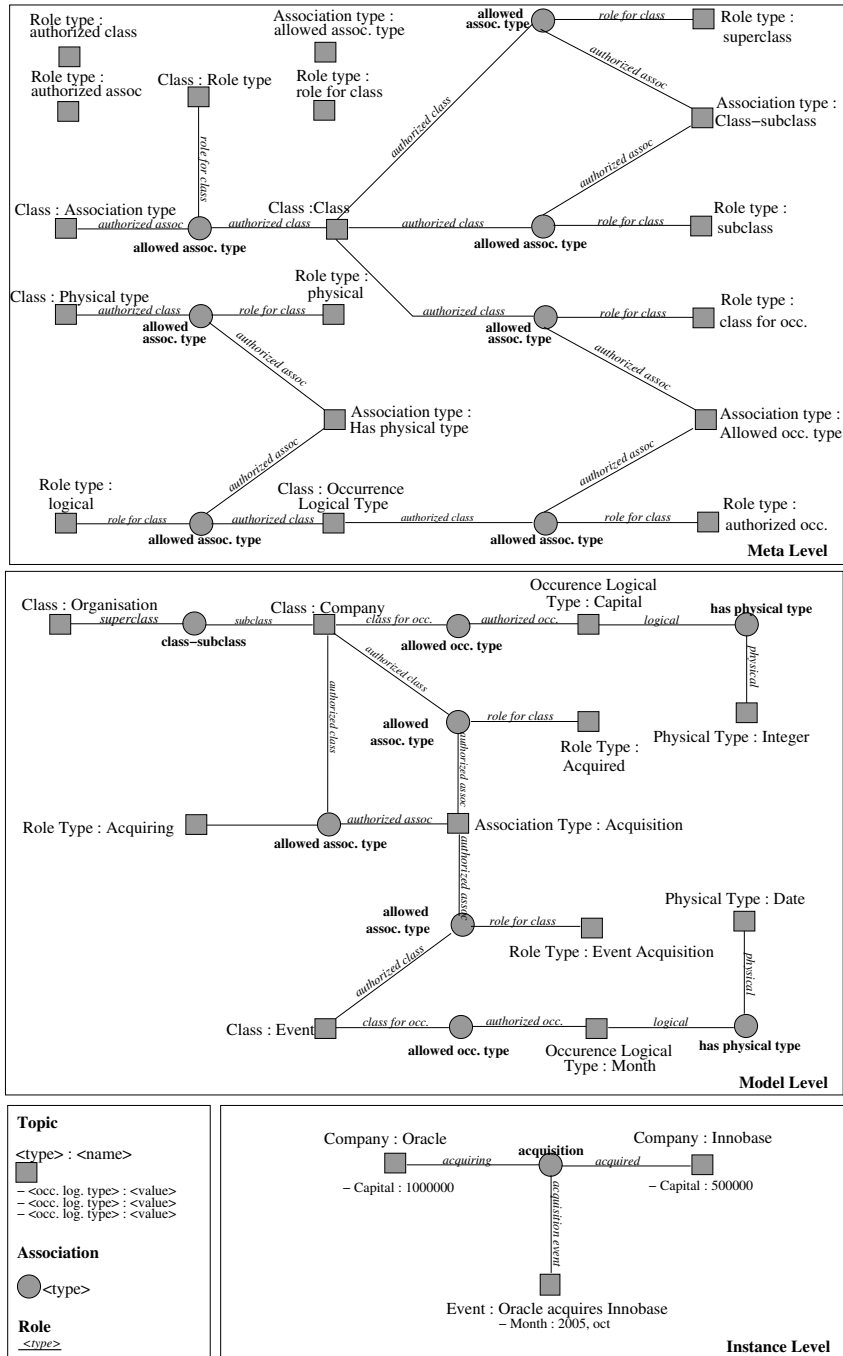
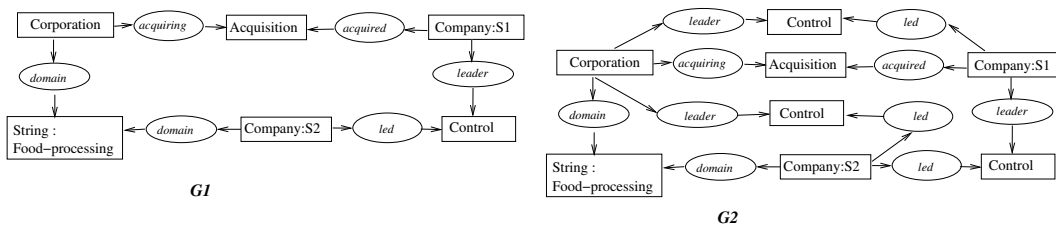


