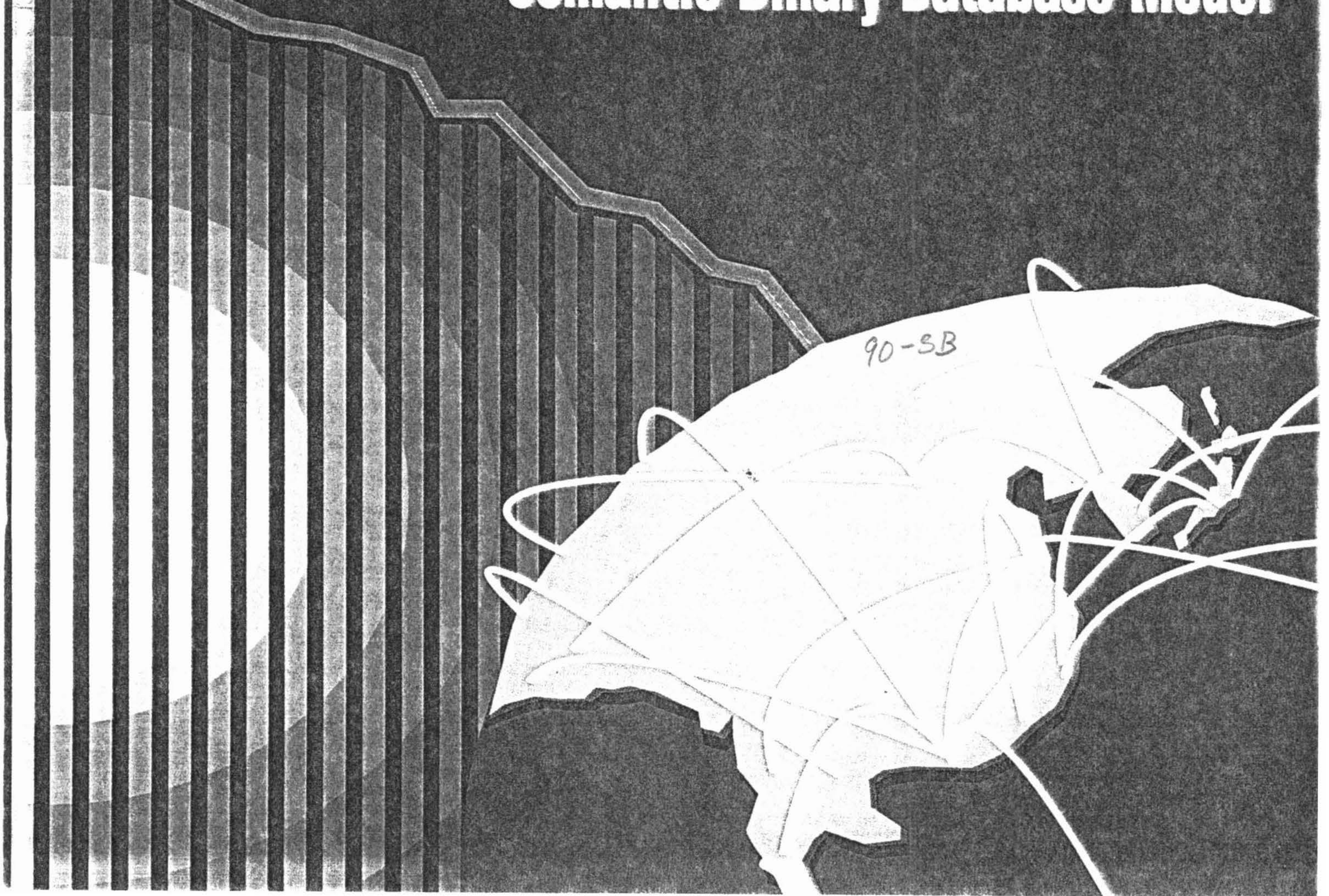
Hong Kong Computer Journal 香港電腦月刊

Journal of the Hong Kong Computer Society • Vol.6 No. 11 • November 1990 • HK$20

Feature: Telecommunications

CT2 Developments

Semantic Binary Database Model



Semantic Binary Database Model

By Naphtali Rishe

Semantic data models offer significant advantages, compared to the relational and older data models, in terms of quality and ease of database design and maintenance, data integrity, flexibility, conciseness of languages and ease of application programming. This paper proposes a variant of a semantic model which combines extensive expressive power with simplicity of use and minimality of the arsenal of semantic constructs, as well as a concise graphical notation. As a consequence, the author believes that this model can be particularly useful to the database practitioner. The model can be used in a stand-alone database management system (DBMS) or as a design tool in top-down development of high-quality databases under the conventional DBMS in the relational, network and hierarchical models.

