Domain Modeling and Knowledge Structures

Annie Stahél and Helle Wegener
København

Abstract

Natural language communication between the end user and knowledge base requires an interface with access to linguistic knowledge. Further support can be provided by a *domain model*, i.e. a module which, besides domain specific knowledge, contains common world knowledge and rules for the inference of implicit knowledge from the facts explicitly represented in the database. In our paper we present a concrete example of domain modeling. Our domain model is based on associative networks and frames. In our presentation we discuss the criteria applied, i.e. our choice of knowledge primitives and the establishment of knowledge structures by means of a network and the mapping of this network into frames.

1. Introduction

The background for what we want to present here today is the FAGFLADE project, a research project carried out at Department of Computational Linguistics at the Copenhagen Business School. FAGFLADE is short for Danish '**fag**sproglig**flade**" which means "special purpose language interface".[1]

The aim of this project is to develop and test theories and methods relevant to the construction of text interpreters for texts written in special purpose language. A text interpreter is a program which transfers the information contained in a natural language text into semantic representations which may serve various purposes. It is not our intention to build a complete text interpreter, but we have taken the development of an interpreter to be an ideal goal which defines an overall project which gives rise to a number of interesting subprojects for the investigation of general theories and principles concerning interpreters, e.g. in the domain of syntactics, semantics, lexical and terminological databases and in the domain of knowledge representation.

One of the subprojects under FAGFLADE concentrates on the construction of a natural language interface which can take a natural language question to a knowledge system as input and return appropriate (natural language) answers to the end user.

[1]This paper is a slightly modified version of a paper presented at a FAGFLADE seminar in Copenhagen in March 1993.

Proceedings of NODALIDA 1993, pages 253-265

The core of our knowledge system is a (fictitious) database VIRKBAS, which registers the relevant information about a firm, its employees, products and customers, etc. Apart from explicit information on staff, orders, complaints and the like, the database has a lot of implicit information concerning various relations between the registered entities.

In order to make this information accessible to users of the knowledge system it must be represented in a domain model which allows a representation that combines domain specific knowledge with an appropriate amount of world knowledge.

A prerequisite for the analysis of natural language questions is specific knowledge of the domain plus a certain amount of knowledge of the world referred to by the questions. An important function of the domain model is to serve as a filter that allows an acceptable user question such as 'Who are the colleagues of NN?' to be converted into a query in a formal database query language like SQL. The model must also be able to reject meaningless questions such as 'What is the salary of a TVset?'

2. The database

The specific knowledge of the domain is explicitly present in our (relational) company database VIRKBAS. The tables of the database have been structured on the basis of the Entity/Relationship diagram shown in figure 1.

The Entity/Relationship diagram shows the entity types of the domain. The database registers information about employees, customers, products, complaints etc. Each box in the diagram represents a type of entity. Each entity type is characterized by a number of attributes. Thus the entity employee, for instance, is characterized by attributes like crnumber (civil registration number), name, address and departmental attachment, among others.

The entity types of the diagram are related to each other: an employee, for instance, is employed in a department. Employees have salaries, positions, sell products, etc. Relationships as these are expressed by rhombs in the diagram. For practical reasons, the rhombs have no names in the diagram, as we are not going to focus on these relationships in the present context.

The degree of the various relations between entity types: one-to-one, one-to-many or many-to-many is important, however, since it determines the database structure. One department for instance, can have attached to it a

number of employees, but an employee can be attached to one department only.

