A Conceptual Modeling Technique Based on Semantically Complete Model, its Applications

Vladimir Ovchinnikov
Lipetsk State Technical University
Russia, 398024, Lipetsk, Victory St., 104-44
ovch@lipetsk.ru

Abstract: The article proposes a conceptual modeling technique based on Semantically Complete model, the main property of which is that relations carry complete information about interconnection of underlying object types. The paper describes consequences of this property; the most important one is the possibility to use a model relation without its proper name. The author introduces Semantically Complete Query Language, expressions of which is formulated without relation names, describes a closure mechanism allowing to request an interconnection of indirectly related object types by means of a simple enumeration of them. The technique is applied to conceptual modeling of a metallurgical enterprise Manufacturing Execution System as an example. The article proposes use of Semantically Complete model as a unified data access interface to distributed data stored in systems of different architectures and placements, introduces a conception of Semantic Browser based on this interface. The browser permits to view and modify data, has an embedded mechanism of semantic navigation between points of view on information, and a mechanism of creation and publication of new view points by users.

1 Introduction

Conceptual modeling plays the key role in high-quality and effective designing of current complex information systems. Development of conceptual modeling techniques proceeds to abstracting from system implementation details, that is expressed by the conceptualization principle [Gr82].

Evolution of data modeling techniques originates from Relational Model [Co71]. The model allows to abstract from details of relation implementation within DBMS, but concepts of application domain (AD) have no entire reflection: they can be represented as different attributes that can have different names. A fact that certain attributes belong to one domain does not mean that they represent one concept.

As a result of tendency to abstracting, ER modeling technique [Ch76, Ch81] was created. It contains AD concepts in an explicit form, but in two ways: as entities and as attributes. During creation of an ER model one should group attributes to entities, thereby deciding about application system implementation.

By further moving on the conceptualization way, Object-Role model was developed [Br95, Ha95], that evolved from the binary modeling technique [NH89] to the technique

generalizing [Br94] the majority of known conceptual modeling techniques [BHW91, HO92, HW93, NH89, Tr91], including ER. An object type (entity) of ORM reflects an AD concept directly. Unlike ER model, this entity is not characterized by an attribute set; all attributes are considered to be independent entities that can have their own connections (fact types) with other entities. Fact types of ORM do not carry complete information about interconnection of object types: knowing about existence of a fact type and its semantics, you can not assert that you know interconnection of underlying object types.

Moving forward, we propose Semantically Complete model, relations of which describe interconnection of underlying object types in the complete way [Ov03, Ov04]. Semantically Complete model may be considered as restricting of Object-Role one, therefore, we shall use its terminology below when discussing Semantically Complete model.

2   Semantically Complete Model

The property of relations to describe interconnection of object types in the complete way is named as semantic completeness, and a relation complying with this property is referred to as semantically complete one. Some constraints are imposed on a semantically complete relation to attain to the property. The first one lies in banning inclusion of an object type to a relation more than once.

Definition 1. A semantically complete relation is a relation that is based on object types, each of which is contained in the relation only once. In other words, a semantically complete relation is built on an object type set.

The second constraint lies in prohibition of alternative relations existence within a semantically complete model. Two semantically complete relations are considered to be alternative if one of them is based on an object type set that is nonstrict subset of another set underlying the second relation.

Definition 2. A semantically complete model is a combination of two sets: a set of semantically complete relations without alternative ones, and a set of constraints based on the relations.As a result, each relation of a semantically complete model describes interconnection of object types completely. In other words, within a model, there are no relations describing the interconnection in different ways. It is necessary to underline that we mean interconnection of all object types of a relation. If two relations have more than one shared object types, they can describe interconnection of these object types differently. But there can not be a relation built on the shared object types only, i.e., these object types have no independent interconnection and just are a decomposition of a concept on several object types.

An important consequence of semantic completeness is possibility to refer to a relation without its proper name. Indeed, since a model has no alternative relations, object type sets are identifiers of appropriate relations. Therefore, a reference to a relation can be

fulfilled with an object type set, for instance, as (Unit, Produced Material Piece). As a result, it is not necessary to assign proper names to relations within a semantically complete model, relations are defined by enumerations of underlying object types.