Since *[Abrial-74]*, many semantic data models have been studied in the computer science literature. Although somewhat different in terminology and the selection of tools used to describe the semantics of the real world, they have several common principles:

- The entities of the real world are represented in a manner transparent to the user. (In the relational model, the entities are represented by the values of keys of some tables and in the network model they are represented by record occurrences.) Hereinafter, the user-transparent representations of real-world entities are referred to as "abstract objects". The "concrete objects" or "printable values" are numbers, character strings, etc. The concrete objects have conventional representations on paper and in the computer.
- The entities are classified into types or categories, which need not be disjoint. Meta-relations of inclusion are defined between the categories.
- Logically-explicit relationships are specified among abstract objects (eg, person p1 is the mother of person p2) and between abstract objects and concrete objects (eg, person p1 has first name Jack). There are no direct relationships among the concrete objects. In most semantic models, only binary relations are allowed since higher order relations do not add any power of semantic expressiveness *[Bracchi-76, Rishe-88-DDF]*, but do decrease the flexibility of the database and representability of partially-unknown information, and add complexity and potential for logical redundancy *[Rishe-88-DDF]*.

This paper presents a semantic binary model (SBM), a descendant of the model proposed in *[Abrial-74]*. This model does not have as rich an arsenal of tools for semantic description as can be found in some other semantic models, eg, the IFO model *[Abiteboul-84]*, the SDM model *[Hammer-81]* (implementation *[Jagannathan-88]*), the Functional Model *[Shipman-81]* (implementation *[Chan-82]*), SEMBASE *[King-84]*, NIAM *[Nijssen-81]*, *[Nijssen-82]*, *[Leung-87]*, GEM *[Tsur-84]*, TAXIS *[Nixon-87]* or the semi-semantic Entity-Relationship Model *[Chen-76]*.

Nevertheless, the SBM has a small set of sufficient simple tools by which all the semantic descriptors of the other models can be constructed. This makes SBM easier to use for the novice, easier to implement and usable for delineation of the common properties of the semantic models.

The SBM represents the information of an application's world as a collection of elementary facts of two types: unary facts categorising objects of the real world and binary facts establishing relationships of various kinds between pairs of objects. The graphical database schema and the integrity constraints determine what sets of facts are meaningful, ie, can comprise an instantaneous database.

An Informal Example

The schema in Figure 1 describes some activities of a dining club, including the following information:

Patron – a member of the club. For each patron it is known what are his two most favourite tables in the club. Each table has one or more names, but no two tables have the same name.

Person – any person who is a patron or waiter or both.

Charge Account – account to which meals can be charged. There is no permanent relationship between an account and a patron. Ad hoc relationships between patrons and accounts are established when the patron's meal is charged to the account. The club does not accept cash for meals. The charge for every meal is always $10, therefore the money amounts do not appear in the database explicitly.

Party – an ad hoc group of patrons eating on a particular date during the same shift, served by the same waiter and charging to the same account.

Definition

Here is a series of definitions of the SBM's concepts.

1) Databases

◆ *Instantaneous Database* – all the information represented in a database at a given instant of time. This includes the historic information which is still kept at that time. Hence the life of a database can be seen as a sequence of instantaneous databases. The first one in the sequence is often the empty instantaneous database – it is the state before any information has been entered.

◆ *Database Model* – a convention of specifying real world's concepts in a form understandable by a DBMS.

2) Categories

◆ *Object* – any item in the real world. It can be either a concrete object or an abstract object.

◆ *Value or Concrete Object* – a printable object, such as a number, a character string or a date. A value can roughly be considered as representing itself in the computer.