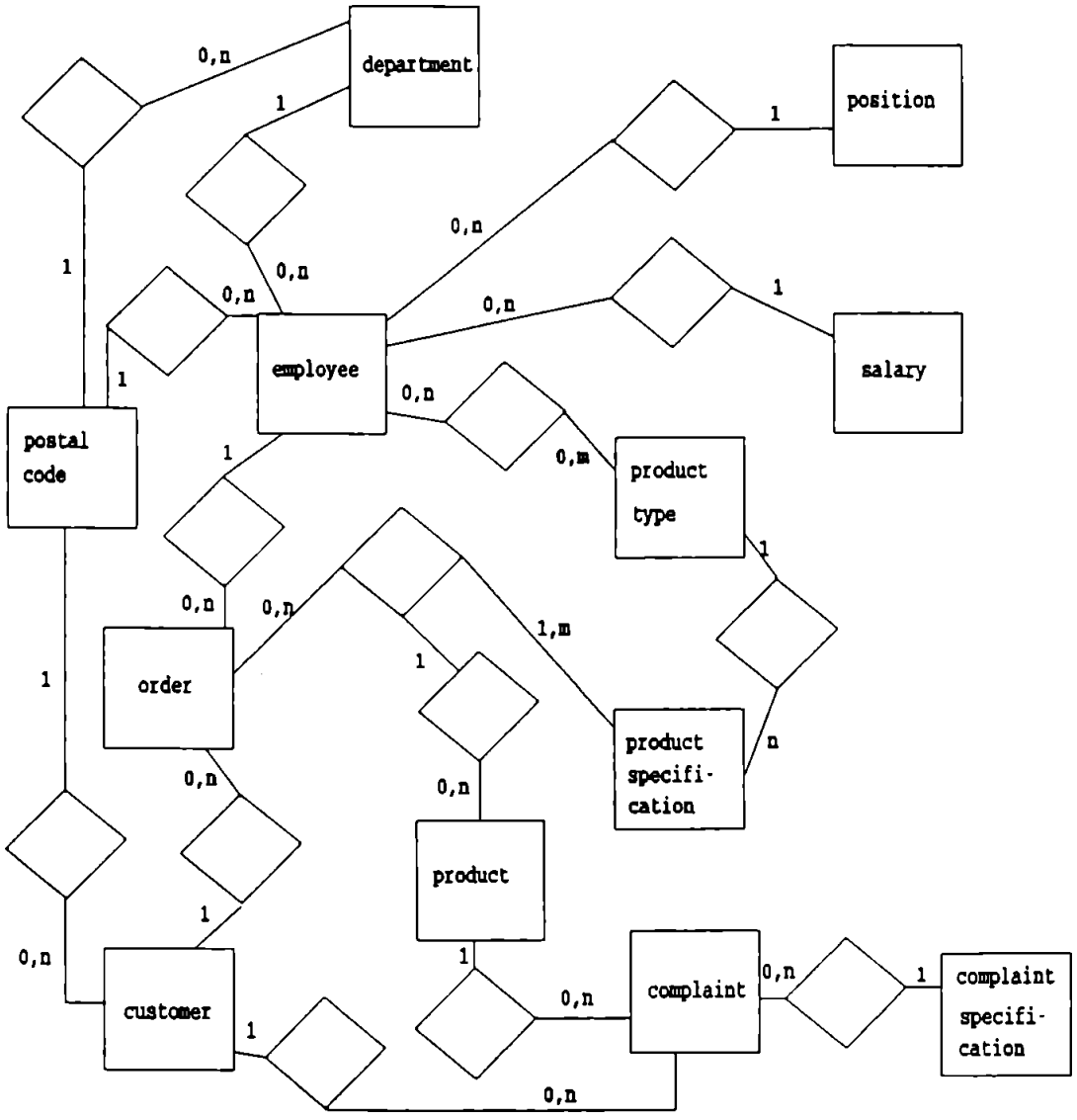


FIG 1: Entity/relationship diagram for the company

Each entity type is mapped into a separate table in the database which defines the properties of the given entity. A one-to-many relation between entity types requires that the entity characterized by the degree "1" is represented in the table of the entity type characterized by the degree "many" by a key. This key makes it possible to access all the information concerning related entities, i.e. related tables.

In the case of a many-to-many relation between 2 entity types, it is normally necessary to create a table to represent this relationship, and this table is then constituted by a key from each of the 2 entities. In the diagram such a relationship exists between "order" and "product specification".

Figure 2 shows the two entity types "department" and "employee" realized as tables in the database. Some of the attributes, which were not included in the diagram, can be seen in the tables where they appear as names of the columns.

dept:

NO	NME	STRT	PCODE	TELNO
1	Sales	Nørrebrogade 12	2200	31859511
2	Administration	Østerbrogade 75	2100	39279140

emp:

NO	FNME	SNME	..	..	..	SX	..	DEPTNO	..	POSTYPNO	WGCODE
1	Signe	Pedersen	..	..	..	f	..	2	..	1	8
5	Hanne	Osteen	..	..	..	f	..	1	..	3	5

FIG 2: Examples of tables of the database

In order to ensure the easiest possible retrieval of information from the database, two views concerning employees and sales activities of the company were created. A view is a virtual table created as a conglomerate of several base tables. Information retrieval from a view is uncomplicated, but virtual tables suffer from certain inadequacies. Updating the base tables via a view is not possible. Furthermore, any restructuring of the base tables would also demand a redefinition of the views. Finally, the meaning of natural language words is defined in terms of semantic predicates related directly to the base tables and not to the views. Consequently, we decided to do away with the views.

