The Multi-Meta Database Model as a Foundation for Schema Reuse in Conceptual Database Design

Mirta Baranović, Slaven Zakošek, Ljiljana Brkić

***Abstract*—** we are proposing a multi-meta model that serves as a foundation for schema reuse in conceptual database design. Multi-meta model is based on entity-relationship model. According to the proposed model corresponding database, called multi-meta database is built. Multi-meta database contains descriptions of the existing database models and serves as a basis for design of new models. To avoid ambiguity regarding the model descriptions the redundancy has been maximally reduced without loss of any existing knowledge. The process of the design of a new database model can cause the restructuring of existing knowledge contained in the multi-meta database. In the proposed non-redundant model the restructuring is simplified, while the consistencies are preserved. The necessary conditions that have to be satisfied by each database model are defined. In case that these conditions are not met, the described procedures for the model restructuring have to be applied.

***Keywords*—**Conceptual modeling, database design, meta data, schema reuse.

I. Introduction

In design of a database model, the principal goal is to satisfy the existing and the future user requirements. The design has to satisfy the usual requirements for data modelling, such as integrity, consistency and non-redundancy. In addition, a special requirement arises - the stability. The stability in this context implies that the existing data structures should be insensitive to changes, which normally affect the database and the corresponding information system throughout its life-cycle. The data model may be enhanced with new elements and new relationships among the elements, but these extensions should not endanger the existing elements and relationships.

To transform the perception of the real world into a conceptual database design is the most sensitive task in development of an information system. The quality of the result heavily depends upon the expertise, intuition and experience of the designer and upon the quality of communication with the users. The final quality of the data model and its stability are a direct consequence of this task. Very often, the designer creates new objects by copying or modifying the already existing ones, used in other databases. The reuse of these properly designed and in practice confirmed elements is surely desirable [9].

A software tool should help the designer to appropriately select and reuse some existing database parts [5] and to enhance them easily with new creations.

The multi-meta database contains descriptions of the existing database models. It enables an easy retrieval and comparison of the elements stemming from different models. Single models contained in the multi-meta base are represented in a form of an entity-relationship model [10].

In the multi-meta base concept, a single element may belong to multiple databases. It makes the difference to the meta-models applied in semantic integration of heterogeneous databases as reported in [3], [1], where each object belongs strictly to a single scheme. The intention of the proposed method is to stimulate the reuse, so the expectation is that the count of the database schemes sharing the same element should steadily grow.

Descriptions of the models contained in the multi-meta base must be non-redundant. For example, it would not be allowed to describe the connections among the entities concurrently through the definitions of relationships and through foreign keys contained in the entity definitions. A non-redundant description will make the retrieval and comparison easier and it will exclude ambiguity from the model description.

The method based on the multi-meta base [6] detects the semantic and structural similarity [8] among the existing models. The results are applied to remove any duplicates and to define the semantic connections among elements. New models are built from a selection of the already existing ones, new creations are added, model consistency is checked and some restructuring performed, when necessary. The schemes must be diluted into atomic parts to make the desired selections easier.

Manuscript received May 4, 2012. This research is supported by the "Semantic Integration of Heterogeneous Data Sources" project, funded by the Croatian Ministry of Science under grant number 036-0361983-2012.

Mirta Baranović is with the University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb, HR-10000 Croatia (corresponding author to provide phone: +385 1 6129 916; fax: +385 1 6129 915; e-mail: mirta.baranovic@fer.hr).

Slaven Zakošek is with the University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb, HR-10000 Croatia (e-mail: slaven.zakosek@fer.hr).

Ljiljana Brkić is with the University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb, HR-10000 Croatia (e-mail: ljiljana.brkic@fer.hr).

II. MULTI-META DATABASE MODEL

A. Basic model of the meta-base

A model of the meta-base is defined that will contain the description of a database. The database model description consists of descriptions of entities, relationships, attributes and domains. The basic model of the meta-base is presented in Fig 1.

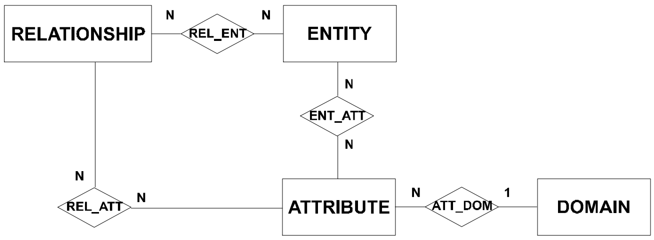

Fig. 1 meta model

The entities contain the respective descriptions of database components; i.e. ENTITY contains descriptions of the entities etc., as follows:

ENTITY = {EntID, EntName, EntLongName, EntDescr, EntComment}

RELATIONSHIP = {RelID, RelName, RelLongName, RelDescr, RelComment}

ATTRIBUTE = {AtrID, AtrName, AtrLongName, AtrDescr, AtrComment}

DOMAIN = {DomID, DomName, DomLongname, DomDescr, DomComment, DomType}

The relationships REL_ENT, REL_ATT, ENT_ATT, ATT_DOM describe the respective connections among the meta-base entities:

REL_ENT = {RelID, EntID, RoleName, Connectivity, WeakEnt}

REL_ATT = {RelID, AttID, RelKeyPart }

ENT_ATT ={EntID, AttID, EntKeyPart}

ATT_DOM = {AttID, DomID}

Where:

The connectivity of a relationship specifies the mapping of the associated entity occurences in the relationship [4].

RoleName introduces different roles of an entity in a relationship, what is of special importance in reflexive and in parallel relationships.

RelKeyPart marks an attribute as part of the relationship primary key. The allowed values are TRUE or FALSE.