◆ *Abstract Object* – a non-value object in the real world. An abstract object can be a tangible item such as a person, a table, a country or an event or an idea. Abstract objects cannot be represented directly in the computer: coding and mapping schemes are needed to physically represent abstract objects in the computer. This term is also used for a user-transparent representation of such an object in the SBM.

◆ *Category* – any concept of the application's real world which is a unary property of objects. At every moment in time such a concept is descriptive of a set of objects which possess the property at that time.

Unlike the mathematical notion of a set, the category itself does not depend on its objects: the objects come and go while the meaning of the category is preserved. Conversely, a set does depend on its members: the meaning of a set changes with the ebb and flow of its members.

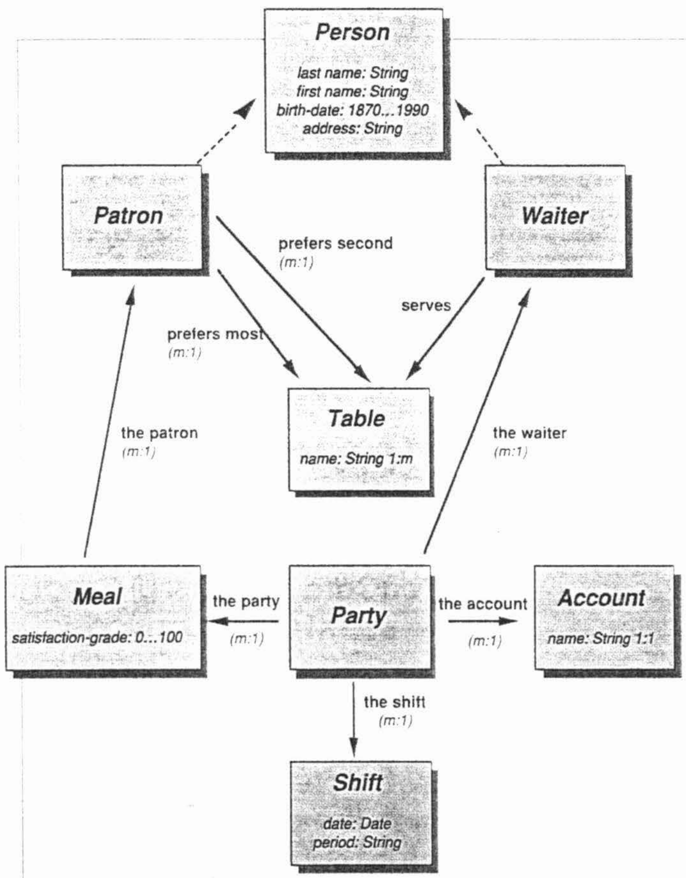
Categories are usually named by singular nouns. For example, *Patron* is a category of abstract objects. The set of all the patrons relevant to the application today is different from such a set tomorrow, since new patrons will arrive or will become relevant. However, the concept of *Patron* will remain unaltered.

An object may belong to several categories at the same time. For example, one object may be known as a person and at the same time as a waiter and as a patron.

Some of the categories in the world of our dining club are: *Waiter*, *Person*, *Account*, *Patron*, *Table*.

◆ *Disjoint categories* – Two categories are disjoint if no object may simultaneously be a member of both categories. This means that at every point in time the sets of objects corresponding to two disjoint categories have empty intersection.

◆ *Subcategory* – A category is a subcategory of another category if at every point in time every object of the former category should also belong to the latter. This means that at every point in time the set of objects corresponding to a category contains the set of objects corresponding to any subcategory of the category. For example, the category *Patron* is a subcategory of the category *Person*. The category *Waiter* is another subcategory of the category *Person*.