An important aspect of conceptual modeling is the way of constraint definition. Since the proposed model has essential distinctions from existent ones, we introduce the way taking into account its features. Let us begin from the base constraints: functional, equal, mandatory, and fullness. These constraints are present in all conceptual modeling techniques with one name or another.

Let exist a relation R based on the object types: $R : (A_1,...,A_n, B_1,...,B_m)$.

Then:

- The functional constraint $A_1,...,A_n \rightarrow B_1,...,B_m$ implies the predicate $\forall a \in \pi_{A_1,...,A_n} R \exists_1 b \in \pi_{B_1,...,B_m} R \{(a,b) \in R\}$ stating that any existent instance combination of the types $A_1,...,A_n$ corresponds to only one instance combination of the types $B_1,...,B_m$.

- The equal constraint $A_1,...,A_n \equiv B_1,...,B_m$ implies the predicate $\forall a \in \pi_{A_1,...,A_n} R \exists_1 b \in \pi_{B_1,...,B_m} R \{(a,b) \in R\} \wedge \forall b \in \pi_{B_1,...,B_m} R \exists_1 a \in \pi_{A_1,...,A_n} R \{(a,b) \in R\}$ stating that any existent instance combination of the types $A_1,...,A_n$ corresponds to only one instance combination of the types $B_1,...,B_m$ and vice versa. The given constraint is equivalent to two opposite functional constraints: $A_1,...,A_n \rightarrow B_1,...,B_m \wedge B_1,...,B_m \rightarrow A_1,...,A_n$.

- The mandatory constraint $(\underline{A_1,...,A_n}, B_1,...,B_m)$ implies the predicate $\forall a \in \pi_{A_1,...,A_n} R \exists b \in \pi_{B_1,...,B_m} R \{(a,b) \in R\}$ stating that any existent instance combination of the types $A_1,...,A_n$ corresponds to at least one instance combination of the types $B_1,...,B_m$. At that, if a model contains other relations including the object types $A_1,...,A_n$, then the part $\forall a \in \pi_{A_1,...,A_n}$ of the previous expression should be modified with the aim of covering all instance combinations of the object types $A_1,...,A_n$ that occur in those relations.

- The fullness constraint $(|A_1,...,A_n|, B_1,...,B_m)$ implies the predicate $\forall a \in \pi_{A_1,...,A_n} R \{a_1 \neq \varnothing \wedge ... \wedge a_n \neq \varnothing\}$ stating that all existent instance combinations of the types $A_1,...,A_n$ contains non-empty instances for all these types. The author inclines to put hard fullness constraint on all relations of any semantically complete model. This aspect will be discussed later.

To describe more complex constraints, we use a query language that is presented below. The enumerated constraints can be imposed not only on model relations but also on relations calculated with a query.

3 Semantically Complete Query Language

Expressions of any query language represent a tree of relation transformations. Now we define the structure of Semantically Complete Query Language expression. Owing to particular properties of Semantically Complete model, the structure differs from existent query languages noticeably.

Let us define a calculable relation according to its usage in the proposed language. For that, we introduce a concept "role index" as a positive integer. A calculable relation has no constraints that are similar to constraints imposed on a semantically complete relation, therefore, one object type can be included in a relation many times. Let us consider a fact of inclusion of an object type to a relation as a relation position.

Definition 3. A calculable relation is a relation built on object types, each position of which is assigned to a role index; at that, one object type is not included in the relation with the same role index more than once.

Each calculable relation has a representation which describes the way of its calculation. A representation can be based on calculable relations as well as on semantically complete relations, thereby forming a tree of resulting relation calculation.

Definition 4. A representation is a mapping based on states of calculable or semantically complete relations; it functionally determines a state of a calculable relation depending on states of base relations.

Let us define main representations of the proposed query language. A positioning is the first one and is absent in other languages. Its existence flows from Semantically Complete model features. Just to simplify the following definition, we introduce a concept "role object type" as a combination of an object type and a corresponding role index.

Definition 5. A positioning is a representation that repeats a base relation with possible changes of object types role indexes. At that, the following is true:
- A positioning is built on one base relation.
- There is a bijection between role object types of base and calculable relations. A constraint that corresponding role object types must be assigned to the same object type is imposed on the bijection.
- A state of a calculable relation consists of the same relation instances (rows) as a state of a base relation, with the difference that object instances are assigned to role object types according to the bijection.

