Evolutionary Information and Decision Support Systems: An Integration Based on Ontologies.[1]

M.V. Hurtado, J. Parets

Department of Languages and Information Systems
University of Granada. ETS Ingeniería Informática.
Avda. Andalucía, 38. 18071 - GRANADA (Spain)
{mhurtado , jparets }@ugr.es

Abstract. In order to use information stored in different repositories, it is necessary to improve the automatization process of decision-making. As a result, it would be necessary to share data. One of the main motivations of our emphasis on Semantic Views building is the possibility of sharing and reusing knowledge between the Information System, the Decision Support System and other external repositories. In this paper, we will try to show that Ontologies can be used in practice as a Semantic View. We also discuss some of the limitations of the Ontology described using Description Logics in a complex domain and the limitations which we have discovered when the intrinsic evolutionary aspects of Information Systems and Decision Support Systems must be modeled. Another important problem is how a change in such systems may produce a propagation of changes which affect the levels of the system (data) and the metasystem (data structure). For this purpose, we propose a specialized ontology based on the Object-Oriented Approach. Subsequently, we have obtained some equivalent heterogeneous graphs and these can be used to represent change propagation in the IS, DSS and Semantic Views. Most of the problems mentioned will be exemplified by means of a concrete case: a DSS for risk operations in financial institutions, the class structure of which is outlined.

1 Introduction

At present, it is widely accepted that information constitutes one of the most important resources in an organization, and information is undoubtedly the key element in the decision-making process. Under these premises, it is necessary to work with a large quantity of heterogeneous and distributed data in the automatization process of decision-making. In such a situation, Decision Support Systems (DSSs) are frequently developed in an *ad hoc* way which does not take into account the fact that part of these data are included in the Information System (IS).

[1] This research is partially supported by a R+D project of the Spanish CICYT (TIC2000-1673-C06-04).

In the first section, we will discuss the need to share these data between both ISs and DSSs in order to avoid problems of redundancy and those arising from updating information and change propagation. This obviously implies the classic problem which arises from the sharing of information.

This problem may be partially solved by providing semantic views over the repositories that hide all the technical and organizational details associated with data.

A study of the main possible mechanisms used in the design of semantic views will then be presented.

In the following section, as a result of research carried out in the last decade, Ontologies [12], [13] described through Description Logics (DLs) [9] are suggested as an interesting semantic representation, which can be used here.

However, although the DLs are apparently equipped with complete sound reasoning and terminating procedures, they still suffer from several limitations not only when they are used to represent complex domains but also efficient evolution mechanisms.

Consequently, a new semantic representation can be considered in order to take into account the evolution of both kinds of systems (Information and Decision Support).

Following the lines of previous papers on software evolution [18], [19], [20], [22], section 7 studies the possible evolution of semantic views to be considered and investigated by providing effective graph mechanisms for semantic evolution.

Throughout the paper, we will use an example based on the study of a DSS for risk operations in financial institutions. This example will help to understand many of the outlined problems and the proposed solutions.

2 Information Integration Through Semantic Views.

When Decision Support Systems are developed in an *ad hoc* way some important problems are produced:

- Redundancy problems: data is in both the IS and the DSS, most of the time in different formats.
- Update problems: changes produced in either the IS or the DSS are not reflected in the other.
- Change propagation problems: when data structures are modified, legacy data can remain inconsistent.

In order to avoid these problems nowadays, organizations need to have solved the problem of sharing information with the IS and with other repositories (available through the communication networks) when the DSS is developed and used. In this situation we do not overlook the fact that data included in different repositories is always interpreted and has a meaning which depends on the system. Obviously this entails the classic problem of information sharing. But the problem of sharing may be partially solved by providing semantic views over the repositories that hide all the technical and organizational details associated with the data. Furthermore, semantic views on specific domains reduce the problem of semantics associated with stored data [3]. Figure 1 shows a general scheme in which semantic views (ontologies) are

used to integrate heterogeneous information content in the IS, DSS and other repositories.