3. The semantic net

Apart from the explicit facts represented in the tables of the database, the domain model, as already mentioned, requires a representation of a certain amount of world knowledge and possibly additional expert information concerning the relations between the entities of the domain. The system must have access to the facts that managing directors as well

as area managers are both a kind of managers, that a department is a part of a firm, and that the managing director is the superior of all other types of employees.

Semantic nets have proved to be useful structures for total representations of individual units of information related to each other in such a way that departing from one unit of information it becomes possible to access information available in related parts of the total structure.

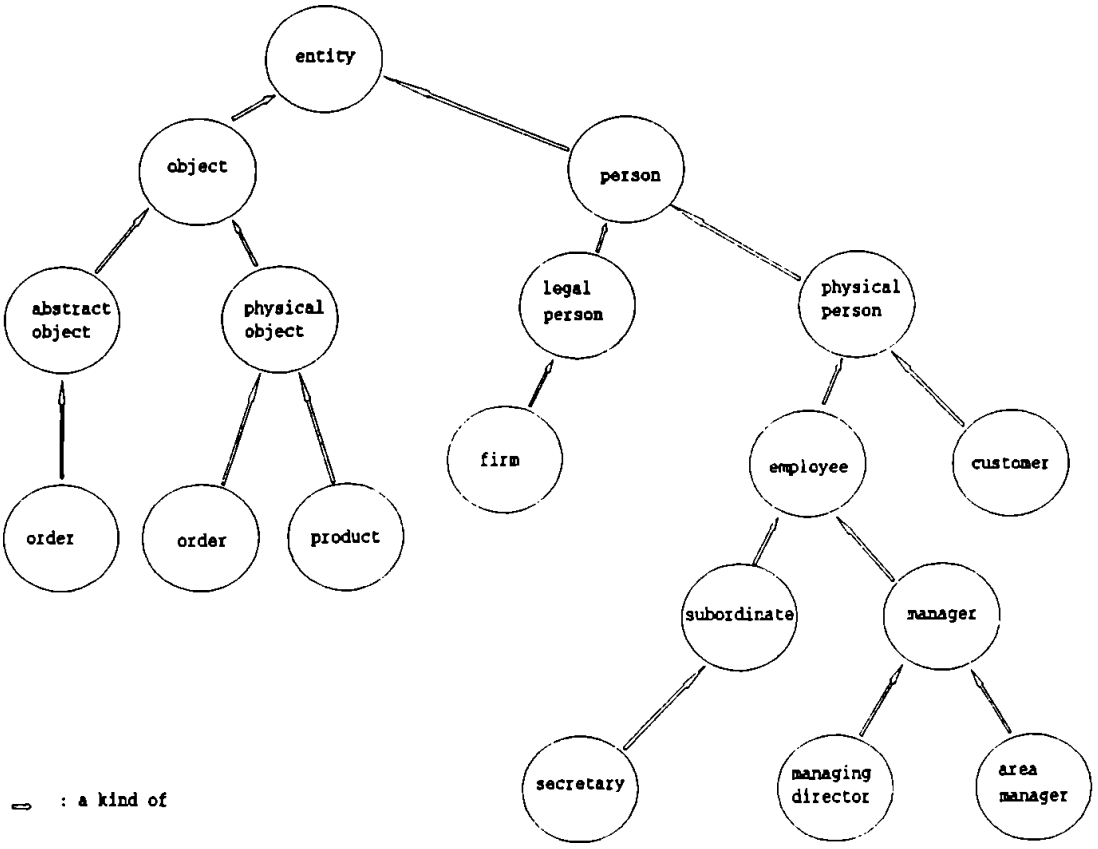


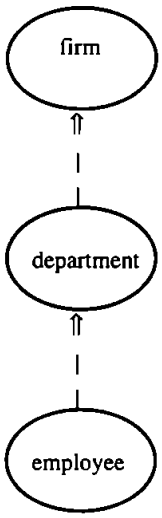
FIG 3: Generic relations

The semantic net consists of nodes representing concepts of the domain (most of which correspond to the entity types of the database), and links between the nodes that represent different types of relations between them. Two types of conceptual relations are established, one of which is the generic relation which can be seen in figure 3. Each daughter node represents a-kind-of the concept represented by the mother node.

