SEMANTIC MODELING THROUGH IDENTIFICATION AND CHARACTERIZATION OF OBJECTS

Dan Jonsson

Dept. of Sociology, Göteborg University
Brogatan 4, S-413 01 Göteborg, Sweden

So-called *semantic data models* represent the structure of some part of reality, on the one hand, and the structure of the data in some database system, on the other hand. Semantic modeling is an important tool for database design. Several techniques and tools for semantic modeling have been proposed [2, 4]. A new approach to semantic modeling – the ICAROS approach – will be presented here.

BASIC ICAROS CONCEPTS

ICAROS is an acronym for *identification and characterization of objects*. The ICAROS approach is based on the simple idea that in order to *describe* something you must simultaneously *identify* it and *characterize* it. Symbolically,

description = identification + characterization.

Figure 1 exhibits the basic concepts used in the ICAROS approach and sketches the interrelationships among these concepts. Objects, object tuples, identifiers, characterizers, aspects, and descriptions will be considered in some detail in this section.

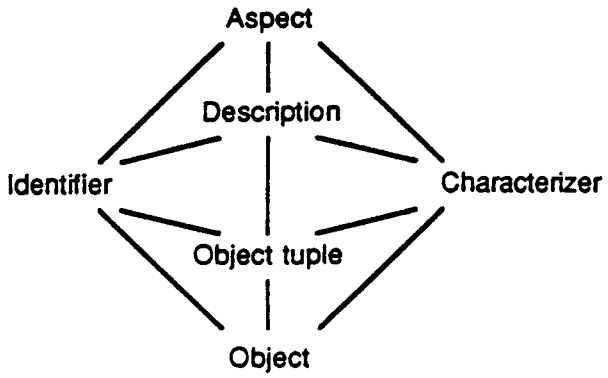

Figure 1. Basic ICAROS concepts

Objects, identifiers and identifier repertories

An *object* is any thing, occurence or phenomenon that can be identified and characterized. Examples of objects are persons, books, courses, companies, and products. (Properties and relationships may be regarded as objects, but such "objects" are not essential in the ICAROS approach, and they do not normally occur in the same models as the objects that they are properties of or relationships between.)

An *identifier* is an item that identifies (designates) a unique object. Examples of identifiers are Employee#, International Standard Book Number (ISBN), and Course ID.

An *identifier repertory* is a non-empty set of identifiers such that no object is identified by more than one identifier in this repertory.

Object tuples, characterizers and characterizer repertories

An *object tuple* $(o_1,\ldots,o_n)$ is an ordered, non-empty set of objects. (An object tuple containing only one object may be identified with its constituent object.)

A *characterizer* is an item that characterizes (depicts) each object or object tuple in a (possibly empty) set of objects or object tuples.

A *characterizer repertory* is a non-empty set of characterizers such that no object or object tuple is characterized by more than one characterizer in this repertory.

A characterizer that characterizes *one* object at a time may be said to specify a *property* that objects may possess. The characterizer also specifies a *set of objects*, namely all objects that possess the property specified. For example, "female" specifies the set of all females, while "50° F" specifies the set of all objects whose temperature is 50° F.

A characterizer that characterizes *several* objects at a time may be said to *relate* these objects or specify a *relationship* holding between these objects. The characterizer also specifies a *set of object tuples*, namely all tuples consisting of objects between which this relationship holds. For example, "married to" specifies the set of all tuples (x,y) such that x is married to y, while "situated between" specifies the set of all tuples (x,y,z) such that x is situated between y and z.

The notion of characterizer as presented above is synonomous with that of *predicate* in predicate logic, and the concept of the *degree* of a predicate is relevant in the present context, too. A characterizer that specifies a property is said to be of *degree 1*; a characterizer that relates *n* objects is said to be of *degree n*. For example, "female" is of degree 1, "married to" is of degree 2, and "situated between" is of degree 3.

On null elements

There is a subtle but important difference between, say, the characterizer "50 years old", on the one hand, and the

characterizer "known to be 50 years old", on the other hand. Specifically, for every characterizer repertory $\{c_1,\ldots,c_n\}$ there exists a corresponding repertory {"unknown", "known to be c_1", ..., "known to be c_n"}. It is often convenient to regard the latter repertory as identical to the original one extended with a *null element*, i.e., as a repertory of the form $\{\emptyset,c_1,\ldots,c_n\}$.