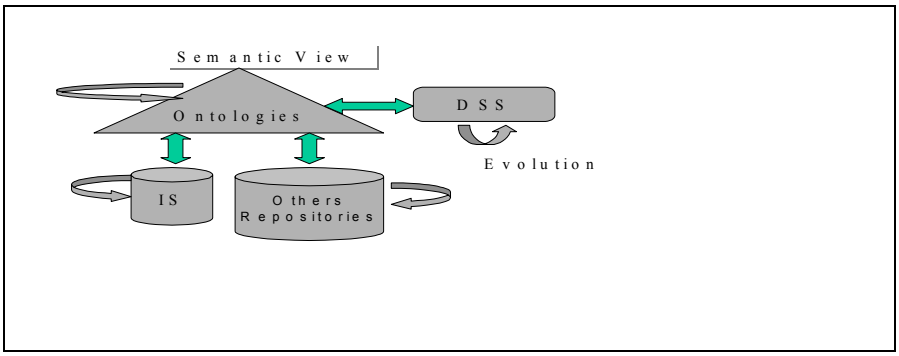

Fig. 1. Partial solution for the problem of information sharing

3 Formalisms Used In Obtaining Semantic Views

Semantic views are usually expressed by using declarative knowledge representation languages. These languages provide syntax, a set of inference rules, a vocabulary of non-logical symbols, and sometimes a graphical representation that restricts the acceptable interpretations of the symbols in the vocabulary.

Different formalisms have been proposed in order to obtain these views:

- *Systems based on DL*, [9]: this kind of system takes an object-centered view, where the world is modeled as individuals connected by binary relationships and grouped into classes (called concepts). In every DL_System, the concepts of the application domain are described by means of concept descriptions. These are built from atomic concepts and roles using the constructors provided by the DL Language. DL-Systems support a variety of inference mechanisms, such as subsumption, inconsistency detection, memberships and others.

- *Knowledge Based Systems* descendants of the KL-ONE [6], [7], [23] called *Terminological Systems*, (i.e. BACK [21], CLASSIC [4], [5], Kris [2], LOOM [14]). These systems are used to make the terminology of an application domain explicit in a similar way to DL_Systems. In addition, Terminological Systems automatically classify these definitions and queries into a taxonomy according to semantic relations such as subsumption and equivalence. The queries also help to discover what the relevant repositories are.

- *Ontologies* [12],[13] are suggested as an interesting semantic representation, which can be used here, for several reasons:

-They provide declarative and concise specifications of semantic information.
-We can obtain a logical schema of the shared information, which reduces the semantic loss in classical data models. For instance, the relational approach does not explicitly convey what the entities and the relationships are in the real model, since both are represented as tables. This situation implies different difficulties when common information belonging to more abstract artifacts is needed.
-They allow the information found in data repositories to be described independently of the underlying syntactic representation of data. Each data repository is viewed at the level of the relevant semantic concepts, and maintains a hierarchical organization of concepts which is very useful when dealing with large collections of definitions (a very simple case would be a type hierarchy which specified classes and their subsumption relationships). In this context, a mapping process of the ontology concepts and the repository terms can be used. The syntax and the semantics of the mapping can found in [15]. DSS and IS can express their information needs using these semantic views (ontologies) and the query processor must obtain the corresponding answer by accessing the underlying data repositories.

For all of these reasons and because ontologies have nowadays been made as unifying formalisms used in different domains, we advocate the use of ontologies as the Semantic View.

4 Understanding Ontologies

In the context of knowledge sharing, the term ontology is used as an explicit specification of a conceptualization [12]. That is to say, the ontology is a description of the concepts and relationships that can exist for an agent or a community of agents.

Ontologies are designed to enable knowledge to be shared and reused. Moreover, ontologies are agreements about shared conceptualizations. Shared conceptualizations include conceptual frameworks for modeling domain Knowledge. Ontological commitments are agreements to use the shared vocabulary coherently and consistently. Consequently, different kinds of systems sharing a vocabulary need not necessarily share a knowledge base.

In our case, a common ontology defines the vocabulary which allows queries and assertions to be exchanged between the IS, DSS and other repositories. Figure 2, which is explained in greater detail in the next section, shows the terms included in the ontology proposed, terms such as *Client*, *Transaction, Asset, Loan*, etc., which can be used for the IS and DSS in a Financial System.

Each system knows facts the other does not. For example the IS has information about *Financial_I or Branch*, which is not required by the DSS. On other hand the DSS has information about *Expedient* or data about *Non-client*, which is not necessary for the IS.

Furthermore, a system that commits to an ontology is not required to answer all the queries that can be formulated in the shared vocabulary.