The construction of the hierarchy is based upon the presence of characteristic features which distinguish the concepts. Thus PHYSICAL PERSON is distinguished from LEGAL PERSON by the feature crnumber.

The nodes LEGAL PERSON and PHYSICAL PERSON represent concepts which are introduced into the net in order to relate FIRM to PERSON to account for the fact that both firms and persons have legal capacity and share certain relations, as we shall see later.

The other type of conceptual relation is the part-whole relation that produces the hierarchy shown in figure 4. An employee is a part of a department, and a department is part of a firm.



– – ⇒ : a part of

FIG 4: Part–whole relations

It is possible to combine the two types of conceptual relations in one net. The semantic net in figure 5 combines our knowledge about the implicit generic and part-whole relations between concepts.

The advantage of combining different types of links in one net is that this makes it possible to represent role relations which exist between the entities of the domain even if the links between the entities involved are of different conceptual types. Furthermore, all the information on the domain concerning inventory of nodes and the various types of relations between them can be read directly out of the net.

Another part of the implicit information that we want to represent concerns other and more complex types of relations between concepts, i.e. the role relations that exist between the nodes of the net. Figure 6 below shows the representation of the role relations in the combined net.

Examples of role relations are relations such as: "be a superior to" or "be a colleague of", or a 3-place relation such as: "buy from" which holds between the nodes "customer", "product" and "firm". Another example is

the role relation "sell" between the concepts "firm" and "product" as ako "physical object". In this way we make implicit information explicit: in our domain firms can sell only physical objects such as radios and television sets. Consequently, inappropriate questions concerning a sale involving arguments other than firms and physical objects will be rejected.