Descriptions and description repertories

A *description* consists of a *characterizer tuple* and an *identifier tuple*. A characterizer tuple is an ordered, non-empty set of characterizers; similarly, an identifier tuple is an ordered, non-empty set of identifiers. A description may be written in the form

$$c_1,\ldots,c_m(i_1,\ldots,i_n),$$

where $i_1,\ldots,i_n$ are identifiers, $c_1,\ldots,c_m$ are characterizers, and each c_k characterizes the object tuple identified by $i_1,\ldots,i_n$ (so that each c_k is of degree n).

This notation generalizes that used in predicate logic in an obvious way. For example, "Mary is the mother of Barbara" may be written as "mother(Mary, Barbara)", while "John is an intelligent boy" may be written as "boy, intelligent(John)".

A *description repertory* is a non-empty set of descriptions such that no object tuple is identified or characterized by more than one description in this repertory.

Aspects and description templates

Note that an identifer may identify an object *as* something. For instance, in the description "mother(Maria, Barbara)", "Maria" identifies one object as a parent, while "Barbara" identifies another object as a child. 'Parent' and 'child' are two *modes of identification* or, equivalently, two *identification aspects*.

Similarly, an object or object tuple may be characterized *with respect to* something by some characterizer. For example, a person may be characterized with respect to age, gender, height, weight etc. Also, the characterizer "angry", say, may serve to characterize a particular person with respect to mood yesterday, mood today, mood tomorrow etc. 'Age', 'gender', 'mood today' etc exemplify *modes of characterization*, or, equivalently, *characterization aspects*.

A *description template* consists of

(a) one or more pairs $(I_k, [I]_k)$ of identification aspects and identifier repertories such that each identifier in $[I]_k$ identifies some object *as* an I_k;

(b) one or more pairs $(C_k, [C]_k)$ of characterization aspects and characterizer repertories such that each characterizer in $[C]_k$ characterizes object tuples *with respect to* C_k.

A description $c_1,\ldots,c_m(i_1,\ldots,i_n)$ is said to *conform* to a particular description template if (a) each i_k belongs to $[I]_k$ and identifies some object as an I_k, and (b) each c_k belongs to $[C]_k$ and characterizes object tuples with respect to C_k.

On repertories

The following terminology will be used in connection with repertories of different types:

- An *atomic* repertory is one that contains a single item.
- Identifier and characterizer repertories are collectively referred to as *primary* repertories.

ICAROS MODELS AND STATEMENT SPACES

A "statement" is a non-empty set of descriptions of some object system. We shall now proceed to examine how sets of potential statements – "statement spaces" – are specified by means of ICAROS models. In the process, the components of ICAROS models will be identified and defined.

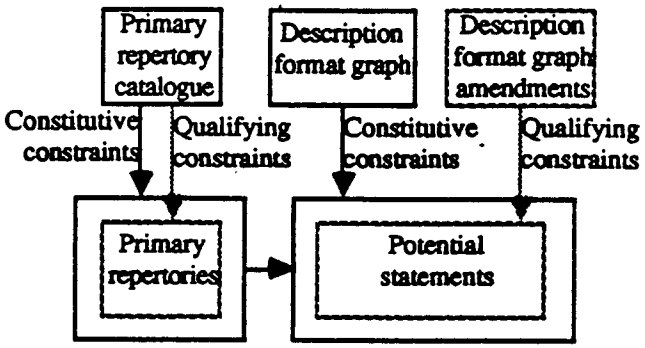

Figure 2. The specification of statement spaces

As illustrated in Figure 2, the specification of a statement space may be seen as a multi-step process with multiple inputs, specifically

- a *primary repertory catalogue* ;
- a *description format graph* ;
- optional description format graph *amendments*.

Each of these inputs – and components of ICAROS models – will be discussed below.

PRIMARY REPERTORY CATALOGUES

A primary repertory catalogue lists (a) one or more *identifier repertories* and (b) one or more *characterizer repertories*.