In short, a commitment to a common ontology is a guarantee of consistency but not completeness, with respect to queries and assertions using the vocabulary defined in the ontology. Finally, it should be noted that ontologies are specified in the form of definitions. Traditionally Description Logics [9] are used to describing the terms definitions of the ontology.

5 Using Application Ontologies to the DSS Domain

In order to understand many of the outlined problems and the proposed solutions, we will use an example based on the study of a DSS. In this section, we show an example of how these mechanisms can be used to make a semantic view.

The proposed example tries to model a DSS to automate the operations and decisions accomplished during a risk operation in a financial institution. Because we are interested in some aspects of the problem, which is considered later, we will simplify the specification.

We need to take into account the fact that in all financial institutions, there are two groups of operations: those involving liabilities (investment funds, fixed term deposits, current accounts, thrift accounts, values, etc.) which constitute a deposit for the institution, and those involving assets (loans, credits, warranties, discounts, etc.) which imply a risk for the institution. The DSS will focus on the last group.

An operation of this type can be requested by individuals as well as by legal entities, whether they are clients of such an institution or not. The financial institution calculates the *Leverage Coefficient* by using information given by the user and information contained in some external repositories such as ASNEF databases (banking databases of possible non-fulfillment and their current situation) and the RAI database (a database which refers to unpaid bank bills, bills of exchange, etc.) and data included in the IS when the customer is a *Client*.

As a result of this coefficient and information provided by the customer, the institution approves, refuses or interrupts the operation and communicates the result to the person who requested it.

Figure 2 shows a partial view of the ontology. The knowledge domain is more complex (for example not all the terms used by the *Debit* operation are considered) but the ontology which we have made only considers some of the important, relevant concepts and roles used in the DSS domain.

In this figure, you can see the concepts which are only used by the IS (*Financial-I, Branch, Current_account, Debit-Transaction*, etc.), other group of concepts and roles such as *Expedient* or *No_client* which are only used by the DSS, and another group which is shared between the DSS and IS. There are others concepts (*Anything* and *Person*) which have only been included for the semantic view design.

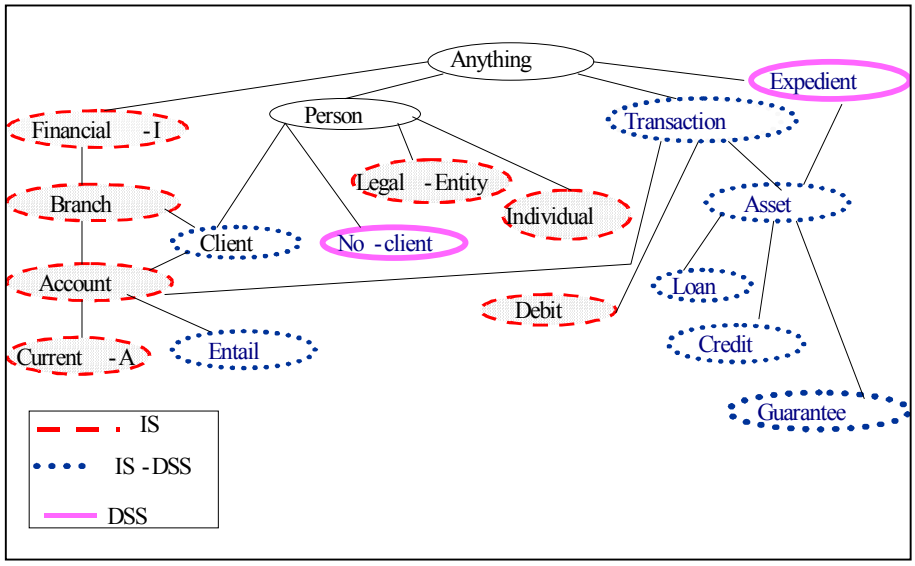

Fig. 2. Partial view of the Ontology used by the IS-DSS for risk operation in financial institutions

In this proposed example, IS is formalised through the Relational Approach. Figure3 shows the mapping between the concept '*Legal-entity*' and its equivalent table. The interface programs are responsible of connecting concepts in the ontology with the relational tables (external model).