Fig. 2. A (partial) knowledge base with its meta, model and instance levels.



**Fig. 3.** Two SGs. As relations are all binary here the edge numbering is replaced by directed edges. The generic marker is not represented.

the graph  $G_1$  of Figure 3 can be seen as expressing the following knowledge: “A corporation specialized in food-processing acquires the company  $S_1$ ;  $S_1$  controls the company  $S_2$ , which is specialized in food-processing as well”. The node labels come from a vocabulary called a *support*, which can be more or less rich. The support considered here is a structure  $S = (T_C, T_R, I, \sigma)$ , where  $T_C$  is a set of concept types and  $T_R$  is a set of relations with any arity (the arity is the number of arguments of the relation).  $T_C$  and  $T_R$  are partially ordered. The partial order represents a specialization relation ( $t' \leq t$  is read as “ $t'$  is a specialization of  $t$ ”).  $I$  is a set of individual markers.  $\sigma$  is a mapping assigning to each relation a signature specifying its arity and the maximal type for each of its arguments. The support can be seen as a rudimentary ontology and SGs encode assertions called *facts*: they assert the existence of entities and relations among these entities.

In a SG a concept node is labeled by a couple  $t : m$  where  $t$  is a concept type and  $m$  is a marker. If the node represents an unspecified entity its marker is the generic marker, denoted by  $*$ , and the node is called a *generic* node, otherwise its marker is an element of  $I$ , and the node is called an *individual* node. E.g. in the SG  $G_1$  of Figure 3, the node [Company:S1] refers to “the” company  $S_1$ , while the node [Corporation:∗] refers to “a” corporation. A relation node is labeled by a relation  $r$  and, if  $n$  is the arity of  $r$ , it is incidental to  $n$  totally ordered edges. Classically, concept nodes are drawn as rectangles and relation nodes as ovals and the order on edges incidental to a  $n$ -ary relation node are numbered from 1 to  $n$ . A SG is denoted by  $G = (C_G, R_G, E_G, l_G)$  where  $C_G$  and  $R_G$  are respectively the concept and relation node sets,  $E_G$  is the set of edges and  $l_G$  is the mapping labeling nodes and edges.

The fundamental notion for comparing SGs is a mapping from a SG to another called a *projection*. In graph terms it is a graph homomorphism. Intuitively a projection from  $G$  to  $H$  proves that the knowledge represented by  $G$  is included in (or implied by) the knowledge represented by  $H$ . More specifically, a projection  $\pi$  from  $G$  to  $H$  is a mapping from  $C_G$  to  $C_H$  and from  $R_G$  to  $R_H$  which preserves edges (if there is an edge numbered  $i$  between  $r$  and  $c$  in  $G$  then there is an edge numbered  $i$  between  $\Pi(r)$  and  $\Pi(c)$  in  $H$ ) and may specialize labels. E.g. see Figure 3: there is a projection from  $G_1$  to  $G_2$  (which is particular since it is injective and does not change labels). See also the SG  $Q$  of Figure 5 (where ‘?’ has to be replaced by  $*$ ): there is a projection from  $Q$  to  $G_2$  (know-

ing that `Corporation`  $<$  `Company`) but not from  $Q$  to  $G_1$ . Conceptual graphs are provided with a semantics in first order logic that we do not present here due to space limitations. The fundamental result of *projection soundness and completeness* shows the equivalence between projection checking and deduction checking on the formulas assigned to SGs ([?] for the soundness and [?] for the completeness).