EntKeyPart marks an attribute as part of the entity primary key. The allowed values are TRUE or FALSE.

1) Transformation of the meta-base

All the entities in the meta-base share a nearly equal structure. For a model analysis it is necessary to process all the meta-base entities, regardless to their type or meaning within the model. Therefore, a generalized entity OBJECT is introduced having the entities RELATIONSHIP, ENTITY, ATTRIBUTE, DOMAIN as specializations. The resulting model is presented in Fig. 2.

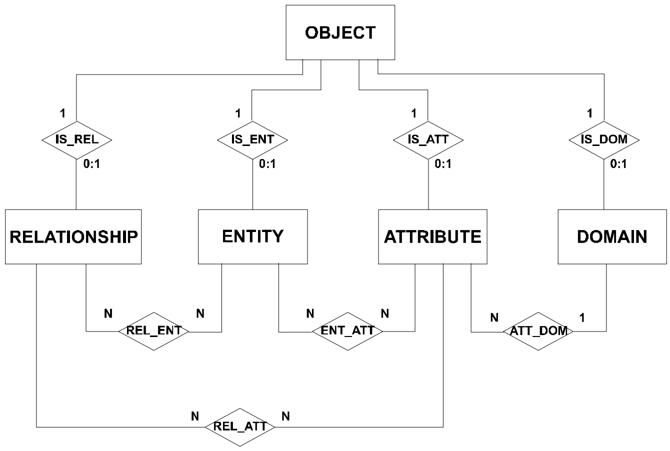

Fig. 2 meta-base model with generalization/specializations

For the generalization, new attributes are introduced: ObjType to enable partitioning and ObjID, as the object identifier. Relational schemes of the entities become:

OBJECT = {ObjID, ObjName, ObjLongName, ObjDescr, ObjComment, ObjType}

ENTITY = {EntID, ObjID}

RELATIONSHIP = {RelID, ObjID}

ATTRIBUTE = {AtrID, ObjID }

DOMAIN = {DomID, ObjID, DomType}

B. Multi-meta model and multi-meta database

The database that will contain the descriptions of different database models is the multi meta-base, in further text referred to as the **mm** base. The information contained in the multi-meta base will continuously be analyzed, deductions will be carried out and it might be revised. Design of a new model starts with the selection of structures from the existing models described in the **mm** base. These structures may be subject to revision. The model can be enhanced further on. After the completion of these procedures, the consistency of the new model is verified. The **mm** base is proposed to enable:

- storing information about single database models
- deductions, made simple as possible, about similarity among the objects from different data models
- an easy structure revision of existing models, i.e. the revision of data describing the models
- a review of the existing models
- an easy choice of concepts from the existing model to help in the design of a new model

To meet the listed objectives, the redundancy should be reduced.

At the time when a multi-meta model is built, it will be attempted to substitute the concepts from a lower level of abstraction with those concepts from a higher level, which already imply some general rules. In this way, the redundancy will be reduced, the model clarity will be increased, maintenance made easier and the model analysis simplified.

The simplest way to obtain a meta model containing the elements from different schemes is to extend it into a new dimension - the database. Such extension means that every object in the meta-model will obtain an additional key attribute - the identifier of its home database. This model is presented in Fig. 3.

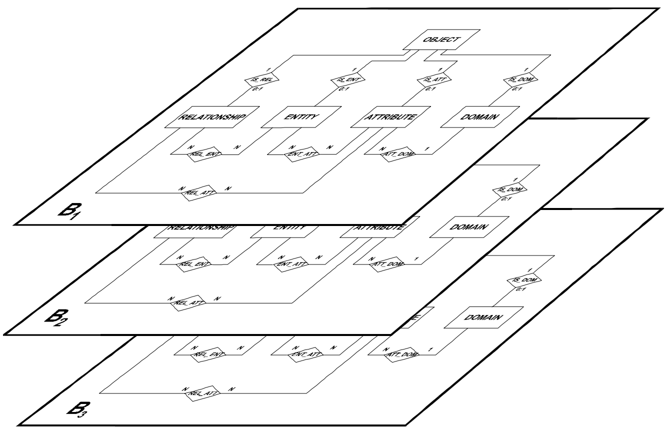

Fig. 3 a possible form of the mm base model

The model in Fig. 3 bears a large redundancy. If an entity, together with its attributes is present in a number of different schemes, it would be described in each of these schemes. This would complicate the schema maintenance and revisions.

By analysing of single schemes, connections will be established among the objects from different planes - databases. At the instance when a connection between the objects from two different databases is established, new objects will be stored redundantly, in each of their home databases.

During the mm model design, the aim is a maximum reduction of redundancy, to make the frequent scheme revisions easy.

The redundancy can be reduced by application of the following rules:

Rule 1

The objects, contained in different databases, are stored independently from their home databases.

Consequence of the Rule 1

• a new relationship is established to connect the objects to their corresponding databases (BASE_OBJ).

Rule 2

A single object, uniquely defined by its identifier, has certain exactly determined semantic and structural properties. There exist no alternative structural properties for any object.

Consequence of the Rule 2

• the relationships contained in REL_ENT, REL_ATT, ENT_ATT and ATT_DOM are independent of their home databases.

This may mean, for example:

• to a given attribute, the same domain is attached in all the databases where this attribute is present

• a given entity has a uniquely defined relational scheme, equal for all the database models where it is present

• a given relationship has a uniquely defined structure (entities involved, the mapping, the relationship attributes) in all the database models where it is present.

The model of the mm base, founded upon the rules 1 and 2 is presented in Fig. 4.