Table Model	Attribute-name	Null?	Domain
	CIF	N	Id
	Sector	S	Sector
	Incorporate-date	S	Date
	CNAE	N	CNAE

SQL Code

```
CREATE TABLE Legal-entity
    (   CIF      char(10)    not null,
        sector  char(20)
        incorporate-date    date
        CNAE   char(2)
  PRIMARY   KEY (CIF) ;
Sector
```

Fig. 3. Mapping between the concept '*Legal-Entity'* and the equivalent table

6 Limitation of the Description Logic in the Complex Domain

Description Logics [9] are Artificial Intelligent formalisms which allow domain knowledge to be represented by focusing on classes of objects and their relationships and by offering inferences on the class structure.

However, previous works [8] have shown that the Description Logics that are equipped with sound, complete, and terminating procedures still suffer from several limitations that are not acceptable when representing complex domains similar to those we have proposed above. Here is a list of the most important limitations:

- The interpretation domain is flat, in the sense that logics consider the world to be made up of elementary objects (grouped in concepts) and the binary relation between them.
- One consequence of the previous property is that n-ary relationships are not supported.

In our example there are certain kinds of properties which cannot be represented by simply modeling the n-ary relation in terms of n-binary relations (for example it might be desirable to assert that *Account* linked to *Expedient* by the respective role is *Entail-Account*.)

- General inclusion axioms are not usually supported. Although inclusion axioms are essential when we want to assert properties of classes and relations as required in the complex domain or when we want to study the possible evolution of semantic views (for example you do not include an axiom of a-cyclic relationships or incompatible relationships).

Although these limitation are important, they are partially solved by special DLs as proposed by Nebel [16], [17], Baader [1] and by De Giacomo and Lenzerini [8]. But these DLs do not include evolution mechanisms. This is a very important problem because these mechanisms are needed when we want to study the possible evolution of semantic views. We should not overlook the fact that both the Information System and the Decision Support System are dynamic systems (i.e. they change through time), active systems (they carry out processes of change) and that they are open systems (changes in the environment produce changes in the system), and for all these reasons change mechanisms are necessary.

7 The Evolution Problem In The Integration Framework

The logical schema of information obtained in the previous semantic view must be enriched in order to allow us to manage those intrinsic evolutionary aspects, and this characterizes the modeling process of both kinds of systems.

In order to provide concrete change mechanisms, we will study the evolution of the IS and the DSS when the IS is formalized using the Relational Approach. This model is very easy to use because of the mathematical rigor in the definition of data representations, operators and the simplicity of data structures. In addition, the technological advance in relational databases facilitates its use in information systems.

The DSS is formalized using the Object-Oriented approach because notions of identity, classification, polymorphism and inheritance promote an interesting way of organizing the objects and their activity.

The O-O approach and some advanced O-O mechanisms (multiple inherence, multiple membership and dynamic classification) introduce changes in the semantic views which improve the intrinsic evolutionary nature of the IS and DSS.

Figure 4 shows the class model of the previously proposed DSS using the Object-Oriented Methodology UML. Examples such as multiple inherence (terms *Open-credit*) or dynamic classification (being a *Client* or *Non-client* depends on the *Trade-relation*) have an implicit semantic which can be considered as terms and roles in a specialized ontology based on the object model