1) Categories are shown by rectangles.
2) Relations from abstract categories to concrete categories are shown inside the boxes of the domain-categories as relation : range type. The range is specified as a programming language data-type. Usually, relations between abstract and concrete categories are m:1. This is the default type of relations whose ranges are concrete categories.
3) Relations between abstract categories are shown by arrows between the categories' rectangles. The name and type of the relation are indicated on the arrow. The default for the type of relations between abstract categories is m:m.
4) Subcategories' rectangles are connected to their supercategories' rectangles by arrows with dashes.
5) The disjointedness of categories is indicated implicitly:
a) Two categories which have a subcategory in common are not disjoint. (The common subcategory does not have to be their immediate subcategory, that is, it may be a subcategory of a subcategory, and so on.)
b) Two categories which are subcategories of one category (not necessarily immediate subcategories) are considered not disjoint, unless otherwise declared.
c) The other categories are disjoint from each other, unless otherwise declared.

Figure 1: A binary schema for a dining club application

◆ *Abstract category* – a category whose objects are always abstract.

◆ *Concrete category, category of values* – a category whose objects are always concrete. *Patron* and *Account* are abstract categories. *String*, *Number* and *Digit* are concrete categories.

Many concrete categories such as *Number*, *String* and Boolean have constant in-time sets of objects. Thus, those concrete categories are actually indistinguishable from the corresponding sets of all numbers, all strings and the Boolean values ({True, False}).

◆ *Finite category* – A category is finite if at no point in time an infinite set of objects may correspond to it in the application's world. Every abstract category is finite.

For example, the categories *Patron, Account and Digit* are finite. The category *Number* may be infinite.

3) Binary Relations

◆ *Binary Relation* – Any concept of the application's real world which is a binary property of objects, that is, the meaning of a relationship or connection between two objects. For example, *Serves* is a relation relating waiters to tables. *Birth-date* is a relation relating persons to numbers.

The relation is descriptive of a set of pairs of objects which are related at that time. The meaning of the relation remains unaltered in time, which the sets of pairs of objects corresponding to the relation may differ from time to time when some pairs of objects cease or begin to be connected by the relation.

Notation: "$x R y$" means that object x is related by the relation R to object y. For example, to indicate that a waiter i serves a table d, we write: i *Serves* d.

◆ *Types*

– A binary relation R is **many-to-one (m:1, functional)** if at no point in time xRy and xRz where $y \neq z$. For example, *Birth-date* is an m:1 relation because every person has only one date of birth.

– A binary relation R is **one-to-many**

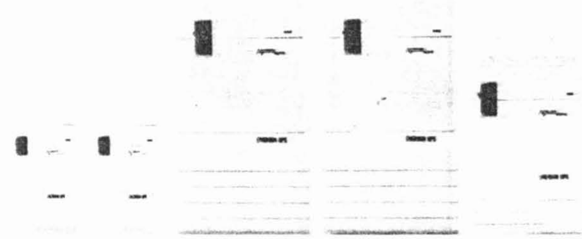
INVITATION

EMERSON UPS FOR PC

Emerson, the leader in UPS protection for two decades, adds three new members to micro- UPS family for providing a reliable power protection to PC and Workstations.

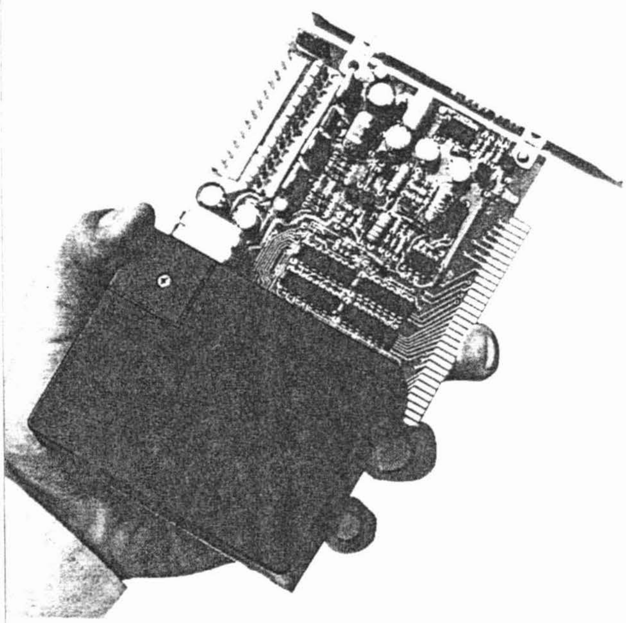
MICRO ACCUPOWER UPS

Range : 150 VA, 300 VA, 500 VA
600 VA, 800 VA, 1250 VA
and 1400 VA



SMALL WONDER UPS

Range : 1000 VA (350 VA
protected by UPS &
650 VA by RH filter)



ACCUCARD

AccuCard is the first UPS that fits in a single half-size XT/AT expansion slot. In the event of power loss, AccuCard provides enough DC battery power to automatically save a snapshot of system memory to the hard disk before the power down the system. When the power comes back, all system status, registers, buffers, memory and data intact will automatically restore. Just as if nothing had happened.