Note that while "unknown" can serve as a characterizer, it cannot be used to identify an object, so a null element representing "unknown" is not permitted as an identifier (cf. the previous discussion of null values). Therefore, primary repertory catalogues are subject to a *constitutive constraint* to the effect that *identifier repertories contain no nulls*.

It is also possible to introduce *qualifying constraints* such as constraints to the effect that some characterizer repertory contains no null element (indicating that there is information available about each object tuple involved).

DESCRIPTION FORMAT GRAPHS

A description format graph consists of:

- One or more *object identification circles*
- One or more *object characterization fields*
- One or more *object description frames*
- *Links*, specifically (a) one or more links connecting each *frame* to one or more *circles*, (b) one or more

links connecting each *frame* to one or more *fields*, (c) one or more links connecting each *circle* to one or more *frames*, and (d) one or more links connecting each *field* to one or more *frames*. (Note that there are no links between a circle and a field. Neither do links exist between two nodes of the same type, i.e., between two frames, two circles, or two fields.)

In concrete description format graphs, links are represented as *lines* connecting *circular*, *elongated* and *rectangular* nodes, representing object identification circles, object characterization fields, and object description frames, respectively.

As a minimum, a description format graph contains one circle, one field, one frame, and two links. Three minimal graphs are shown in Figure 3.

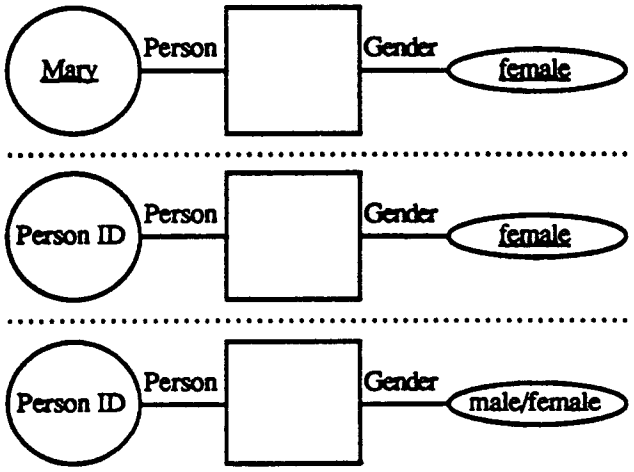

Figure 3. Three single-frame description format graphs

Object identification circles

Each object identification circle contains a reference to an *identifier repertory* in the primary repertory catalogue. An underlined identifier references an atomic repertory containing only this identifier.

Object characterization fields

Each object characterization field contains a reference to a *characterizer repertory* in the primary repertory catalogue. An underlined characterizer references an atomic repertory containing only this characterizer.

Links

Each link is associated with a label referencing an identification or characterization *aspect*.

Object description frames

Each object description frame specifies a *description template* as described below:

- ◊ Each link connecting a frame and a *circle* specifies a pair $(I_k, [I]_k)$, where I_k is the identifier aspect referenced in the label associated with the link and $[I]_k$ is the identifier repertory referenced in the circle connected to the link.
- ◊ Similarly, each link connecting a frame and a *field* specifies a pair $(C_k, [C]_k)$, where C_k is the characterization aspect referenced in the label associated with the link and $[C]_k$ is the characterizer repertory referenced in the field connected to the link.
- ◊ Consequently, since each frame is linked to at least one circle and at least one field, each frame specifies one or more pairs $(I_k, [I]_k)$ and one or more pairs $(C_k, [C]_k)$, i.e., a description template.

By introducing additional circles, fields, frames, and links, arbitrarily complex description format graphs can be constructed. A moderately complex graph is shown in Figure 4.