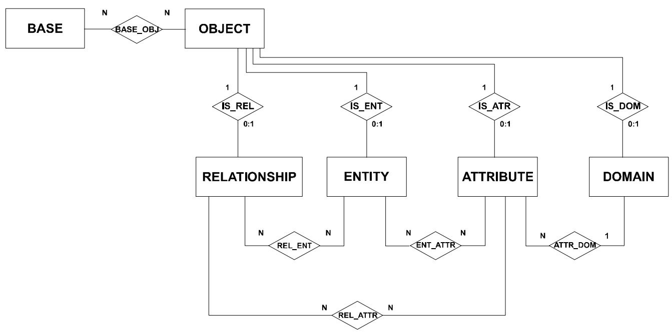

Fig. 4 model of the mm base, founded on the rules 1 and 2

For the scope of further redundancy reduction, the interconnection of objects to the databases is considered.

Thanks to the rules 1 and 2, and because all the objects of a database are interconnected, a question arises whether all the database objects have to be explicitly connected to the database. One can suppose that connections to only certain objects are essential, whereby all the other connections can be deduced.

On the ground of an additional analysis of the multi-meta model, some facts have been found and they will be elaborated in some definitions that follow.

Let a scheme **MM** (or model **MM**) correspond to the model in Fig. 4 and let a database **mm** represent the current value of **MM**. Let **mm** contains different database descriptions stored in the following relations defined over their respecive relational schemes:

$\mathcal{B}$ BASE	$\mathcal{BO}$ BASE_OBJ
$\mathcal{O}$ OBJECT	$\mathcal{RE}$ REL_ENT
$\mathcal{R}$ RELATIONSHIP	$\mathcal{RA}$ REL_ATT
$\mathcal{E}$ ENTITY	$\mathcal{EA}$ ENT_ATT
$\mathcal{A}$ ATTRIBUTE	$\mathcal{AD}$ ATT_DOM
$\mathcal{D}$ DOMAIN	

Abbreviations are introduced for attribute names from the model **MM**:

#B database identifier (baseID)
#O object identifier (objID)
#R relationship identifier (relID)
#E entity identifier (entID)
#A attribute identifier (attID)
#D domain identifier (domID)

Definition 1

Let the set $\mathbf{R_B}$ members be the identifiers of all the relationships contained in the database **B**.

$\mathbf{R_B}$ = { #R1, …., #Rp }

$$\mathbf{R_B} = \delta_{\#O \leftarrow \#R}(\pi_{\#O}(\sigma_{\#B = \mathbf{B}}(\mathcal{BO} \rhd\lhd (\sigma_{ObjType = “R”}(\mathcal{O}))))) \quad (1)$$

It has to be proved that the membership of all the other objects in the database results from the set $\mathbf{R_B}$ and the sets $\mathcal{RE}$, $\mathcal{RA}$, $\mathcal{EA}$ and $\mathcal{AD}$.

The sets $\mathcal{RE}$, $\mathcal{RA}$, $\mathcal{EA}$ and $\mathcal{AD}$ are invariant in respect to their members' positioning within different databases. The stored information can be reduced to the data about relationship memeberships in databases.

Definition 2

Structural properties of a relationship R_i (i = 1, , p) are defined by

- the entities $E_1, \ldots, E_m$ involved
- relationship attributes: $A_1, \ldots, A_n$.

Let the set RE_{Ri} contain the identifiers for entities involved in the relationship R_i :

RE_{Ri} = { $\#E_1, \ldots, \#E_m$ }

Let the set RA_{Ri} contain the identifiers for the attributes from the relationship R_i :

RA_{Ri} = { $\#A_1, \ldots, \#A_n$ }

Definition 3

Let the members of the set $\mathbf{RE_B}$ be the identifiers of the involved entities connected to all the relationships contained in the database **B**:

$$\mathbf{RE_B} = \bigcup_{\#R_i \in \mathbf{R_B}} RE_{Ri} \quad (2)$$

Let $\mathbf{E_B}$ be a set of identifiers for all the entities contained in the database **B**.

In the database **B**, no entity can exist which is not connected to another entity through at least one relationship. Therefore, the set $\mathbf{RE_B}$ contains the identifiers of all the entities of the database **B**. That implies: $\mathbf{E_B} = \mathbf{RE_B}$

Lemma 1

Entities from the database **B** can be deduced from the sets $\mathbf{R_B}$ and RE_B.

Proof

The set RE_{Ri} can be expressed as:

$$RE_{Ri} = \pi_{\#E}(\sigma_{\#R = \#Ri}(\mathcal{RE}))$$

From the definition 3, it follows:

$$\mathbf{E_B} = \bigcup_{\#R_i \in \mathbf{R_B}}(\pi_{\#E}(\sigma_{\#R = \#Ri}\ \mathcal{RE}))) = \pi_{\#E}(\bigcup_{\#R_i \in \mathbf{R_B}}(\sigma_{\#R=\#Ri}(\mathcal{RE})))$$

Finally: $$\mathbf{E_B} = \pi_{\#E}(\mathbf{R_B} \rhd\lhd \mathcal{RE})) \quad (3)$$

A conclusion follows that the membership of entities in **B** can be deduced using the sets $\mathbf{R_B}$ and $\mathcal{RE}$.