A positioning allows to use one object type in different roles, that simplifies an expression notation. Any expression formulated with a positioning can be reduced to the form without it.

A projection of the proposed language practically does not differ from well known one, except that it is based on role object types.

Properties of Semantically Complete model have a dramatic influence on the next representation that is the composition. A composition is similar to a natural join of relational algebra, but a join criterion is not attribute name coincidence, it is coincidence of role object types.

Definition 6. A composition is a representation being a superposition of base relations on role object types.

The special role of composition becomes apparent when its properties are combined with the property of Semantically Complete model to refer to relations without their proper names. The first consequence of this combining is that notation of composition of binary relations is reduced to a chain of object types. For instance, a composition of the relations (Unit, Produced Material Piece), (Produced Material Piece, Material Piece), (Material Piece, Production Order) can be described by

(Unit–Produced Material Piece–Material Piece–Production Order).

A composition of nonbinary relations is simply described by enumeration of relations with comma.

The second consequence is that apparent describing of a relation join criterion, as it is accepted in SQL and other query languages, is not necessary. A relation superposition is always fulfilled according to coincidence of role object types.

And the third consequence: it is not necessary to consider a selection as an independent representation. Actually, a selection is a composition of two relations, one of which is defined on a condition clause (on the logical level). As a result of such a composition, only relation instances complying with a condition is selected from the first relation:

((Material Piece–Thickness), (Thickness > 0.5)) or

(Material Piece–(Thickness > 0.5)).

Union and minus of the proposed query language are similar to those accepted in relational algebra, but matching of relation positions is defined by coincidence of role object types, and not in the manual way. Therefore, equality of relations' arities becomes unnecessary; relations can have different sets of role object types; it is not necessary to take care of appropriate allocation of object types to relation positions.

4 Default Closure of Semantically Complete Model

When one formulates a statement, he often considers indirectly connected concepts as if they are connected directly, omitting intermediate steps. Semantically Complete model has a mechanism allowing to request a connection between indirectly connected object types, not enumerating all used relations. This is a default closure.

Definition 7. A default closure is a nonstrict subset of a relation set of a semantically complete model.

A calculable relation describing interconnection of indirectly connected object types can be requested by simple enumeration of the object types, without formulating an

appropriate relation composition in the apparent way. For instance, if the relations (Slab, Material Piece), (Material Piece, Production Order) are included in a default closure, the calculable relation describing interconnection of the object types Slab and Production Order can be requested as

<div align="center">(Slab, Production Order), or</div>

<div align="center">(Slab–Production Order)</div>

that is outwardly indistinguishable from a reference to a semantically complete relation.

To attain such property, a constraint stating that there are no alternative paths between object types within a default closure is imposed. A path implies a composition of certain connected relations. Two paths are considered to be alternative if they have at least two common object types included in different semantically complete relations.

An algorithm of calculation of a relation based on a set of indirectly connected object types can be represented using a default closure as follows:
- decompose a closure on subsets of connected relations;
- check that all requested object types pertain to one subset;
- execute a composition of necessary relations of this subset;
- execute a projection of the composition result on the requested object types.

As you see, the default closure property are based on the composition properties. The first consequence of the closure property is simplification of query formulation: to request interconnection of indirectly connected object types, it is not necessary to define a composition of relations in the apparent way. The second one is high expandability since a change of a model structure does not result in changes of elaborated queries if a closure structure is retained.

5 Semantically Complete Modeling Technique Notation

To represent a semantically complete model graphically, one may use the ORM graphical notation. But complex constraints can not be defined in this way. Thereto, we propose to use a textual notation. In addition, the textual notation has a structure that is close to the structure of natural languages phrases, and is clear to AD experts to a greater extent.

Semantically Complete model textual notation is a set of phrases being clear to AD experts. Each phrase is a statement about existence of a certain semantically complete relation. For example, the phrase "A Material Piece is produced on a Unit" states about existence of the object types "Material Piece" and "Unit", and about existence of the semantically complete relation (Material Piece, Unit) within the model.