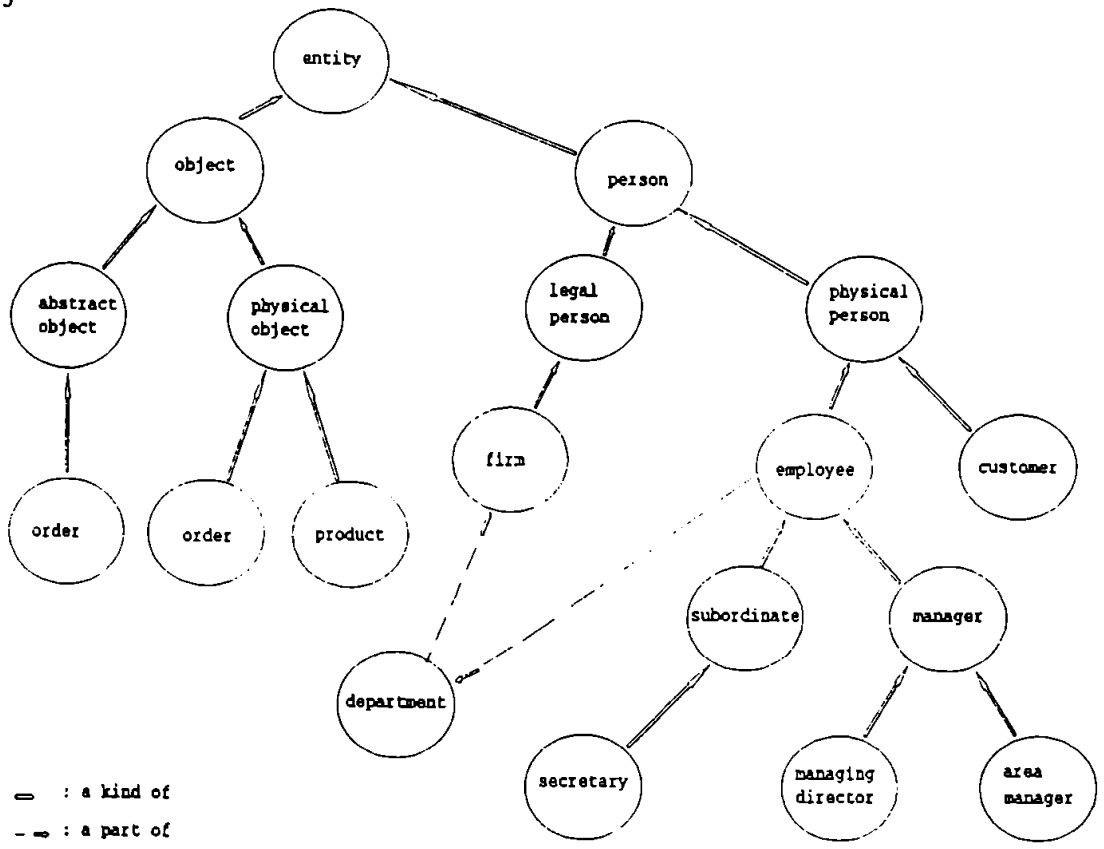


FIG 5: Generic and part-whole relations in one net.

The role relations are shown with double lines between nodes in the net. They are not directed. We consider an explicit marking of the direction redundant as the role relations are defined by the types of the concepts involved combined with the types of thematic roles associated with the concepts in a given relation. The thematic roles will be discussed in the description of the frames below.

The (identification and) choice of role relations is determined by our expectations of the questions that the end users will typically ask about this specific domain: What does the firm sell to customer NN? Who does the firm do business with? Who are the colleagues of NN?

When we introduce the role relations in this combined net the result, however, is a net which contains no less than three different types of links representing 2 types of conceptual relations, the akó and the apo links, and the role relation links. This presents certain problems concerning the

representation of role relations in the net, as the type of conceptual relation between two nodes determines the inheritance of role relations.