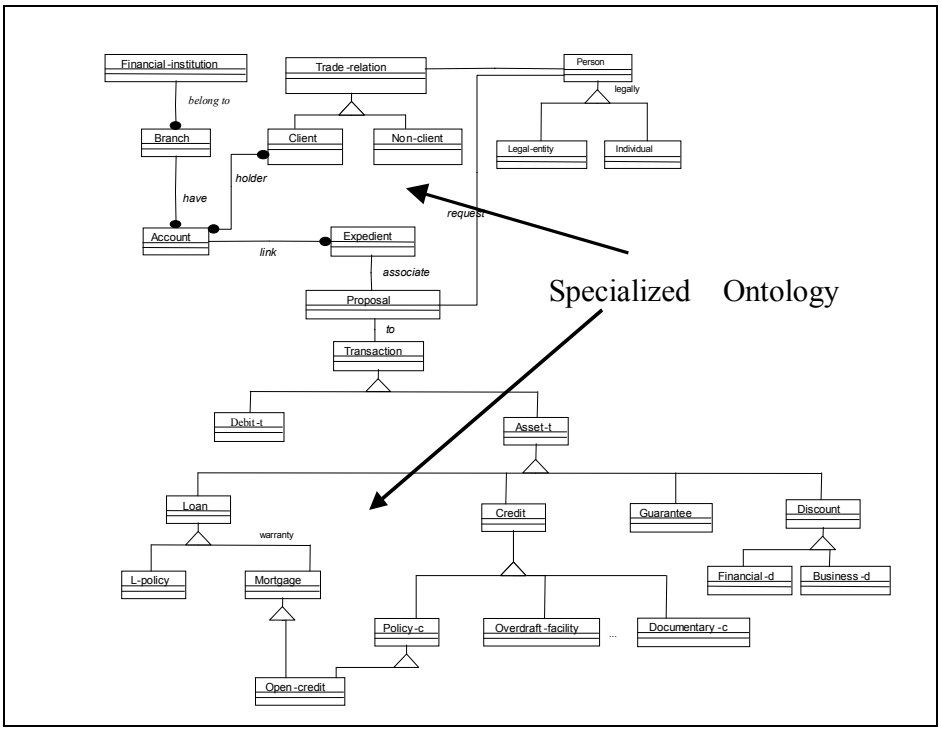

Fig. 4. Class Model of the DSS using the O-O Methodology UML.

7.1 A Specialized Ontology Based on the Object Model

Although Description Logics allow changes to be made in Semantic Views, these changes have to be made *by hand* by changing the models and re-structuring the system. This is one reason for introducing implicit dynamic mechanisms (such as

classification, reusability and maintainability) that may be considered as an extension of the O-O approach.

Classification

The ontologies, described using DL, also allow implicit classification and sub-classification by means of the concepts and the roles. The instances are facts of type 'is a' and the classification relationships can be expressed by means of definition.

In the O-O approach, classification is a core concept. The objects which share data structure and behavior are grouped into classes, and the classes can be sub-classified by means of inheritance mechanisms. This implies a high degree of abstraction, which describes important properties and ignores irrelevant ones, and this is a conceptual process which is independent of the programming languages. This process is usual in human knowledge acquisition and allows abstract concepts to be expressed.

The classification schema in object-orientation is always explicit and implies a description of the static structure of the classes of the system and their relationships(see figure 4).

Reusability

Two important features of object-orientation entail a better reusability than in DLs: inheritance and information hiding.

- Inheritance has important benefits when developing Ontologies:
  - Code reusability: the code of the behavior of a class is reused by its subclasses thereby increasing maintainability and reliability.
  - Code sharing: different users and systems can share the same classes.
  - Interface consistency: inheritance guarantees that the inherited behavior is the same for the subclasses and that the objects of the subclasses interact in a very similar way.
  - Rapid prototyping: the classes developed by previous systems can be reused and refined.
- Information hiding means that when concepts and roles are reused, it is only necessary to understand their nature and interface.

Maintainability

Some of the previous properties increase maintainability in Ontologies based on the object model.

- Explicit classification facilitates the introduction of new classes and the re-restructuring of previous ones
- Inheritance allows the reuse of previous concepts.

Information hiding allows the code of the behavior of a class to be changed without changes being made in the uses of the class.

7.2 Facing The Evolution

The IS, DSS and Semantic Views are transformed over time because their structures are transformed, and as a result these systems evolve.

In this context of evolution, some of the following changes will be considered :
- Changes in the data structure of the IS: structural changes in the IS such as creation, elimination or modification of tables, can entail changes in mapping rules. However, other changes such as creation, elimination or modification class instances or tuples in the tables do not imply changes in the structure of the ontology.

-Changes in the data structure of the DSS: in the same way, operations such as creation, elimination or modification of classes or links can change concepts or roles in the ontology and the generated data which might need to be included in the IS. Sometimes, this fact may produce a change in the structure of the IS, when there are no tables where the information can be included.

- Changes in the terms or in the relationships between terms in the semantic views: in this case it is necessary to take into account that the integrity of the Semantic View is guaranteed, and to consider what changes must be propagated to the Decision Support System or to the Information System.

Consequently, the widest and most integrated vision of the evolution in these systems is needed as a result of the propagation of changes in the IS, DSS or Semantic Views to the others.

7.3 Graphical Approach

As we mentioned above, certain advantages in the specification of the semantic views evolution can be obtained when an O-O Approach enriched with advanced object-oriented techniques, such as dynamic classification, multiple inheritance, etc., is used. These techniques improve the evolutionary nature of semantics views.

Moreover, the previous example of the UML Class Model (figure 4) can be equivalent to a heterogeneous graph (see figure 5) where the nodes are different types of classes (ellipses in the graph) and arcs are different types of relationships between them (associations, specialization, generalization, etc.) or relationships defined by users (*have, link, associate-to..)*.