Within the proposed technique, it is important to name object types in accordance with their semantics from the point of view of experts. In this case, a designer has not to switch between two term rows: terms for thought expression and terms for modeling; both rows are joined to one used in both processes. In such interpretation, an object type and a AD concept is the same. Therefore, we use the conception "concept" as a synonym

of the conception "object type" in the framework of Semantically Complete model [Ov04].

Base constraints can be defined in a phrase itself as well as below a phrase in square brackets with an indent. For example, in the following way:
<u>A Produced Material Piece</u> is a Material Piece
      [Produced Material Piece $\equiv$ Material Piece]
A Produced Material Piece is produced on a Unit
      [Produced Material Piece $\rightarrow$ Unit]
      We took the following constraint designations, reasoning from intuitive clarity:
- a functional constraint – in square brackets below a phrase with the symbol $\rightarrow$;
- an equal constraint – in square brackets below a phrase with the symbol $\equiv$;
- a mandatory constraint – by underlining of an appropriate object type in a phrase;
- fullness constraint is considered to be in force if a concept is not bounded by vertical lines, for instance, in the phrase "A Produced Material Piece is produced on a |Unit|" the concept "Produced Material Piece" has a fullness constraint, and the concept "Unit" has no it.

The proposed textual notation allows to represent a model in the way that is transparent for an AD expert not being a specialist in the field of information system designing. Strict formal basis of the model harmonizes with easy readable representation.

Constraints applied to calculable relations are described in square brackets in the following order: a query and, after semicolon, a constraint on a result relation.

Let us describe a notation of Semantically Complete Query Language. A positioning of a semantically complete relation is defined like a reference to the relation, but a role index of an object type within the relation is indicated in round brackets after an object type name. For instance, (A(1), B(2), C(1)) is the positioning of the relation (A, B, C). Absence of a role index is considered as the role index 1.

A composition of relations can be defined in two ways. The first one lies in enumerating relations with comma in round brackets. For instance, ((A, B), (B, C)) is the composition of the relations (A, B) and (B, C). This way is applicable to relations of any arity. The second way is applicable to binary relations only and lies in describing a composition with the symbol '–' in round brackets. In this case, a composition is fulfilled on those binary relations which are formed by pairs of role object types gathered around a symbol '–'. For instance, (A–B–C) is the composition of the relations (A, B) and (B, C), that is equivalent to the way of describing as ((A, B), (B, C)). This approach allows formulating a composition of binary relations as a chain of role object types; it is compact and intuitively transparent.

A union of two relations is defined in round brackets by means of the keyword "union" or the symbol "U" between the relations: ((A, B) U (B, C)) or ((A, B) union (B, C)). A difference of two relations is defined in round brackets by means of the keyword "minus" or the symbol "/" between the relations: ((A, B) / (B, C)) or ((A, B) minus (B, C))

A relation projection is described by enumerating role object types being projected in round brackets after a projecting relation, via a point. For instance, (A, C, D).(C, D) is the projection of the relation (A, C, D) on the role object types C and D.

A relation defined with a condition is described as a logical clause in round brackets. For instance, ((A, B), (B > C)) is the composition of two relations, one of which is a relation based on a condition.
Inclusion of a semantically complete relation in a default closure is defined with a bold font type of the relation, for instance:

A Produced Material Piece is a Material Piece

[Produced Material Piece ≡ Material Piece]

We shall give an example of a proposed notation application in an appropriate section.

6 Properties of Semantically Complete Modeling Technique

Semantically Complete Modeling Technique is characterized by high conceptualization and relative simplicity of its use due to the following properties:
- relations carry complete information about interconnection of object types;
- relations are requested without their proper names; it allows to analyze a model considering object types only;
- model development and discussion are performed in the same terms;
- query formulation is performed without relations' proper names;
- a composition of binary relations is represented as a chain of object types;
- a general criterion of relation position matching holds when executing composition, union, or difference; it is not necessary to indicate the criterion in an explicit form;
- a selection is a particular case of a composition;
- a request of interconnection of indirectly connected object types, being included in a default closure, is represented as an enumeration of object types.

We separately underline the Semantically Complete model property to expand without changes of created queries. It follows from the default closure mechanism and non-composite semantics of relations.