Definition 4

According to the definition 2, the set RA_{Ri} contains attributes of the relationship R_i , $\#R_i \in \mathbf{R_B}$:

$RA_{R\,i}$ = { $\#A_1, \ldots, \#A_n$}

Let $\mathbf{RA_B}$ contain the attribute identifiers of all the relationship contained in **B**:

$$\mathbf{RA_B} = \bigcup_{\#R_i \in \mathbf{R_B}} RA_{R\,i} \quad (4)$$

Definition 5

Let the set EA_{Ri} contain the attribute identifiers for the entities connected through the relationship R_i , $\#R_i \in \mathbf{R_B}$:

$EA_{R\,i}$ = { $\#A_1, \ldots, \#A_k$}

Let $\mathbf{EA_B}$ contain attribute identifiers from all the entities in **B**:

$$\mathbf{EA_B} = \bigcup_{\#R_i \in \mathbf{R_B}} EA_{Ri} \quad (5)$$

Definition 6

Let $\mathbf{A_B}$ contain all the attributes from **B**. Then, it follows:

$$\mathbf{A_B} = \mathbf{RA_B} \cup \mathbf{EA_B} \quad (6)$$

Lemma 2

Attributes in **B** result from the set $\mathbf{R_B}$ and sets $\mathcal{RA}$, $\mathcal{RE}$, $\mathcal{EA}$.

Proof

The set RA_{Ri} can be expressed as:

$$RA_{Ri} = \pi_{\#A}(\sigma_{\#R = \#Ri}(\mathcal{RA}))$$

From the definition 4:

$$\mathbf{RA_B} = \bigcup_{\#R_i \in \mathbf{R_B}} \pi_{\#A}(\sigma_{\#R=\#Ri}(\mathcal{RA})) = \pi_{\#A}(\bigcup_{\#R_i \in \mathbf{R_B}}(\sigma_{\#R = \#Ri}(\mathcal{RA}))$$

and: $$\mathbf{RA_B} = \pi_{\#A}(\mathbf{R_B} \rhd\lhd \mathcal{RA})) \quad (7)$$

The set EA_{Ri} can be expressed as:

$$EA_{Ri} = \pi_{\#A}(\sigma_{\#R = \#Ri}(\mathcal{EA} \rhd\lhd \mathcal{RE}))$$

From (5):

$$\mathbf{EA_B} = \bigcup_{\#R_i \in \mathbf{R_B}} \pi_{\#A}(\sigma_{\#R=\#Ri}(\mathcal{EA} \rhd\lhd \mathcal{RE}))$$

$$\mathbf{EA_B} = \pi_{\#A}(\bigcup_{\#R_i \in \mathbf{R_B}}(\sigma_{\#R = \#Ri}(\mathcal{EA} \rhd\lhd \mathcal{RE}))), \text{ and:}$$

$$\mathbf{EA_B} = \pi_{\#A}(\mathbf{R_B} \rhd\lhd (\mathcal{EA} \rhd\lhd \mathcal{RE})) \quad (8)$$

Finally, from (6), (7) and (8) it follows:

$$\mathbf{A_B} = \pi_{\#A}(\mathbf{R_B} \rhd\lhd \mathcal{RA})) \cup \pi_{\#A}(\mathbf{R_B} \rhd\lhd (\mathcal{EA} \rhd\lhd \mathcal{RE})) \quad (9)$$

The attribute membership in **B** can be determined using the set $\mathbf{R_B}$ and sets $\mathcal{RA}$, $\mathcal{RE}$ and $\mathcal{EA}$.

Definition 7

Let RD_{Ri} contain the identifiers for domains over which the attributes are defined belonging to R_i, $\#R_i \in \mathbf{R_B}$:

$$RD_{R\,i} = \{ \#D_1, \ldots, \#D_n\}$$

Let $\mathbf{RD_B}$ be the set of domain identifiers for all the relationships in **B**:

$$\mathbf{RD_B} = \bigcup_{\#R_i \in \mathbf{R_B}} RD_{R\,i} \tag{10}$$

Definition 8

Let the set ED_{Ri} contain the identifiers for the domains, over which are defined the attributes, belonging to the entities, which are connected through the relationship R_i , $\#R_i \in \mathbf{R_B}$:

$$ED_{Ri} = \{ \#D_1, \ldots, \#D_k\}$$

Let $\mathbf{ED_B}$ contain the domain identifiers for all the attributes of all the entities in **B**:

$$\mathbf{ED_B} = \bigcup_{\#R_i \in \mathbf{R_B}} ED_{Ri} \tag{11}$$

Definition 9

Let $\mathbf{D_B}$ contain all the domains form **B**. Then, it follows:

$$\mathbf{D_B} = \mathbf{RD_B} \cup \mathbf{ED_B} \tag{12}$$

Lemma 3

The domains of the database B result from the set $\mathbf{R_B}$ and sets $\mathcal{RA}$, $\mathcal{RE}$, $\mathcal{EA}$, $\mathcal{AD}$.

Proof

The set RD_{Ri} can be formulated as:

$$RD_{Ri} = \pi_{\#D}\,(\sigma_{\#R = \#Ri}\,(\mathcal{RA} \triangleright\triangleleft \mathcal{AD}))$$

From the definition **7**:

$$\mathbf{RD_B} = \bigcup_{\#R_i \in \mathbf{R_B}} \pi_{\#D}(\sigma_{\#R = \#Ri}\,(\mathcal{RA} \triangleright\triangleleft \mathcal{AD}\,))$$

$$\mathbf{RD_B} = \pi_{\#D}(\bigcup_{\#R_i \in \mathbf{R_B}} (\sigma_{\#R = \#Ri}\,(\mathcal{RA} \triangleright\triangleleft \mathcal{AD}\,)) \quad \text{and:}$$

$$\mathbf{RD_B} = \pi_{\#D}\,(\mathbf{R_B} \triangleright\triangleleft (\mathcal{RA} \triangleright\triangleleft \mathcal{AD}\,)) \tag{13}$$

The set ED_{Ri} can be formulated as:

$$ED_{Ri} = \pi_{\#D}\,(\sigma_{\#R = \#Ri}\,(\mathcal{RE} \triangleright\triangleleft (\,\mathcal{EA} \triangleright\triangleleft \mathcal{AD}\,))$$