## 4 A conceptual graph inference engine for ITM

This section details the transformation from ITM to SGs and shows the benefits for ITM.

### 4.1 Transformation from ITM knowledge base to SGs

We denote by  $f$  the transformation from an ITM base to SGs. This transformation encodes the domain ontology into a support and the annotation base into a SG (or a set of SGs). It is illustrated by Figure 4.

**Transformation of a domain ontology into a support.** Let us recall that the domain ontology is a TM of the model level.

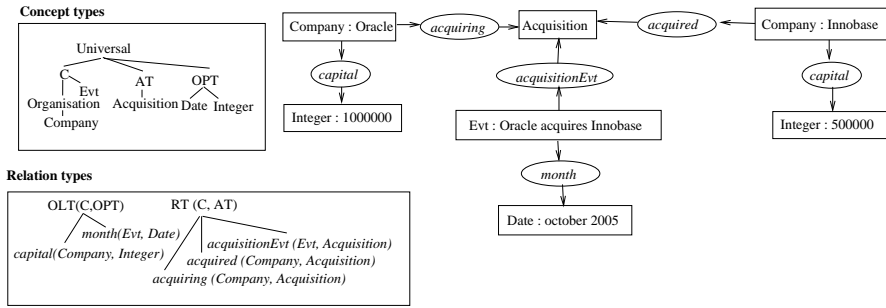
1. Three concept types  $C$  (for topic classes),  $AT$  (association types) and  $OPT$  (occurrence physical types) are created as subtypes of a universal type  $\top$ . To each topic  $t$  with `type class`, `association type` or `physical type`,  $f$  assigns a concept type with label  $name(t)$  subtype of  $C$ ,  $AT$  or  $OPT$  resp. The partial order on  $T_C$  is induced by the `class-subclass` association.
2. Two relations  $RT$  and  $OLT$ , with signature  $\sigma(RT) = (C, AT)$  and  $\sigma(OLT) = (C, OPT)$  resp., are created and represent in the support the supertypes of the role types and the occurrence logical types resp. Each topic  $t$  with type `role type` involved in a ternary association of type `allowed association type` connecting  $t$  to topics  $c$  and  $a$  becomes a relation with label  $name(t)$ , subrelation of  $RT$ ; its signature is  $\sigma(name(t)) = (f(c), f(a))$ . Recall that each topic  $t$  with type `occurrence logical type` is linked by an association with type `has physical type` to a topic  $opt$  and by an association with type `allowed occurrence type` to a topic  $c$ ;  $t$  becomes a relation, subrelation of  $OLT$ , with label  $name(t)$  and signature  $\sigma(name(t)) = (f(c), f(opt))$ .

**Transformation of an annotation base into a SG.** Let us recall that the annotation base is a TM of the instance level.

1. Each named topic  $t$  (i.e.  $name(t)$  is defined) becomes an individual concept node labeled by  $f(type(t)) : name(t)$ ; furthermore  $name(t)$  is inserted as an individual marker into  $I$ .
2. Each unnamed topic  $t$  (resp. association  $a$ ) becomes a generic concept node labeled by  $f(type(t)) : *$  (resp.  $f(type(a)) : *$ ).

3. Each edge  $e$  connecting an association  $a$  and a topic  $t$  becomes a relation node labeled by  $f(\text{type}(e))$ . Its two incident edges labeled 1 and 2 are resp. linked to  $f(t)$  and  $f(a)$ ;
4. Each occurrence  $o = (v, \text{opt}, \text{olt})$  in a topic  $t$  becomes a relation node labeled by  $f(\text{olt})$ . Its two incident edges labeled 1 and 2 are resp. linked to a concept node  $f(t)$  and to an individual concept node labeled  $f(\text{opt}) : v$  (this individual node is created only if it does not exist yet).