Let us enumerate a row of additional properties following from Semantically Complete model constraints:
- If we draw an analogy with Relational model, a semantically complete model is in the 5NF when its relations are elementary to an extent that they can not be decomposed on two or more relations. Relations of a model is recommended to create just in such elementary way since it ensures the maximal expandability and clearness.
- If the previous property holds, an additional prohibition of empty values existence may be imposed on the model since relations have non-composite semantics and, therefore, empty values may be substituted by relation instance absence, within a relation state.

- If empty values are prohibited, a model state is represented as a set of relation instances (cohesions), where a cohesion is a set of pairs (a value, an object type). This property follows from that a semantically complete relation can be determined by an object types set of a cohesion easily. It is not necessary to associate a cohesion to a relation in an explicit form.
- Semantic completeness of relations, especially absence of alternative relations, causes simplification of queries by navigation creation.
- Constraints of Semantically Complete model result in decrease of equivalent representations quantity of one AD.

7   An Example of Semantically Complete Modeling Application

Consider a MES (Manufacturing Execution System) as applied to plate rolling, namely the system part tracking actual production, as an example of a proposed technique application. The peculiarity of a metallurgical manufacture is that making of a final product can not be represented as a hierarchy BOM (Bill of Materials) since a material piece can be cut and welded in an arbitrary way during processing. The key requirement made to such systems is the following: a system should contain information about all existent material pieces (raw, intermediate or final) and a full history of their processing.
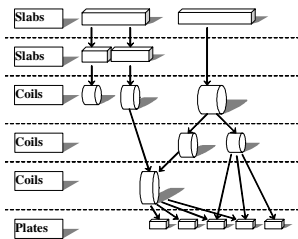

Fig. 1. Transformation of Material Pieces During Processing

Material pieces undergo the following transformations during processing: slabs can be cut, coils can be welded or cut, and plate packs can be collected from different coils (figure 1). Transformations occur on units, at that, one material piece is produced on a single unit and is consumed for producing other material pieces on another unit.

A semantically complete model of this AD was developed and discussed with experts. A part of the model is shown below:
A Unit produces <u>Produced Material Pieces</u>
      [Produced Material Piece → Unit]
A Unit consumes <u>Consumed Material Pieces</u>
      [Consumed Material Piece → Unit]
A <u>Produced Material Piece</u> is a <u>Material Piece</u>
      [Produced Material Piece ≡ Material Piece]
A <u>Consumed Material Piece</u> is a Material Piece
      [Consumed Material Piece ≡ Material Piece]

A Material Piece is assigned on a Production Order
>   [Material Piece → Production Order]

A <u>Material Piece</u> has a Thickness
>   [Material Piece → Thickness]

A <u>Slab</u> is a Material Piece
>   [Slab ≡ Material Piece]

A <u>Coil</u> is a Material Piece
>   [Coil ≡ Material Piece]

A <u>Pack</u> is a Material Piece
>   [Pack ≡ Material Piece]

A <u>Unit</u> has an Average Capacity
>   [Unit → Average Capacity]

A <u>Material Transformation</u> produces a Produced Material Piece
>   [Material Transformation → Produced Material Piece]

A <u>Material Transformation</u> consumes a Consumed Material Piece
>   [Material Transformation → Consumed Material Piece]

A <u>Material Transformation</u> is characterized by an Input Part Offset
>   [Material Transformation → Input Part Offset]

A <u>Material Transformation</u> is characterized by an Input Part Length
>   [Material Transformation → Input Part Length]

A <u>Material Transformation</u> is characterized by an Output Part Length
>   [Material Transformation → Output Part Length]

A <u>Material Transformation</u> is characterized by an Output Part Number
>   [Material Transformation → Output Part Number]

>   Such AD representation is a good mean for discussing with experts. In our case, it is discovered that the AD has the directed graph of material part transformations during processing.

General form graph is the structure that is complex to represent in information systems. We wittingly chose this example to demonstrate how this problem may be solved with Semantically Complete Modeling. As we can see from the model, the problem is solved by forming concepts that are equivalently connected with a base concept being a node or an arc of the graph. In our case, for the concept "Material Piece", representing graph nodes, we declare two concepts equivalently connected with it: "Produced Material Piece" and "Consumed Material Piece". Incidence of graph arcs, represented by the concept "Material Transformation", and nodes is defined by means of the relation "A <u>Material Transformation</u> produces a Produced Material Piece" for incoming arcs, and by means of the relation "A <u>Material Transformation</u> consumes a Consumed Material Piece" for outcoming arcs.