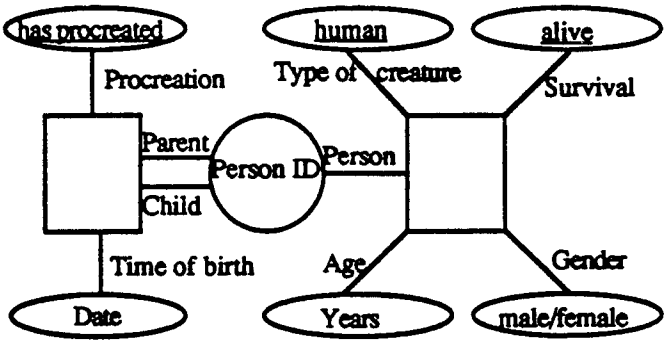

Figure 4. A two-frame description format graph

A shorthand notation

Description format graphs that reflect real-world object systems and concerns tend to obey the following structural constraints:

- ◊ Each frame is linked to exactly one field referencing an atomic characterizer repertory.
- ◊ Each field is linked to a single frame; hence, each characterizer repertory is associated with a unique characterization aspect.
- ◊ All links connected to a circle have the same label; in other words, each identifier repertory is associated with a unique identification aspect.

Under these assumptions, a convenient shorthand notation for description format graphs may be used without creating ambiguity:

- ◊ The constituent characterizer in the unique atomic characterizer repertory associated with a particular frame is exhibited within that frame.
- ◊ Fields are not represented as such; instead, the characterization aspects and/or corresponding characterizer repertories associated with a particular frame are listed within that frame.
- ◊ Links between circles and frames are not labeled. The unique identification aspect and/or identifier repertory associated with a particular circle is exhibited within that circle.

This shorthand notation is used in the description format graph for a supplier and parts example (adapted from Date [3]) shown in Figure 5.

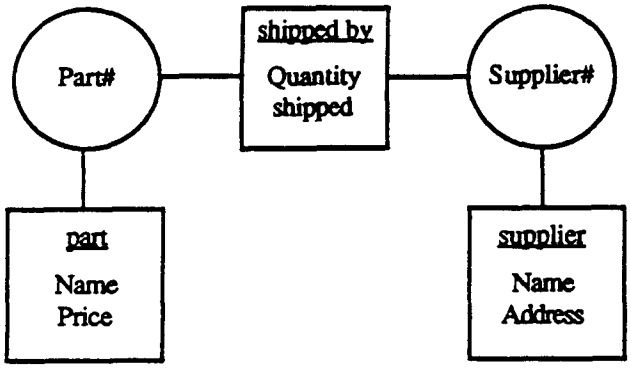

Figure 5. A simplified description format graph for a supplier and parts example

If the three structural constraints listed above are not fully satisfied, it is often convenient to use some hybrid between the full description format graph notation and the shorthand notation described above. The description format graph in Figure 4, for instance, may also be represented as shown in Figure 6.

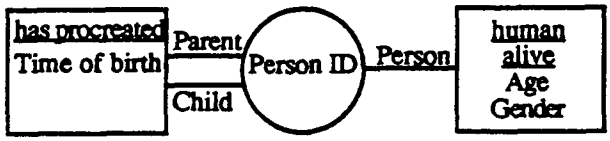

Figure 6. A hybrid description format graph

STATEMENTS AND CONSTRAINTS

Specification of statement spaces

As indicated in Figure 2, a primary repertory catalogue and a description format graph in conjunction specify a set of potential statements, i.e., a *statement space*. A *potential statement S consistent with an ICAROS model that includes a description format graph G* is a non-empty set of descriptions such that (a) each description in S conforms to some description template specified by a frame in G, and (b) for *each frame* in G the descriptions in S conforming to the corresponding description template constitute a *description repertory*.

For example, the potential statements specified by the three graphs in Figure 3 are

(a) a single statement that contains a single description: 'female(Mary)';

(b) all statements that contain at most one description of the form 'female(Person ID)' for each Person ID in the primary repertory catalogue;

(c) all statements that contain at most one description of the form 'female(Person ID)' or 'male(Person ID)' for each Person ID in the primary repertory catalogue.

Constraints on statement spaces: syntactical distinctions

Many different constraints can be defined on statement spaces, and many of these constraints are quite complex. Three possible ways of classifying constraints are with regard to

- ◊ the number of distinct *aspect occurences* (or equivalently, *links*) referenced in the definition of the constraint (the *extension* of the constraint);
- ◊ the number of distinct *descriptions* referenced in the definition of the constraint (the *order* of the constraint);
- ◊ the number of distinct *description templates* (or equivalently, *frames*) referenced in the definition of the constraint (the *range* of the constraint).