From the definition 8 it follows:

$$\mathbf{ED_B} = \bigcup_{\#R_i \in \mathbf{R_B}} \pi_{\#D}(\sigma_{\#R = \#Ri}\,(\,\mathcal{RE} \triangleright\triangleleft (\,\mathcal{EA} \triangleright\triangleleft \mathcal{AD}\,)))\text{ , or:}$$

$$\mathbf{ED_B} = \pi_{\#A}\,(\bigcup_{\#R_i \in \mathbf{R_B}} (\sigma_{\#R = \#Ri}\,(\,\mathcal{RE} \triangleright\triangleleft (\,\mathcal{EA} \triangleright\triangleleft \mathcal{AD}\,)))$$

Finally:

$$\mathbf{ED_B} = \pi_{\#A}(\mathbf{R_B} \triangleright\triangleleft (\mathcal{RE} \triangleright\triangleleft (\mathcal{EA} \triangleright\triangleleft \mathcal{AD}\,))) \tag{14}$$

From expressions (12). (13), (14), it follows:

$$\mathbf{D_B} = \pi_{\#D}(\mathbf{R_B} \triangleright\triangleleft (\mathcal{RA} \triangleright\triangleleft \mathcal{AD})) \cup \pi_{\#A}(\mathbf{R_B} \triangleright\triangleleft (\mathcal{RE} \triangleright\triangleleft (\mathcal{EA} \triangleright\triangleleft \mathcal{AD}))) \tag{15}$$

meaning that the membership of domains in **B** can be determined using the set $\mathbf{R_B}$ and sets $\mathcal{RA}$, $\mathcal{RE}$, $\mathcal{EA}$, $\mathcal{AD}$.

Theorem 1

The membership of all the objects in a database is completely described by the membership of relationships.

Proof

Sets of entities, attributes and domains of a database are determined by the set $\mathbf{R_B}$ and sets $\mathcal{RE}$, $\mathcal{RA}$, $\mathcal{EA}$, $\mathcal{AD}$, according to the lemmas 1, 2 and 3. As the sets $\mathcal{RE}$, $\mathcal{RA}$, $\mathcal{EA}$, $\mathcal{AD}$ are invariant in respect to their membership in different databases, membership of objects in a database depends only on the set $\mathbf{R_B}$.

As a consequence of the Theorem 1 the relationship BASE_OBJ is substituted by a new relationship BASE_REL. The mapping of the relationship BASE_REL is N:N. The ultimate version of the **mm** model is presented in Fig. 5.

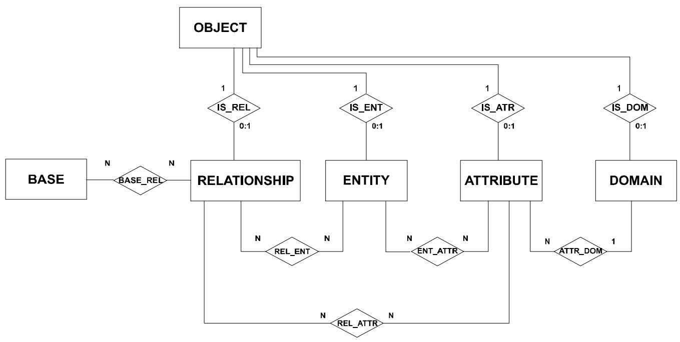

Fig. 5 The ultimate version of the **mm** model

The ultimate structure of the mm model is following:

- relational schemes of the entities:

BASE = {<u>baseID</u>, baseName, baseDescr, baseComment}
OBJECT = {<u>objID</u>, objName, objLongName, objDescr, objComment, objType}
ENTITY = {<u>entID</u>, objID}
RELATIONSHIP = {<u>relID</u>, objID}
ATTRIBUTE = {<u>attID</u>, objID}
DOMAIN = {<u>domID</u>, objID, domType}

III. Description of the Database Models Contained in the MM Base

A. Description of the relationships

According to [2] and [4] the structure of the relationships can be derived from the keys of the entities involved and from the corresponding connectivity. In addition, the relationship description can contain some own attributes in the non-key part.

Definition 10

Let R_i , $R_i \in \mathcal{R}$ be a relationship of degree m and let E_k , $k \in (1, \ldots, m)$ be the entities involved in R_i .

Let $\#R_i$ be the identifier of R_i.

The identifiers of those entities which are connected through R_i, are defined by the set $RE_{Ri} = \{ \#E_1, \ldots, \#E_m\}$. The interconnection of the entities results from the relation $\mathcal{RE}$(REL_ENT) of the **mm** base.

$$RE_{Ri} = \pi_{\#E} (\sigma_{\#R = \#Ri} (\mathcal{RE})) \qquad (16)$$

Let $RA_{Vi} = \{ \#A_1, \ldots, \#A_n\}$ contain the own attributes of R_i. These own attributes are described in the relation $\mathcal{RA}$(REL_ATT) of the **mm** base.

$$RA_{Ri} = \pi_{\#A} (\sigma_{\#R = \#Ri} (\mathcal{RA})) \qquad (17)$$

For each entity E_k , $k \in (1, \ldots, m)$ a key is defined. The definition of entity keys results from the relation $\mathcal{EA}$(ENT_ATT) of the **mm** base.