Let us give the following statement as an example of a complex constraint based on a query: "thicknesses of all material pieces, consumed during producing a given one, should be the same".
>   [(Thickness–Consumed Material Piece–Material Transformation–Produced
>   Material Piece): Produced Material Piece → Thickness]

Note that this formulation of the constraint uses the default closure. Instead of "Thickness–Material Piece–Consumed Material Piece", we have used "Thickness–Consumed Material Piece" since the relations (Material Piece, Thickness) and (Material Piece, Consumed Material Piece) are included in the default closure.

This section shows an application of Semantically Complete Modeling in the context of a metallurgical production, but usability of the technique does not limited to this AD. Semantically Complete model, which may be imagined as restriction of Object-Role one, can be applied to conceptual modeling of any AD as well as other conceptual techniques.

8 Semantically Complete Model as a Data Access Interface

The main destination of Semantically Complete model is conceptual modeling. The properties of the model does not allow to use it as a model of data storage as a result of low storage efficiency. But it can be used as an access interface to data stored with one of known tools, for instance, a RDBMS.

Partitioning of data storage and data interface creates additional expandability since a data storage structure can be changed without changes of data interface in a wide range. Changes of a data storage structure result in adjustment of a structure reflection to an interface only.

Integration of some data servers to an entire environment with an interface in the form of Semantically Complete model allows to ensure the unified data access interface with transparency of places and methods of physical data storage. The last allows developing a unified browser for this distributed information environment, we name it as Semantic Browser, with the following functionality:
-   browsing and modifying of data physically stored in different DBMS and other data storage systems, distributed among different servers;
-   maintaining of conception of view points (pages) stored in a shared distributed repository; points of view contain connected data sources (semantically complete queries) represented to users with components stored in the same distributed repository;
-   easy definition of master-detail dependences between data sources: it is necessary only to indicate which data source is master, and which is detail, no more, due to absence of an explicit join criterion within the language;
-   unified mechanism of filtering of data sources content within points of view, saving of filters in a user context and their reusing; a filter is just another relation being in a composition with a query of a data source;
-   creation and publication of new view points by users;
-   automatic support of navigation between points of view: a user can trigger a transition from any data source to those view points which reflect appropriate object types within their main data source;
-   before a user triggers a transition, he sees a menu of possible transitions divided in the transition classes: in context of one object type instance (from a current cell of a

grid, for example), in context of selected rows or columns, in context of a whole state of a data source;

- after a transition is fulfilled, a main data source of a new view point is filtered according to object types instances being in the transition context.
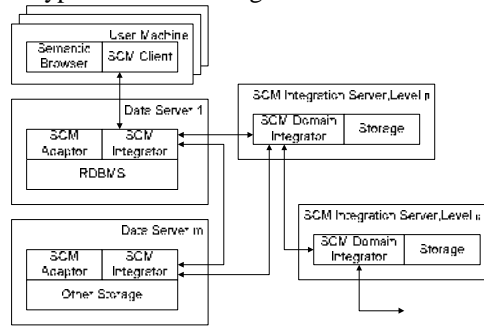

Fig. 2. Architecture of Information Environment Based on Semantically Complete Model

A general architecture of the information environment based on Semantically Complete model is illustrated on the figure 2. The key components of this architecture are:

- SCM Adaptor. It is responsible for organizing a data access interface as a semantically complete model. It holds information about reflection of a relational scheme (or another data representation) to a semantically complete model. It executes expressions of Semantically Complete Query Language and data modifications, performing their translation to SQL (or other data processing languages being executed within the storage), and executing the last.
- DBMS or other storage. It performs base functions of data storing and execution of expressions of an appropriate data processing language.
- SCM Integrator. It fulfills distributed semantically complete queries and data modifications, holds information about distribution of used relations among servers, interacts with a superior SCM Domain Integrator for getting information about distribution of newly used relations.
- SCM Domain Integrator. It holds information about distribution of relations among data servers, transmits unprocessed requests to a superior SCM Domain Integrator.
- SCM Client. It organizes a semantically complete data access interface on a user machine, may execute distributed semantically complete queries and data modifications.
- Semantic Browser. It constructs and shows view points to users by executing appropriate semantically complete queries, translates a user's changes to data modification expressions and executes them, fulfills context transitions to other view points.