$f$  is a coding mapping, in the sense that it is *injective* on the set of annotations that can be defined relative to a domain ontology. This property makes it *reversible* and thus enables returning inferred knowledge (e.g. an answer to a query) to ITM.



**Fig. 4.** The transformation  $f$  applied to the knowledge represented in Figure 2

## 4.2 Enhancing ITM representation and reasoning capabilities

What are the gains for ITM? First,  $f$  indirectly provides formal semantics to ITM, which is that of conceptual graphs. Second, it enables enriching ITM by two features: a declarative query language, and inference rules integrated into the query answering mechanism. As  $f$  is reversible these enrichments are straightforward.

**Query language.** ITM search functions are able to search substructures of the network restricted to one topic or one association and its neighbors. On the other hand, SGs come with a search mechanism based on projection, which is able to find substructures of *any* shape. Let us consider a knowledge base  $K$  composed of a support and a fact base  $F$ . Given a query  $Q$  represented by a SG,  $K$  answers to  $Q$  if there is a projection from  $Q$  to  $F$ , i.e.  $Q$  can be deduced from  $K$ . Every projection from  $Q$  to  $F$  can be seen as an answer to  $Q$ . More generally, a query may include distinguished (generic concept) nodes. In this case the answer is restricted to these nodes. Classically, these nodes are marked by a '?' symbol

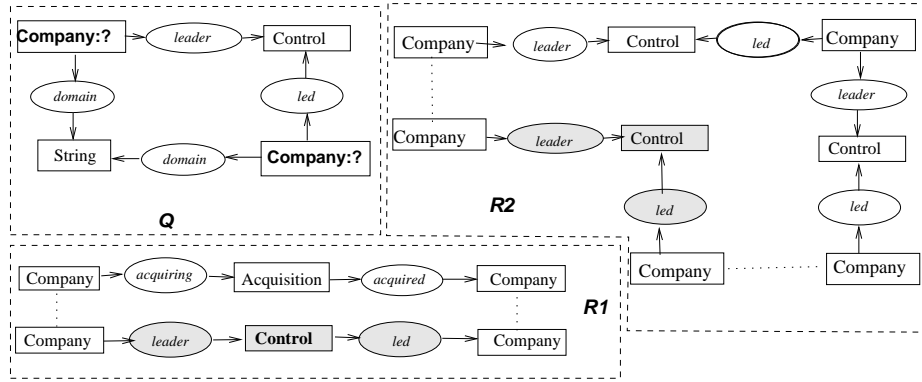


Fig. 5. A query and two rules

(See Figure 5:  $Q$  asks for couples of companies satisfying certain conditions). This kind of query is equivalent in expressive power to conjunctive queries in databases. By extending the syntax of TMs with the '?' symbol we obtain a query language for ITM. It is important to notice that, due to the reversibility of  $f$ , every answer to the query (the answer being the projection itself or the subgraph of  $F$  image of  $Q$  for this projection) can be translated into an answer to the original TM query problem.

**Rules.** In the  $SG$  family, SGs are used to represent queries and facts but also as building blocks for more complex kinds of knowledge, including *inference rules*. An inference rule expresses knowledge of form “if *hypothesis* then *conclusion*”, where hypothesis and conclusion are both SGs. In Figure 5 two rules are represented. Each dotted line connects a node in the hypothesis and a node in the conclusion; these nodes are called *connection nodes*. The nodes of the conclusion which are not connection nodes are colored in gray. The rules express properties of the concept type “control”: acquiring a company leads to control it ( $R1$ ) and exercising control is transitive ( $R2$ ).