Let K_k be the set of identifiers of the key attributes belonging to the entity E_k .

$$K_k=\pi_{\#A}(\sigma_{EntKeyPart=TRUE}(\mathcal{EA} \rhd\lhd (\sigma_{\#E = \#Ek} (RE_{Ri})))) \qquad (18)$$

Let **K** be the set of identifiers of all the key attributes from entities $E_1, \ldots, E_m$:

$$\mathbf{K} = \bigcup_{k=1}^{m} K_k$$, where m is the degree of the relationship R_i

Then, it must hold:

$$\mathbf{K} \cap RA_{Ri} = \mathbf{K} \cap \{\#A_1, \ldots, \#A_n\} = \emptyset$$

In other words, the set of own attributes $\{A_1,\ldots, A_n\}$ of the relationship R_i must not contain any key attributes from the entities involved in R_i.

Definition 11

Let A_{Ri} be the set of identifiers of all the attributes of the relationship R_i . This set is defined by the set of its own attributes (RA_{Ri}) and the set of key attributes (set **K**) from the entities involved in R_i .

$$A_{Ri} = RA_{Ri} \cup \mathbf{K} \qquad (19)$$

A rule deriving from the definitions **10** and **11** can be defined:

Rule 3

In the **mm** base, the relationship REL_ATT will contain only its own attributes. The set of all the relationship attributes consists of their own attributes and of the key attributes of the involved relationships.

1) Definition of relationship keys

For every relationship, keys are defined. This definition derives directly from the involved entities and their connectivities.

Definition 12

According to [4], in a relationship connecting the entities $E_1, \ldots, E_k, \ldots, E_m$ connectivity "1" of an entity E_k means that for any value of all the other entities $E_1, \ldots, E_m$, except E_k , there cannot be more than one value of E_k.

As the interconnection of entities in a relationship is represented by the entity keys, a functional dependency can be formulated:

$$\bigcup_{j=1}^{m} K_j \setminus K_k \rightarrow K_k \qquad (20)$$

where the sets K_j , (j = 1, .., m) define the keys of entities $E_1, \ldots, E_m$

Definition 13

A relationship has at least one key. If in a relationship multiple entities with connectivity 1 are involved, the relationship will have as many keys as many of such entities are involved.

If in a relationship no entity of connectivity 1 is involved, the relationship key will consist of all the primary keys of the involved entities.

Let VE_{Vi} be a set to describe the interconnection of entities through the relationship R_i. According to (16):

$$RE_{Ri} = \sigma_{\#R = \#Ri} (\mathcal{RE}).$$

- If for a relationship R_i the condition $\sigma_{connectivity = 1} (RE_{Ri}) = \emptyset$ holds, there will be a single key:

$$\mathbf{K}_{Ri} = \bigcup_{j=1}^{m} K_i \qquad (21)$$

, where K_i is the key of the entity E_i, m is the degree of R_i

- If $\sigma_{connectivity = 1} (RE_{Ri}) \neq \emptyset$, the number of keys will be **card** $(\sigma_{connectivity = 1} (RE_{Ri}))$.

The keys of the relationship are defined:

for every E_k, $\#E_k \in \pi_{\#E} (\sigma_{connectivity = 1} (RE_{Ri}))$ the relationship key is defined:

$$\mathbf{K}_{Ri,\, k} = \bigcup_{j=1}^{m} K_j \setminus K_k \qquad (22)$$

, where K_k is the key of entity E_k , and m is the degree of R_i

From the Definition 13 it results that it is possible to uniquely define the keys of a relationship R_i , founded upon the definition of the relationship R_i , the set RE_{Ri} and upon the definitions of keys in the involved entities.

A rule can be derived from the Definition 13:

Rule 4

The keys of the relationships are not explicitly stored in the mm model. They are deduced in a procedure described by the Definition 13.

In further text, for reasons of completeness, in the relational schemes the attributes deriving from relationship definition will be mentioned, using the notation:

AttName - relationship attributes deriving from the relationship definition

AttName - own relationship attributes

2) Weak entities and weak relationships

The key of a weak entity consists of the owner entity key and of its own key attributes. A weak entity inherits a part of its key through a weak relationship connecting it to the owner entity.

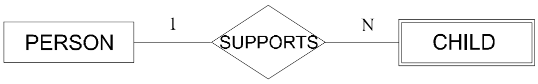

Fig. 6 a weak entity

Example 1

Child is a weak entity and it can be identified through a weak relationship SUPPORTS.
Let the relational schemes of the entities be:

PERSON = {PersonID, FirstName, LastName, DateOfBirth }
CHILD = {PersonID, ChildName, hildDateOfBirth}

As the mapping of the relationship SUPPORTS is 1:N, after the definition12 and (22), the relationship is:

SUPPORTS = {***PersonID,ChildName*** }

Further on, one can define:

Definition 14

For the weak entitiy sets in the **mm** base, only their own attributes will be described in the set ENT_ATT.

Definition 15

Let R_k , $R_k \in \mathcal{R}$ be a weak relationship.
Let EO_k be the owner entity and let EW_k be a weak entity, connected through the relationship R_k.
Let the set KO_k define the key of the owner entity EO_k.
Then, it follows:

$$KO_k = \pi_{\#A}(\sigma_{EntKeyPart = TRUE}(\mathcal{EA} \triangleright\triangleleft (\sigma_{WeakEnt=FALSE \wedge \#R= \#Rk}(\mathcal{RE})))) \qquad (23)$$

Let the set KW_k’ define the key attributes of the entity EW_k:

$$KW_k’ = \pi_{\#A}(\sigma_{EntKeyPart = TRUE}(\mathcal{EA} \triangleright\triangleleft (\sigma_{WeakEnt = TRUE \wedge \#R = \#Rk}(\mathcal{RE})))) \qquad (24)$$

The set to define the key of the weak entity EW_k is:

$KW_k = KO_k \cup KW_k$’

yielding:

$$KW_k=\pi_{\#A}(\sigma_{EntKeyPart=TRUE}(\mathcal{EA} \triangleright\triangleleft(\sigma_{\#R=\#Rk}(\mathcal{RE})))) \qquad (25)$$

Definition 16

The keys of a weak entity are determined according to the Definition 14 and 15.

a) Multiply weak entities

Multiply weak entities are concurrently weak in respect to multiple entities - owners. Most often, they represent relationships which, for different reasons, have been represented by the entities. If possible, they should be substituted by relationships.