For simplicity, only constraints of order 1 (and range 1) and order 2 (and range 1 or 2) will be considered here. (Note that the range of a constraint cannot exceed its order.)

In the definitions below, let $(X_1, [X]_1),\ldots,(X_m, [X]_m)$ and $(Y_1, [Y]_1),\ldots,(Y_n, [Y]_n)$ be two (not necessarily distinct) description templates with conforming descriptions of the forms $x_1,\ldots(\ldots,x_m)$ and $y_1,\ldots(\ldots,y_n)$, respectively.

First order constraints exemplified

A constraint of extension, order, and range 1 may for instance decree that although v belongs to $[X]_k$, $x_k \neq v$ in each description associated with a particular frame. Assume, in particular, that $x_k \neq \emptyset$, where $\emptyset$ is the null value discussed above. In that case the corresponding aspect X_k is said to be subject to a frame-specific *not-null* constraint. (Incidentally, only characterization aspects can be subject to such constraints, since identifier repertories do not contain nulls in the first place.)

A constraint of extension 2, order 1 and range 1 might be that $x_i=x_j$, $x_i=f(x_j)$, or $x_i \neq x_j$ in each description associated with a particular frame.

Second order constraints exemplified

A constraint of order 2 is one that applies to two different descriptions D and D' in the same statement. D and D' may be associated with the same frame or with two different frames – i.e., the constraint may be of range 1 or 2.

For example, the presence of some description D in a statement may imply the presence of another description D' in that statement. Assume, in particular, that the presence of some D such that $x_i=v$ implies the presence of some D' such that $y_j=v$. In that case Y_j is said to be *existentially dependendent* on X_i.

On the other hand, the presence of a description D in a statement may imply the *absence* of another description D' in that statement, and this is a another general format for second order constraints.

Assume, for instance, that the presence in a statement of some D such that $x_k=v$ implies the absence in that statement of another description D' such that $x_k=v$. In that case, the corresponding aspect X_k is said to be subject to a *key constraint*.

A similar case involves the assumption that the presence in a statement of some D such that $x_i=v_1$ and $x_j=v_2$ implies the absence in that statement of another description D' such that $x_i=v_1$ and $x_j \neq v_2$. In that case, X_j is said to be *functionally dependent* on X_i.

These definitions can easily be generalized to cover the

cases where a key constraint involves several aspects and where an aspect is functionally dependent on several others. Also note that if one or more aspects $X_1,\ldots,X_n$ associated with a particular frame are subject to a key constraint then all other aspects associated with that frame are functionally dependent on $X_1,\ldots,X_n$.

Constitutive and qualifying constraints: a semantical distinction

It is an important consequence of the definitions of identifier repertories, characterizer repertories and description repertories that a description repertory whose descriptions all conform to the same description template does not contain more than one description with the same identifier tuple. (The simple proof of this assertion is omitted to save space.)

This implies that the *identification aspects* associated with a particular frame are subject to a joint key constraint, and as a consequence of this key constraint each *characterization aspect* associated with this frame is functionally dependent on these identification aspects. These are *constitutive* constraints, i.e., constraints that are *logical consequences* of the way that ICAROS models represent object systems.

In addition to the constitutive dependencies described in the preceding paragraph, a large variety of additional, *qualifying* constraints may be introduced. These constraints restrict statement spaces by specifying what is *empirically possible* and what is not.

QUALIFYING CONSTRAINTS AND AMENDED DESCRIPTION FORMAT GRAPHS

Qualifying constraints may be represented by amending description fomat graphs in different ways. One way of representing existential and functional dependencies will be explained below by means some examples.

An order system example

An amended description format graph for an order system is presented in Figure 7.

The existential dependencies represented by arrows within circles in Figure 7 may be paraphrased thus: (a) If a customer has placed an order then this customer exists; (b) if an item is included in an order then this item exists; (c) each order is ordered by some customer, and if a customer has placed an order then this order exists; (d) each order includes some item, and if an item is included in an order then this order exists; (e) consequently, if a customer has placed an order then this order specifies some item; and vice versa. In addition, the arrow in the 'Ordered by' frame represents the functional dependency that (f) each order is ordered by at most one customer.