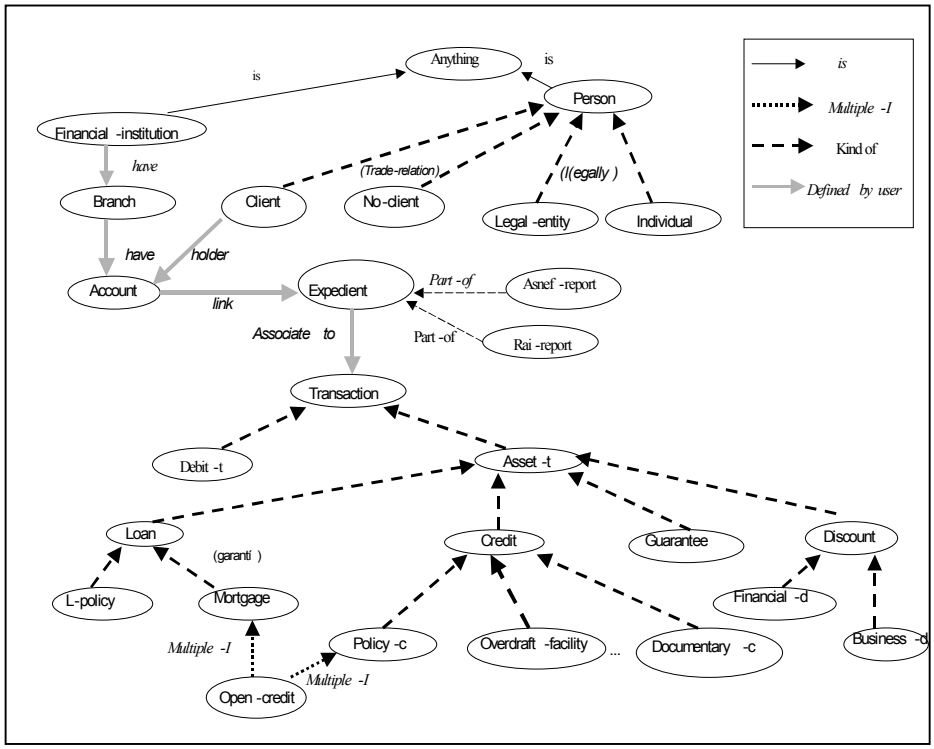

Fig. 5. Heterogeneous graph equivalent to the UML class model example for the DSS for risk operation

We are now researching practical ways of specifying these graphs. In previous papers [10], [11] we presented a set of a basic operations on the graph (such as Create_node, Del_node, Create-arc, Del_arc, Nodes-connected-to, Connection-by, etc.) and a set of basics restrictions on the graph (tree, acyclic_graph, weak_connected, etc.), restrictions on the nodes (unify name, etc.) and restrictions on the arcs (have a label, acyclical, reflexive, anti-symmetrical, transitive, incompatible_with and so on).

These operations allow the structure of the graph to be changed, and the set of restrictions on the graph, nodes and arcs help to propagate the changes.

The relation defined by the user restrictions and semantics must be explicitly defined. In the graph, not only is a special semantics adopted to O-O relationships (generalization, specialization, etc.) but there are also some implicit restrictions for these relationships. For instance, Figure 6 shows that the relation 'Kind of' always verifies some restrictions such as acyclical restriction, anti-symmetrical property, and it is incompatible with *Part of* or with *is-a* .

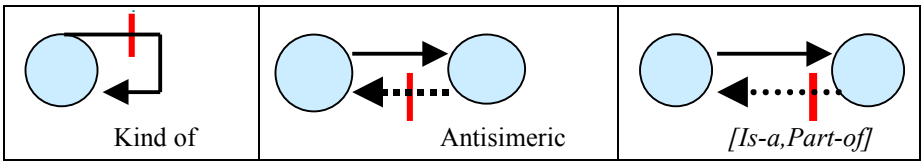

Fig. 6. Restrictions about *kind-of* relation

Changes in the structure of the IS, DSS and Semantic View would result in change operations in the structures and restrictions of the graph. This fact helps the propagation of the changes to be represented and automated. For instance, Figure 7 shows how the IS structure evolves as a result of introducing a new dynamic classification on the **Account.** This causes a change propagation in the mapping rules and also a change in the **Link relation** used by the DSS. All of these changes result in change operations in the graph. Before allowing the change, the preconditions of each operation must be checked.