Invitation. For expanding our distribution network, we invite micro-computer sellers to join us as authorized distributors. For more information, please call **Liebert H.K. Marketing** at 555 9302.

Liebert Hong Kong Ltd.

1/F, Express Industrial Building
43 Heung Yip Road, Aberdeen, Hong Kong
Tel: 555 9302 Telex: 82087 LBTHK HX Fax: 873 4096

(1:m) if at no point in time xRy and zRy where $x \neq z$.

- Relations which are of neither of the above types are called **proper many-to-many (m:m).** For example, *Serves* is a proper m:m relation because a waiter can work in many tables and a table may employ many waiters.

- A binary relation which is both m:1 and 1:m (always) is called **one-to-one (1:1).** For example, if accounts are identified by their names, then the *Account-Name* relation is 1:1, meaning that every account has at most one name, and no character string is the name of two different accounts:

Suppose that in the current situation in our real world, the following is true: every person has at most one name and no two persons have the same name.

This does not mean that *Name* is a 1:1 relation between persons and strings. *Name* would be a 1:1 relation if the above condition were true at all times: past, present and future.

- A binary relation is **proper m:1** if it is m:1 and not 1:1.

- A binary relation is **proper 1:m** if it is 1:m and not 1:1

For example, all of the types of relations mentioned in the previous example are proper.

Since the *Account-Name* is 1:1, it is also 1:m, m:1 and m:m. Since this relation is proper 1:1, it cannot be proper 1:m, m:1 or m:m.

Figure 2 shows the classification of all relations.

◆ *Domain and range* - A category C is the domain of R if it satisfies the following two conditions:

i) whenever xRy then x belongs to C (at every point in time for every pair of objects); and

ii) no proper subcategory of C satisfies condition (i).

A category C is the range of R if:

i) whenever xRy then y belongs to C (at every point in time for every pair of objects); and

ii) no proper subcategory of C satisfies condition (i).

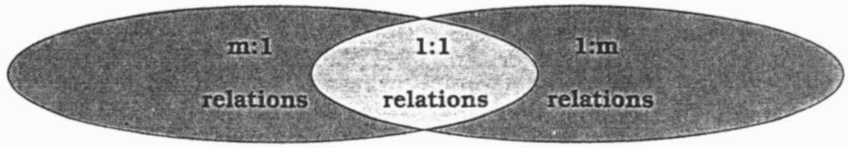


Figure 2: The m:m relations

For example, the domain of *Account-Name* is the category *Account* and its range is the category *String*. The domain of *Serves* is *Waiter* and the range is *Table*.

◆ *Total binary relation* - A relation R whose domain is C is total if at all times for every object x in C there exists an object y such that xRy. (At different times, different objects y may be related to a given object x)

Note: Most relations are not total on their domains. For example, though the domain of the relation *Birth-Date* is the category *Person*, the date of birth of some relevant persons is irrelevant or unknown. Thus, the relation *Birth-Date* is not total.

4) Non-binary Relationships

◆ *Non-binary relationships* - real-world relationships that bind more than two objects in different roles. For example, there is a relationship between a waiter, an account and a shift during which the waiter works for the account.

Such complex relationships are regarded in the SBM as groups of several simple relationships.

For example, the non-binary relationship of the previous example is represented in the SBM by a fourth object, a party and three binary relations between the party and the waiter and the account.

In general, the SBM represents any non-binary relation as:

i) An abstract category of events. Each event symbolises the existence of a relationship between a group of objects.

ii) Functional binary relations, whose domain is the category (i). Each of those functional binary relations corresponds to a role played by some objects in the non-binary relation.