An educational institution example

Figure 8 shows an amended description format graph for an educational institution example discussed by Lyngbaek and Vianu [5]. Note that in conventional semantic modeling terminology, 'Instructor' and 'Student' are *subtypes* of 'Person'. In the graph in Figure 8, this is reflected by the fact that 'Person' is existentially dependent on 'Instructor' as well as on 'Student'.

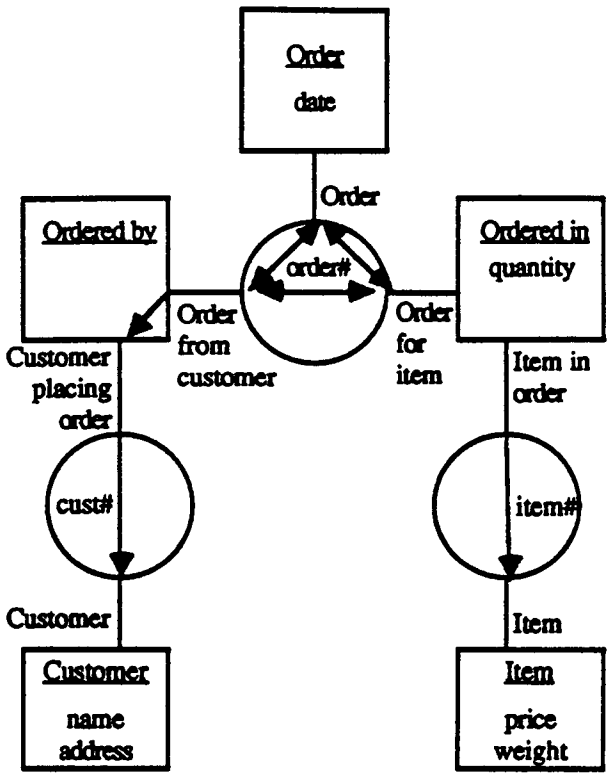

Figure 7. An amended description format graph for an order system

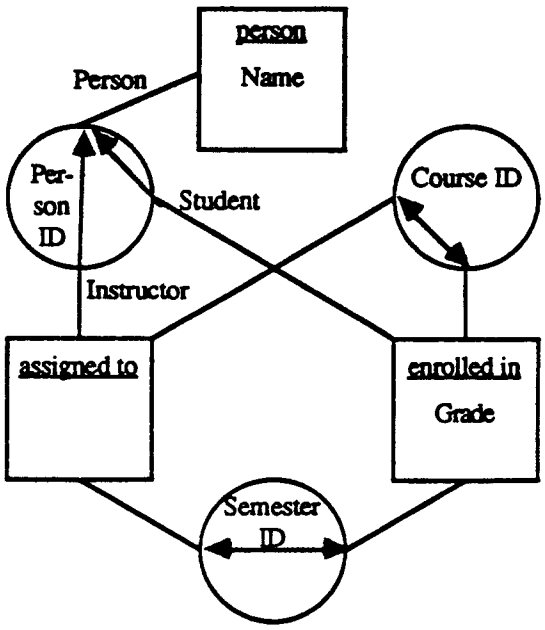

Figure 8. A amended description format graph for an educational institution

An equipment type example

A final example, adapted from Blaha et al. [1], is intended to further illustrate the notions of object types, subtypes, and inheritance.

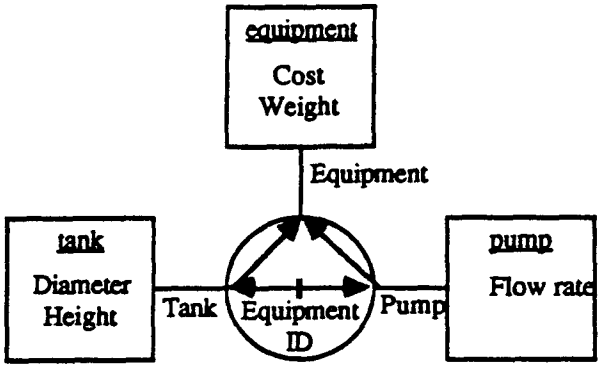

Figure 9. An amended description format graph for an equipment type example

The symbol <—|—> between the 'tank' and 'pump' links represents a sort of "inter-frame key constraint". Specifically, if a statement contains a description of the form "pump,...(x)" then it does not contain a description of the form "tank,...(x)", and vice versa, since pumps are obviously different from tanks.