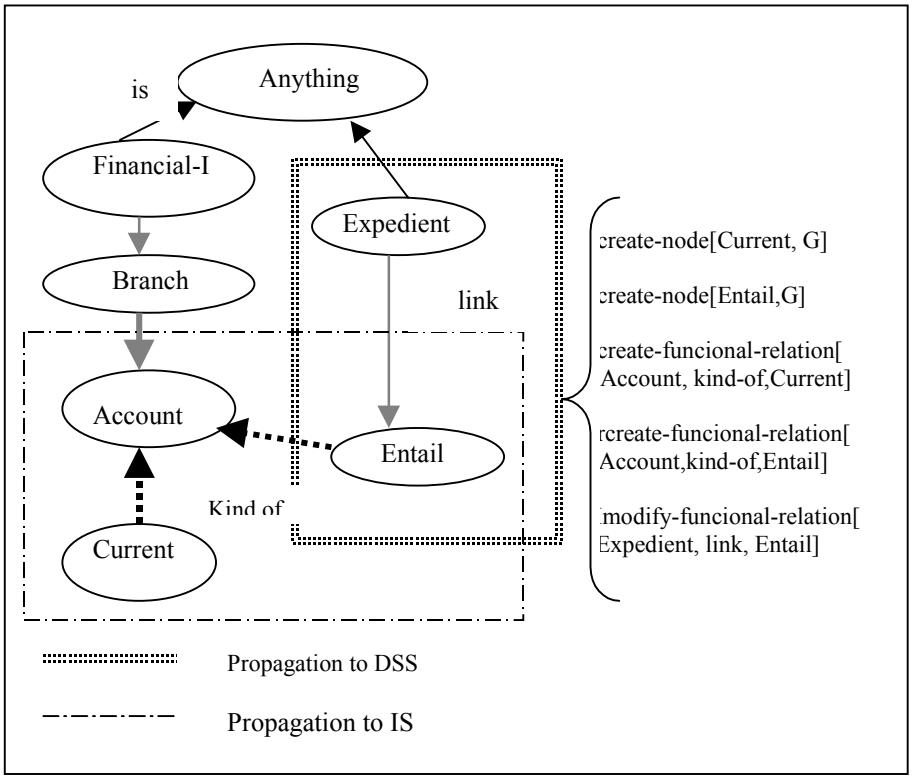

Fig. 7. Graph operation and change propagation.

We think that a graphical approach has certain advantages. From the point of view of formalization, it is easier for propagation changes to check the preconditions of operations and restrictions on the graph than the propagation change in the concepts

and roles through the DL. It is also possible for abstract data types or classes to be used for implementation (for example a C** data structure). Therefore, from a user's point of view, this approach is easier to understand because it encodes the shared information by means of a simple graph with the labeled concepts and relations.

8 Conclusions

Ontologies are an explicit partial specification of a conceptualization for the purpose of modular design, redesign and the reuse of knowledge.

The use of semantic views based on Ontologies allows, on one hand, the flexible encapsulation of data in repositories and, on the other hand, the sharing of information between the IS and the DSS.

A translation of the schemas used by ISs and DSSs to a common semantic schema is needed in order to improve information sharing.

Evolution mechanisms of both types of systems and semantic views are needed in order to be able to take advantage of the dynamic, active and open characteristics of these systems.

As a result of several limitations found in Description Logics when they are used to describe ontologies for complex domains, we have proposed that ontologies be described using an object-oriented approach and therefore using graphical representation and graph restriction.

Once again it is important to note that our approach provides mechanisms to propagate changes, and that it focuses on the evolutionary nature of the Information System, the Decision Support System and Semantic Views.