Thus, the fact that objects $x_1, \ldots, x_n$ participate in an n-ary relation R in roles $R1, \ldots, Rn$, is represented by:

- an object e in the category R',

- binary relationships $eR1x1, \ldots, eRnxn$.

5) Instantaneous Databases

◆ *Formal representation of an instantaneous binary database* - as a set of facts, unary and binary.

◆ *Unary fact* - a statement that a certain abstract object belongs to a certain category. For example, (The person whose name is "Jane Howards") is a *Patron*.

◆ *Binary fact* - a statement that there is a certain relationship between two given objects. For example, the *Birth-Date* of (the person whose name is "Jane Howards") is 1968, and (the waiter whose name is "John Smith") *Serves* (the table whose name is "Presidential").

Note: In order to be in the current instantaneous database, the fact must have been explicitly or implicitly entered at some time and never cancelled since.

6) Semantic Binary Schemas

◆ *Semantic Binary Model* - A data structure generating all the binary instantaneous databases.

◆ *Semantic Binary schema* is a description of the names and the properties of all the categories and the binary relations existing in an application's world.

All the instantaneous databases under the schema should have only those categories and relations listed in the schema. The sets of pairs of objects corresponding, in the instantaneous database, to the categories, and the sets of pairs of objects corresponding to the relations, should satisfy the properties indicated in the schema.

The schema should list the following properties of the categories and relations: the sub-categories, the domains and ranges of the relations and the types of the relations (proper m:m, proper m:1, proper 1:m, 1:1)

Dr Naphtali Rishe is an associate professor of computer science at Florida International University. He has extensive experience in database applications and database systems in the industry and has chaired the steering and program committees of the PARBASE-90 conference. Dr Rishe obtained his Ph D in computer science from Tel Aviv University.

Uses of the Model

The SBM can be used as a semantic database management system and as a database design tool in conjunction with conventional DBMSs.

◆ *Semantic DBMS*

We have developed the model as an experimental semantic DBMS *[Rishe-88-EO], [Rishe-88-TM], [Vijaykumar-87], [Jain-87]*. The semantic DBMS is capable of rendering better services to its users than services offered by the relational and other conventional DBMS, in terms of ease of database design, flexibility, ease of application programming in fourth-generation languages and in non-procedural languages and enforcement of integrity constraints. This is not surprising because a system that knows more about the semantics of the user's data can certainly offer better services. In addition, our experimental implementation has shown that the semantic models have potential for much more efficient implementation than the conventional data models. This is due to two reasons:

- All the physical aspects of representation of information by data are user-transparent in the semantic models. This creates greater potential for optimisation: more things may be changed for efficiency considerations without affecting the user programs. This is a continuation of a trend in the relational model. The relational model has more data independence than older models. For example, the order of rows in the tables (relations) is transparent to the user. The semantic models have even more user-transparency.

  For example, the representation of real-world entities by printable values is transparent to the user. One may recall that not long ago the relational model was criticised as less efficient than the network and hierarchical models. However, it is clear now that optimising relational database systems have potential of much higher efficiency than the network and hierarchical systems due to its data independence.

  In semantic models, the system knows more about the meaning of users' data as well as connections between such data. This knowledge can be utilised to organise the data so that meaningful operations can be performed faster at the expense of less meaningful or meaningless operations.

Currently we are working on an experimental massively-parallel database machine to support the SBM *[Rishe-88-AM]*.

◆ *Use in logical design of conventional databases*

We have developed a methodology for logical design of relational, network and hierarchical schemas and integrity constrains using semantic binary schemas *[Rishe-88-DDF], [Rishe-88-MT]*. This is a top-down methodology. In this methodology, a conceptual description of an enterprise is designed using a SBM. Then this description is converted into the relational database design. The result is a high-quality conventional database schema and integrity constraints.

We have also developed a tool which automates all the busy work of the methodology and provides graphic output *[Rishe-88-MT], [Jain-87]*. With respect to the intelligent design decisions, the tool accepts instructions from its user who is a database designer, or, when the user defaults, makes decisions itself based on "rule-of-thumb" principles. ■

The author gratefully acknowledges the advice of Li Qiang, Nagarajan Prabhakaran, Doron Tal and David Barton.