The transformation is illustrated in the following example:

Example 2

Model in Fig. 7 is described by the schemes:

STUDENT = { StudID, StudFirstName, StudLastName }
TEACHER = { TeacherID, TeacherFirstName, TeacherLastName }
COURSE = { CourseID, CourseName }
EXAM = { StudID, CourseID, TeacherID, Grade }
CRS_EXM = {***StudID, CourseID***}
TCHR_EXM = { ***StudID, CourseID, TeacherID*** }
STDNT_EXM = { ***StudID, CourseID*** }

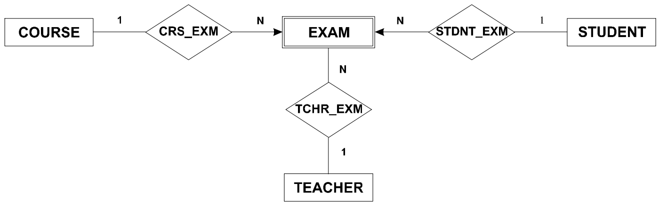

Fig. 7 an example of a multiply weak entity

From the displayed relational schemes and corresponding keys, it can become obvious that the schemes CRS_EXM, TCHR_EXM and STDNT_EXM are redundant and contained within the scheme EXAM. Therefore, a transformation into the model in Fig 8 is possible.

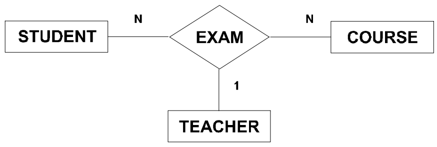

Fig. 8 a relationship substitutes the multiply weak entity

Relational schemes for the model in Fig. 8 are:

STUDENT = { StudID, StudFirstName, StudLastName }
TEACHER = { TeacherID, TeacherFirstName, TeacherLastName }
COURSE = { CourseID, CourseName }
EXAM = { ***StudID, CourseID***, ***TeacherID***, Grade }

The transformation into a relationship makes the model simpler and the relationships among the objects become more visible. However, some multiply weak entities cannot be transformed into relationships.

Example 3

Entities in Fig. 9 are described by the following schemes:

BOOK = { BookID, BookTitle }
MEMBER = { MemberID, MemberFirstName, MemberLastName }
BORROWING = {***BookID, MemberID***, BorrowingDate, ReturningDate }

Fig. 9 Multiply weak entity that cannot be transformed into a relationship

The entity BORROWING is weak in respect to the entities BOOK and MEMBER.

The weak entity from the Example 3-3 cannot be transformed into a relationship because its key, in addition to the identifiers of the two entities, contains the attribute **BorrowingDate**, which is not a key of any entity in the model.

As the weak entities can appear weak in respect to more than one entity, the definition **15** must be generalised.

Definition 17

Let EW_k, $EW_k \in \mathcal{E}$ (ENTITY) be a weak entity.
Let the set **RO** define the weak relationships for the entity EW_k:

$$\mathbf{RO} = \pi_{\#R}\,(\sigma_{WeakEnt = TRUE \wedge \#E = \#EWk}\,(\mathcal{VE})) \qquad (26)$$

Let $\mathbf{ER_k}$ define all the owner entities for EW_k:

$$\mathbf{ER_k} = \pi_{\#E}\,(\sigma_{WeakEnt = FALSE}\,(\mathcal{RE} \rhd\lhd \mathbf{RO})) \qquad (27)$$

The key of the weak entity EW_k is defined as:

$$KW_k = \pi_{\#A}(\sigma_{EntKeyPart = TRUE}(\mathcal{EA} \rhd\lhd \mathbf{ER_k})) \cup \pi_{\#A}(\sigma_{EntKeyPart = TRUE \wedge \#E = \#EWk}(\mathcal{EA})) \qquad (28)$$

The key of a weak entity consists of its own key attributes and of the key attributes belonging to all the entities connected to it through the weak relationships.

b) Specialisations, as weak entities

A specialisation entity without an own key, represents a special case of a weak entity with the relationship (0:1):1 to the owner entity.

Therefore, for the specialisation entities without their own key, the rules valid for the weak entities can be applied.

3) Problem of naming the attributes in reflexive and parallel relationships

a) Reflexive relationships

In the reflexive relationships, a same entity appears in two different roles. In formulation of the relational scheme of such a relationship, the entity key attributes will appear twice, each time in a different role. After the definition 13, through the operation of union, one of the keys would get lost and the relationship would lose its genuine meaning. Therefore, the following rules are defined:

Rule 5

In descriptions of the reflexive relationships, one of the tuples in the relation $\mathcal{RE}$, describing the reflexive relationship, must bear the name of a role, different from any entity names.

Rule 6

In the definition of relational scheme of a relationship, the names of the roles are used. The name of a role can be appended to the names of key attributes in an entity. Alternatively, if the entity name is contained in the key attribute names, the name of the entity contained in the attribute name is substituted with the role name.

b) Parallel relationships

Any two relationships are parallel if they connect the same pair of entities. In formulation of relational schemes for such relationships, if the entities do not have different role names, the schemes for two or more relationships would be produced with the same keys. If such relationships did not contain any own attributes, they would be described by same schemes. Such a description would yield a wrong conclusion that one of these relationships is superfluous. To avoid this wrong conclusion, the following rule is formulated:

Rule 7

In description of the parallel relationships, the tuples in the relation $\mathcal{RE}$, describing a relationship parallel to another one, must have the role name defined which is different from the entity names. For the definition of the relational scheme the role names are used.