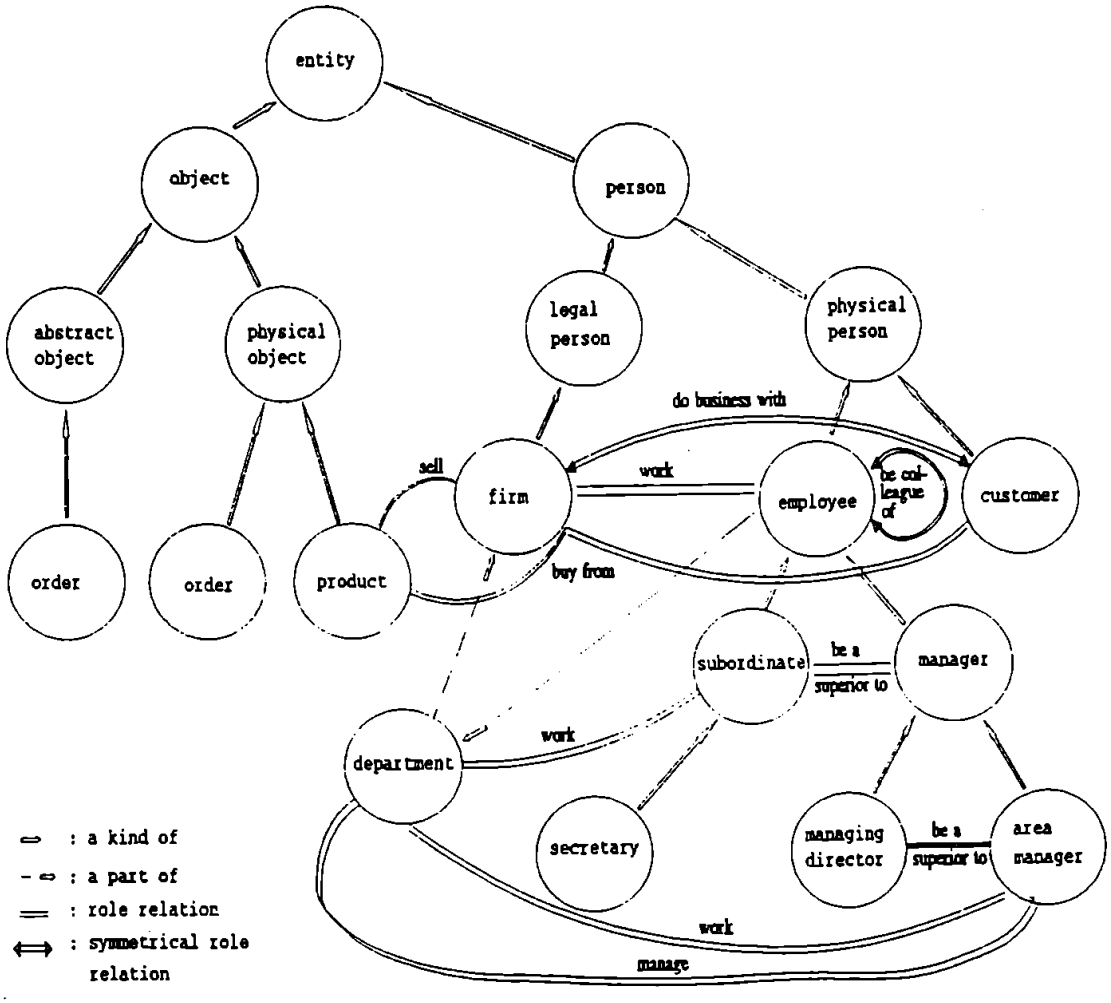


FIG 6: The combined net with role relations.

Let us look at an example: What is represented by establishing the role relation WORK between the nodes FIRM and EMPLOYEE? The role relation WORK is inherited by MANAGER from EMPLOYEE, as MANAGER stands in a generic relationship to this node, i.e. is a kind of EMPLOYEE, as well as by AREA MANAGER who, in his turn, is a kind of MANAGER. Managers as well as area managers work in a firm.

DEPARTMENT, however, which stands in a part-whole relation to FIRM, does not inherit any role relation from the mother node FIRM, since the daughter nodes in a partwhole relation do not inherit the characteristics of the mother node. This very appropriately reflects the fact that a managing director does not work in a department.

On the other hand, in order to represent that an AREA MANAGER works in a DEPARTMENT, this role relation has to be stated explicitly between these two nodes.

4. Frames

The semantic net models the relations between the concepts of the domain. The nodes of the net and the links between them are the knowledge primitives of the model. In order to be able to operate on the knowledge contained in the net and, ultimately in the database, we need a complete description of the units of information constituted by the nodes and the role relations.

For each node and role relation in the net the domain model contains a frame.

4.1 Structure and content of the frames

A frame is a data structure which represents the knowledge attached to each node or role relation, i.e. their definitorial and structural properties. All types of information are represented as feature specifications of the classical slot:filler structure. The feature values may be atomic values, e.g. the name of another frame or a specification of the datatype required for the value in question, or it may be a complex value consisting of another feature:value pair.

The basic structure of a frame is shown in figure 7.

```
[ frame   :   {framename}

  relations : (( conc : ( {type} : {frame name} ) )
                ( role : {frame name} )              )

                  rolestruct: ( { role type} : {frame name} )

  attributes: ( {attribute : datatype} )                 ]
```

FIG 7: The frame structure

The structure of the frame is based on 3 types of information: an identification of the frame, a relational description and attribute specifications.

The **frame name**, which identifies the frame, is the name of a node in the net or the name of a role relation, cf. the following 2 examples:

```
frame : EMPLOYEE
frame : WORK
```

The relational description, **relations**, may contain 2 types of relations, **conc:** the conceptual relations: ako or apo, and **role**, the role relations defined by a name for the specific type of relation, "work", "be colleague of", etc.

Role relations, which appear only in frames that describe concepts, are further defined by **rolestruct**, which contains a specification of the thematic role structure of the relation. The thematic structure is defined by a specification of role type, **role type**, and the name, **frame name**, of the frame that represents the value of a possible filler for the role type slot.

The frame for EMPLOYEE contains the following relational description:

```
relations : ( ( conc : ( ako : PHYSICAL PERSON ),
                       ( apo : DEPARTMENT      ) )
              ( role : ( WORK                  ),
                       ( BE COLLEAGUE OF       ) )
```

An employee is a kind of physical person and a part of a department and, an employee works *somewhere* and is a colleague of *somebody*. Identification of the relevant *somewhere* or *somebody* takes place via the frames for the respective role relations.