A rule  $R$  can be applied to a SG  $F$  if there is a projection from its hypothesis to  $F$ . Applying  $R$  to  $F$  according to such a projection  $\pi$  consists in “attaching” to  $F$  the conclusion of  $R$  by merging each connection node of the conclusion with the image by  $\pi$  of the corresponding connection node in the hypothesis. See Figures 5 and 3: if  $R1$  is applied to  $G_1$  ( using  $\text{Corporation} < \text{Company}$ ) and  $R2$  to the resulting SG, one obtains  $G_2$ . Since a knowledge base is now composed of a support, facts (say  $F$ ) and a set of rules (say  $\mathcal{R}$ ), the query mechanism has to take implicit knowledge coded in rules into account. The knowledge base answers to a query  $Q$  if a SG  $F'$  can be derived from  $F$  using the rules of  $\mathcal{R}$  such that  $Q$  can be projected to  $F'$ . Let us consider again Figures 3 and 5 : the base containing the fact  $G_1$  and the rules  $R1$  and  $R2$  answers to  $Q$ ; indeed  $Q$  can be projected to  $G_2$ , which is derived from  $G_1$  by  $R1$  and  $R2$ . Forward and backward chaining schemes have been defined, which are sound and complete

with respect to logical deduction [?]. As previously, a simple extension of TM syntax and transformation  $f$  enables representation and processing of rules.

## 5 Perspectives

We have described the main steps towards providing the ITM knowledge management tool with graph-based reasoning. The transformation  $f$  from topic maps to conceptual graphs is natural and reversible, which enables on one hand return of query results to ITM and on the other hand enrichment of ITM by inference rules. A first prototype that shows the feasibility of this approach has been developed. It allows transformation by  $f$  of ITM knowledge (domain ontology and annotation base, as well as inference rules), loading the result into CoGITaNT, querying it and returning the answers to ITM. The next step will involve extending the query language to take operators on physical types (e.g. the comparison operator  $<$ ) into account. Other perspectives are related to the semantic web. Indeed several works have pointed out the closeness of SGs and the annotation language RDF/S proposed by the W3C [?]. In particular it has been proved that SG projection is sound and complete with respect to deduction in RDF/S [?].  $f$  could thus be used as a base for defining an RDF/S exchange format for ITM knowledge that would be consistent with its SG semantics, and could be extended to inference rules.

## References

- [AdMRV02] P. Auillans, P. Ossana de Mendez, P. Rosenstiehl, and B. Vatant. A formal model for topic maps. In *Proc. of ISWC'02*, volume 2342, pages 69–83, 2002.
- [Bag05] J.-F. Baget. Rdf entailment as a graph homomorphism. In *Proc. of ISWC'05*, 2005.
- [BM02] J.-F. Baget and M.-L. Mugnier. Extensions of Simple Conceptual Graphs: The Complexity of Rules and Constraints. *JAIR*, 16:425–465, 2002.
- [CM92] M. Chein and M.-L. Mugnier. Conceptual Graphs: Fundamental Notions. *Revue d'Intelligence Artificielle*, 6(4):365–406, 1992.
- [Gen97] D. Genest. Cogitant. <http://cogitant.sourceforge.net>, 1997.
- [Hay04] P. Hayes. Rdf semantics. Recommendation, W3C, 2004.
- [ISO00] ISO/IEC:13250. Topic maps: Information technology – document description and markup languages. <http://www.y12.doe.gov/sgml/sc34/document/0129.pdf>, 2000.
- [SM96] E. Salvat and M.-L. Mugnier. Sound and Complete Forward and Backward Chainings of Graph Rules. In *Proc. of ICCS'96*, volume 1115 of *LNAI*, pages 248–262. Springer, 1996.
- [Sow84] J. F. Sowa. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, 1984.
- [Top01] TopicMaps.Org. Xml topic maps (xtm) 1.0. <http://www.topicmaps.org/xtm/1.0/>, 2001.