Bibliography

[Abiteboul-84] S Abiteboul and R Hull. "IFO: A Formal Semantic Database Model", proceedings of ACM SIGACT-SIGMOD Symposium on Principles of Database Systems.

[Abrial-74] J R Abrial, "Data Semantics", in J W Klimbie and K L Koffeman (eds.), Data Base Management, North Holland.

[Bracchi-76] Bracchi, G, Paolini, P, Pelagatti, G. "Binary Logical Associations in Data Modelings". In G M Nijssen (ed.), Modeling in Data Base Management Systems. IFIP Working Conference on Modeling in DBMS's.

[Chan-82] Chan, A, Danberg, Sy, Fox, S, Lin, W-T K, Nori, A, and Ries, D R "Storage and Access Structures to Support a Semantic Data Model". Proceedings of the Eighth International Conference on Very Large Data Bases. IEEE Computer Society Press.

[Chen-76] P Chen, "The Entity-relationship Model: Toward a unified view of data." ACM Trans Database Syst1, 1, 9-36.

[Hammer-81] M Hammer and D McLeod, "Database Description with SDM: A Semantic Database Model", ACM Transactions on Database Systems, Vol 6, No 3, pp 351-386.

[Jagannathan-88] D Jagannathan, R L Guck, B L Fritchman, J P Thompson, D M Tolbert. "SIM: A Database System based on Semantic Model". Proceedings of SIGMOD International Conference on Management of Data. Chicago, June 1988. ACM-Press.

[Jain-87] A Jain, Design of a Binary Model Based DBMS and Conversion of Binary Model Based Schema to an Equivalent Schema in other major Database Models. M S Thesis, University of California, Santa Barbara.

[King-84] R King, "SEMBASE: A Semantic DBMS". Proceedings of the First Workshop on Expert Database Systems. University of South Carolina. (pp 151-171)

[Leung-87] C M R Leung and G M Nijssen. From a NIAM Conceptual Schema into the Optimal SQL Relational Database Schema, Australian Computer Journal, Vol 19, No 2.

[Nijssen-81] G M Nijssen "An architecture for knowledge base systems". Proceedings of SPOT-2 conference, Stockholm.

[Nijssen-82] G M A Nijssen and J Van Bekkum, "NIAM – An Information Analysis Method", in Information Systems Design Methodologies: A Comparative Review, T W Olle, et al (eds), IFIP, North-Holland.

[Nixon-87] B Nixon, L Chung, I Lauzen, A Borgida and M Stanley, "Implementation of a compiler for a semantic data model: Experience with Taxis." Proceedings of ACM SIGMOD Conference, San Francisco.

[Rishe-88-AM] N Rishe, D Tal and Q Li, "Architecture for a Massively Parallel Database Machine" Microprocessing and Microprogramming. The Euromicro.

[Rishe-88-DDF] N Rishe, "Database Design Fundamentals: A Structured Introduction to Databases and a Structured Database Design Methodology". Prentice Hall, Englewood Cliffs, NJ.

[[Rishe-88-EO] N Rishe, "Efficient Organization of Semantic Databases". Submitted to *Information Systems*.

[Rishe-88-MT] N Rishe, "A Methodology and Tool for Top-down Relational Database Design". Submitted to *Knowledge and Data Engineering*.

[Rishe-88-TM] N Rishe, "Transaction-management System in a Fourth Generation Language for Semantic Databases". ISMM International Conference on Mini and Microcomputers.

[Shipman-81] D W Shipman, "The Functional Data Model and the Data Language DAPLEX". ACM Transactions on Database Systems, Vol 6, no 1, 140-173.

[Tsur-84] S Tsur, C Zaniolo, "An implementation of GEM – supporting a semantic data model on a relational backend". Proceedings of ACM SIGMOD International Conference on Management of Data, May 1984.

[Vijaykumar-87] N Vijaykumar, "Toward the Implementation of a DBMS based on the Semantic Binary Model". M S Thesis, University of California, Santa Barbara.