The first value specified for **role** above is the name of the frame for the role relation WORK, which contains the following specification of the thematic role structure of the relation:

```
rolestruct: ((  (actor    : EMPLOYEE   )
                (locus    : FIRM    ) ),
            ( (actor      : SUBORDINATE )
              (locus      : DEPARTMENT  ) ),
            ( (actor      : AREA MANAGER )
              (locus      : DEPARTMENT  ) ))
```

The relation WORK implies 2 participants associated with 2 types of thematic roles: actor and locus. In our domain the respective participants of a working relation can be: an employee and a firm, or a subordinate and a department, or an area manager and a department.

The last slot, **attributes**, contains the attributes of a node in the net, specified as one or more attribute names followed by a specification of a **data type**, i.e. the type of the data which can appear as the attribute value in question. Only frames which describe concepts contain attribute slots (so far).The frame for EMPLOYEE contains several attributes:

```
attributes : ( ( Position no  : INTEGER )
               ( Wage code    : INTEGER )
               ( Emp. date    : INTEGER )
               ( Dept. no     : INTEGER ) )
```

The attributes are drawn from tables in the database where they denote properties of the entities. The attributes correspond to the columns of the tables.

We have now described details of the frame structure and content. In the following we present examples of complete frames in order to show how these frames relate to each other in accordance with the semantic net in figure 6 above. The frames in examples (1) to (4) define conceptual relations. Example (5) defines a role relation.

(1)

```
[ frame   :  PERSON
  relations:     ( ( conc : ( ako  : ENTITY  ) )     ]
```

(2)

```
[ frame   :  LEGAL PERSON
  relations: ( ( conc : ( ako : PERSON    ) )

  attributes:             (( Name        :  a STRING      )
                           ( Address     :  s STRING      )
                           ( Telephone   :  an INTEGER    ) )]
```

(3)

```
[ frame   :  PHYSICAL PERSON
  relations : ( ( conc : ( ako : PERSON  ) )

  attributes :          ( ( FirstName  : a STRING     )
                          ( SurName    : a STRING     )
                          ( Address    : a STRING     )
                          ( Telephone  : an INTEGER   )
                          ( CR nr.     : an INTEGER   ) )]
```

(4)

```
[ frame   :  EMPLOYEE
  relations : ( ( conc : ( ako : PHYSICAL PERSON  )
                         ( apo : DEPARTMENT     ) )
                ( role : ( WORK          )
                         ( BE COLLEAGUE OF ) )

  attributes : (( Position code : an INTEGER    )
                ( Wage code     : an INTEGER    )
                ( Emp. date     : an INTEGER    )
                ( Dept. code    : an INTEGER    ) )     ]
```

(5)

```
[ frame   :  {framename}

  relations : ( ( conc : ( ako : STATE         ) )

  rolestruct: (((actor          : EMPLOYEE      )
                (locus          : FIRM          )),
               ((actor          : SUBORDINATE   )
                (locus          : DEPARTMENT    )),
               ((actor          : AREA MANAGER  )
                (locus          : DEPARTMENT    ))   )  ]
```

5. Conclusion

Our domain model represents a fragment of the world. It contains explicit knowledge about what we consider relevant entities and relations in the domain. It also contains implicit knowledge about the domain, i.e. the knowledge which can be defined as knowledge, either about general

logical relations between entities of the domain – or about more complex relations considered to be relevant world knowledge.

The logical basis for the representation of the explicit knowledge of the domain is established in the E/R diagram. The operational representation is established in the tables of the database.

The logical basis for the description of the implicit information is established by a semantic net that represents the entity types or concepts of our domain and different types of relations between them. The operational definition of the implicit information is stated in the frames.

The frames constitute the operational keys of the system. They combine reference to the explicit information in the database tables with the implicit relational knowledge represented in the semantic net.

Bibliography:

Börner, Stefan. 1987. *Datenbankunterstützung für wissensbasierte Systeme.* LILOG REPORT 10, IBM Deutschland GmbH, Stuttgart.

Cruse, D.A. 1986. *Lexical Semantics.* Cambridge University Press.

Date, C. J. 1990. *An Introduction to Database Systems.* 5th edition. AddisonWesley Publishing Company.

Woods, W. A. 1991. *Understanding Subsumption and Taxonomy: A Framework for Progress.* In Sowa (ed.): *Principles of Semantic Networks*, pp. 4595. Kaufmann.