References

1. Baader ,F.(1990) A formal definition for expressive power of Knowledge Representation Languages. In *Proceedings ECAI-90*, Stockholm, Pp53-58.
2. Baader ,F and Hollunder, B.(1991). KRIS: Knowledge Representation and Inference System. ACM SIGART *Bulleting (3),*Pp.8-14.
3. Blanco,J.M., Goñi, A, Illarramendi, A. (1994), Building a federated database system: an approach using a knowledge based system. International Journal on Intelligent and cooperative Information Systems, 3(4). Pp.415-455.
4. Borgida, A.,Brachman, R.J., McGuiness, D.L.,and Resnick (1989): CLASSIC: a structural data model for objects. *Proceedings ACM SIGMOD* International Conference on Management of Data, Pp.59-67.
5. Borgida, A and Devanbu, P (1992): Knowledge base management systems using description logics and their role in software information systems. Information Processing 92 (vol. 3) Pp.171-181. Elsevier Science Publishers.
6. Brachman, R., McGuinness,D.L., Patel-Schneider, P.F., Resnick, L.A., and Borgida, A. (1990) When and How to use a KL-ONE like Language. Principles of Semantic Networks, Pp.401-456. J.Sowa ed.
7. Brachman, R. and Schmolze, J.G.(1985) An overview of the KL-ONE knowledge representation system. Cognitive Science, 9(2). Pp171-216.

8. De Giacomo, G and Lenzerini, M.(1994). Boosting the correspondence between description logics and prepositional dynamic logics. In *Proceeding of AAAI-94*, Pp.205_212.
9. Devanbu, P. and Jones, M.(1994). The use of description logics in KBSE systems. Proc.*17th International on Conference on Software Engineering*. Sorrento, Italy.
10. García-Cabrera, L., Parets-Llorca, J.(2000): A Cognitive Model for Adaptive Hypermedia Systems. The 1st International Conference on Web Information Systems Engineering, Workshop on WWW Semantics. IEEE Press. Hong-Kong, China, June (2000) 29-33.
11. García-Cabrera, L., Rodríguez-Fórtiz, M. José, Parets-Llorca, J.(2000): Formal Foundations for the Evolution of Hypermedia Systems. 5th European Conference on Software Maintenance and Reengineering, Workshop on FFSE. IEEE Press. Lisbon, Portugal, March (2001) 5-12.
12. Gruber, T. (1995). Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal of Human and Computer Studies, 43 (5/6), pp 907-928.
13. Guarino, N. and Giaretta, P. (1995) Ontologies and Knowledge Bases: Towards a Terminological Clarification. In N. Mards (Ed.) Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing 1995. IOS Press, Amsterdam. Pp.25-32.
14. McGregor, R.M.(1988). A deductive Pattern Matcher. *Proc. of AAAI-88.*
15. Mena, E., Kashyap, V.,Sheth, A., Illaramendi, A. (1996) OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. *In Proc. Of the First IFCIS International Conference on Cooperative Information Systems (CoopIS'96)* Brussels (Belgium), June. IEEE Computer Society Press.
16. Nebel, B.(1990). Reasoning and revision in Hybrid Representation Systems. Lecture Notes in Artificial Intelligence. Springer-Verlag.
17. Nebel, B.(1991).Terminological cycles: Semantics and computational properties. Principles of Semantics Networks, Pp331-361. Morgan Kaufmann, Los Altos.
18. Parets, J., Torres J.C. (1996) A Language for Describing Complex-Evolutive Software System. In F.Pichler, R. Moreno Diaz (eds.): Computer Aided Systems Theory-EUROCAST'95. Lecture Notes in Computer Science 1030. Berlin: Springer-Verlag, Pp181-199.
19. Parets, J., Torres J.C. (1996) Software Maintenance versus Software Evolution: An approach to Software System Evolution. IEEE Conference and Workshop on Computer Based Systems. Friedrichafen. March, pp.134-141.
20. Parets, J.;Anaya , A., Hurtado, M.V.;Paderwski, P.;Rodriguez, M.J.; Sanchez, G.;Torres, J (1999) A Software Development Tool for Evolutionary Prototyping of Information Systems. In Computer and Computational Engineering in Control. Electric and Computer Engineering. World Scientific and Engineering Society Press. pp.347-352.
21. Peltason, C., Schmiedel, A., Kindermann, C. and Quantz, J. (1989). The BACK system revisited. Dept. of Computer Science, Technical University of Berling, Technical Report KIT-75.
22. Rodriguez, M.J.; Parets, J.; Paderewski, P.; Anaya, A.; Hurtado,M.V. (2000) HEDES: A System Theory based tool to support evolutionary Software Systems. Lectures Notes in Computer Science. Vol. 1798. Pp 450-467.Ed. Springer-Verlag.
23. Woods, W.A. and Schmolze, J.G.(1992): The KL-ONE family. Semantic Networks in Artificial Intelligence, Pp133-178.Pergamon Press.