Embodiment of the environment to Internet will allow to use features of the proposed approach widely, to hasten development of Internet applications and to increase data integration level.

36

9 Conclusion and Further Work

Thus, Semantically Complete Modeling has ample quantity of distinctive features which, being taken apart, perhaps have no considerable weight, but, being taken together, result in significant simplification and expandability of a conceptual model. The property of Semantically Complete Query Language to interact with a model by means of AD terms only allows to develop fundamentally new tools and applications with Semantically Complete model. Use of Semantically Complete model as a data access interface permits to create a unified environment of distributed data access and navigation. Embodiment of the environment to Internet allows to hasten development of Internet applications and to increase data integration level. At that, semantic orientation of the proposed model and environment plays a special role.

At present time, the author works on the following:
- designing and implementation of Semantically Complete data access interface with RDBMS as a data storage;
- designing and implementation of data integration software of the described environment;
- designing and implementation of Semantic Browser.

Bibliography

[BHW91]   van Bommel, P.; ter Hofstede, A.H.M.; van der Weide: Semantics and verification of object-role models: Information Systems, 16(5), 1991; pp.471-495.

[Br94]   Brouwer, S.J.; Martens, C.L.J.; Bronts, G.H.W.M.; Proper, H.A.: Towards a Unifying Object Role Modelling Approach: Proc. of the First International Conference on Object-Role Modelling (ORM-1), Magnetic Island, Australia, 1994; pp. 259-273.

[Br95]   Bronts, G.H.W.M.; Brouwer, S.J.; Martens, C.L.J.; Proper, H.A.: A Unifying Object Role Modelling Approach: Information Systems, 20(3), 1995; pp. 213-235.

[Ch76]   Chen, P.P.S.: The entity-relationship model – towards a unified view of data: ACM Transactions on Database Systems, 1(1), 1976; pp. 9-36.

[Ch81]   Chen, P.P.S.: A Preliminary Framework for Entity-Relationship Models: Entity-Relationship Approach to Information Modeling and Analysis, Saugus, Calif., 1981.

[Co71]   Codd, E. F.: Normalized Data Base Structure: A Brief Tutorial: Proc. 1971 ACM SIGFIDET Workshop on Data Description, Access, and Control, San Diego, Calif., 1971.

[Gr82]   van Griethuysen, J.J., editor: Concepts and Terminology for the Conceptual Scheme and the Information Base. Publ. nr. ISO/TC97/SC5-N695, 1982.

[Ha95]   Halpin, T.A.: Conceptual Schema and Relational Database Design: Prentice-Hall, Sydney, Australia, 2nd edition, 1995.

[HO92]   Halpin, T.A.; Orlowska, M.E.: Fact-Oriented Modelling for Data Analysis: Journal of Information Systems, 2(2), 1992; pp. 1-23.

[HW93]   ter Hofstede, A.H.M.; van der Weide: Expressiveness in conceptual data modeling: Data & Knowledge Engineering, 10(1), 1993; pp. 65-100.

[NH89]   Nijssen, G.M., Halpin, T.A.: Conceptual Schema and Relational Database Design: a fact oriented approach. Prentice-Hall, Sydney, Australia, 1989.

[Ov03]   Ovchinnikov, V.V.: A Conceptual Modeling Technique without Redundant Structural Elements: Journal of Conceptual Modeling (www.inconcept.com/jcm), 29, 2003.

[Ov04]   Ovchinnikov, V.V.: Improving Controllability of Vast Conceptual Models: Journal of Conceptual Modeling (www.inconcept.com/jcm), 31, 2004.

[Tr91]   de Troyer, O.M.F.: The OO-Binary Relationship Model: A Truly Object Oriented Conceptual Model.: Proc. of the Third International Conference CaiSE'91 on Advanced Information Systems Engineering, vol. 498 of Lecture Notes in Computer Science, Trondheim, Norway, 1991; pp. 561-578.