On the other hand, if a statement contains a description of the form "pump,...(x)" or "tank,...(x)" then it does also contain a description of the form "equipment,...(x)", since pumps and tanks are two types of equipment. Consequently, the characterization aspects 'Cost' and 'Weight' are "inherited" by pumps and tanks.

DATABASE DESIGN

Space limitations prevents me from giving a detailed account of how to translate an ICAROS model into a database schema. Briefly,

- ◊ each *non-atomic repertory* in a primary repertory catalogue corresponds to a *domain*;
- ◊ each *frame* in a description format graph corresponds to a *relation*;
- ◊ each *link* connecting a *frame* to a circle or a field referencing a non-atomic primary repertory corresponds to an *attribute* allocated to the *relation* corresponding to this frame;
- ◊ among the links connected to particular frame, those associated with *identification aspects* correspond to the *primary key* for the corresponding relation;
- ◊ functional and existential *dependencies among aspects* correspond to *dependencies among attributes*.

CONCLUSIONS

The trend in semantic modeling research, unfortunately, seems to be toward a proliferation of notions.

In the entity-relationship approach [2], for example, *entities* (e.g., 'John', 'Mary') are distinguished from *relationships* between entities (e.g., 'marriage'). Both entities and relationships may be associated with *attributes* (e.g., 'length', 'long'). In addition, entities and relationships may be "weak" (existence-dependent) or "regular" (non-weak). To these notions, others have added *generic objects, object types, has-subtype relationships, has-instance relationships* etc.

ICAROS models, by contrast, adher to a "back to basics" philosophy and view reality as populated by individual objects only. What is the deeper reason for this divergence?

Most semantic modeling tends to proceed on the assumption that each concept used in semantic models should correspond to some particular type of real-world phenomenon. Also, reality is commonly regarded as including objects that possess properties, are associated through relations, etc; hence, a multitude of corresponding concepts is used in semantic models.

ICAROS models, on the other hand, do not attempt to *directly* represent reality. What they do is to specify a set of sets of *descriptions* of reality, introducing a level of indirection between semantic models and reality. Specifically, ICAROS models represent identifiers, characterizers, and aspects; these, in turn, serve to describe objects.

This representation philosophy makes it possible to apply Ockam's principle *"entia non multiplicanda sunt prater necesitatem"* (entities should not be multiplied without necessity) as far as the so-called real world is concerned. Distinctions between different types of real-world phenomena are replaced by distinctions between different types of signs or different modes of signification. For example, the distinctions between object instances and object types, individual objects and generic objects, entities and attributes, and entities and relationships need not be separately represented in the model, since they can all be expressed in terms of the distinction between *identifiers* and *characterizers*. In addition, the strategic distinction between constitutive and qualifying constraints is also based on the the distinction between identifiers and characterizers.

Identifiers and characterizers, of course, are new names for old concepts used in first-order predicate logic. On the other hand, ICAROS modeling introduces the notions of (identification and characterization) *aspects* and (identifier, characterizer, and description) *repertories*, and this seems to be what has to be added to predicate logic to obtain a suitable logical foundation for semantic modeling.

REFERENCES

[1] Blaha, R.M., Premerlani, W.J. & Rumbaugh, J.E. Relational database design using an object-oriented methodology. *CACM*, **31**, 4, 414-427, 1988.

[2] Chen, P.P. The entity-relationship model: Toward a unified view of data. *ACM TODS*, **1**, 1, 9-36, 1976.

[3] Date, C.J. *An introduction to database systems. Vol 1.* (4th ed.) Reading, Mass., 1986.

[4] Leung, C.M.R. & Nijssen, G.M. Relational database design using the NIAM conceptual schema. *Inform. Systems*, **13**, 2, 219-227, 1988.

[5] Lyngbaek, P. & Vianu, V. Mapping a semantic database model to the relational model. *Proc. ACM SIGMOD 1987 Annual Conference*, 132-142.