Rule 8

The name of the role can be appended to the key attribute names, or, if the entity name is contained in the attribute key name, the entity name is replaced with the role name.

B. Description of entities

To simplify the selection and recombination of the objects from the **mm** base, it is desirable that the objects be atomic, i.e. that they are described exclusively by their own properties and that no single database object would simultaneously contain information about several semantically different objects.

This requirement is especially important for the entities. Their attribute sets often contain foreign keys and their own relationship attributes for the 1:N mapping. Therefore, the following rule is defined:

Rule 9

In the **mm** base in the relation $\mathcal{EA}$, defined over the relational scheme ENT_ATR , a single entity is coupled only to its own attributes. All the attributes that in any way relate to the connections with the other entities, must be stored within the relationships.

IV. CONCLUSION

In the **mm** base, every object is connected to the other objects that define its structural properties. An attribute is connected to its domain, entities are connected to its attributes. An exception are the domains. They belong to the lowest semantic category and they have no lower objects to further describe their structure. Relationships are connected to the entities involved. For every entity, its connectivity in a given relationship is defined. Relationships are also connected to their own attributes.

A database model is defined by its relationships. From the relationships contained in a database and from their connections to the objects that define their structural properties, all the elements of a database model can be derived.

Single database models are described in the **mm** base in a non-redundant way. For each object, the description of only its own properties is stored. All the other properties, deriving from connections to other objects, are described through connections to these objects. The entities contain only their own attributes. Foreign keys are removed from the entity descriptions. The relationships are described by their own attributes only. The identifiers for all the entities, connected through a relationship, are deduced from the definition of the relationship. The keys of relationships are not formulated explicitly, but they derive from the keys of entities involved and from their connectivity. The rules for construction of relational schemes are defined.

For models not in concordance with the described rules, the adequate prescribed transformations are performed.

REFERENCES

[1] A.P .Ambrosio., E. Métais, J.N. Meunier: “The Linguistic Level: Contribution for Conceptual Design, View Integration, Reuse and Documentation”, Data & Knowledge Engineering, No. 21, 1997, pp. 111-129.

[2] Peter Pin-Shan Chen: “The Entity-Relationship Model - Toward a Unified View of Data”, ACM Transactions on Database Systems, Vol. 1, No. 1, 1976, pp. 9-36.

[3] W.W. Song, P. Johannesson, J. A. Bubenko Jr.: “Semantic Similarity Relations and Computation in Schema Integration”, Data & Knowledge Engineering, No. 19, 1996, pp. 65-97.

[4] T.J. Teorey, D. Yang, J.P. Fry: “A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model”, Computing Surveys, Vol.18, No. 2, 1986, pp. 197-222.

[5] R. Malhotra: “Meta-modeling framework: A new approach to manage meta-modelbase and modeling knowledge”, Knowledge-Based Systems, Volume 21 Issue 1, February, 2008, pp.133-139.

[6] M. Baranovic, S. Zakosek, D. Kalpic: “The Multi-Meta Model Aimed for Schema Reuse in Conceptual Database Design“, Models and methods in Applied Sciences, Proceedings of the 2nd International Conference EDSCM’12, Saint-Malo & Mont Saint-Michel. France, April 2-4, 2012, pp. 200-205.

[7] Lj. Brkić, M. Baranović, I. Mekterović : “Improving the Completeness and Timeliness by Horizontal Fragmentation of Data Warehouse Tables“, Proceedings of the 11th WSEAS International Conference on TELECOMMUNICATIONS and INFORMATICS (TELE-INFO '12) 2012.

[8] I. Astrova, A. Kalja : “Mapping of SQL Relational Schemata to OWL Ontologies“, Proceedings of the 6th WSEAS International Conference on Applied Informatics and Communications, Elounda, Greece, August 18-20, 2006 (pp375-380)

[9] K. Koyama, K. Seta, Y. Hayashi, M. Ikeda: “Clarifying Modeling Premises As Explicit Ontologies – Axioms For Conceptual Model Management”, SEPADS'06 Proceedings of the 5th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems, Madrid, Spain, 15-17.02.2006, pp. 36-42.

[10] A. Calì, G. Gottlob, A.Pieris: Ontological query answering under expressive Entity-Relationship schemata. Inf. Syst. 37(4): 320-335 (2012)

Mirta Baranović is an associate professor at the University of Zagreb, Faculty of Electrical Engineering and Computing and the Head of the Department of Applied Computing. Her research interests include databases, data warehouses and information systems. She served as Vice Dean for students and Education at the University of Zagreb, Faculty of Electrical Engineering and Computing. Currently, she is a leader of the scientific project Semantic Integration of Heterogeneous Data Sources, financed by Croatian Ministry of Science, Education and Sports.

Slaven Zakošek received his B.Sc., M.Sc. and Ph.D. degree in Computer Science from the University of Zagreb, Faculty of Electrical Engineering and Computing in 1987, 1992 and 2004 respectively. He has been affiliated with Faculty of Electrical Engineering and Computing as a research engineer, research assistant and assistant professor at the Department of Applied Computing. His research interests include data base systems and information systems.

Ljiljana Brkić received her B.Sc, M.Sc and Ph.D. degree in Computer Science from the Faculty of Electrical Engineering and Computing University of Zagreb in 1992, 2004 and 2011 respectively. She has been affiliated with Faculty of Electrical Engineering and Computing as a research engineer from 1993 and as a research assistant at the Department of Applied Computing from 2006. Her research interests include data bases, data warehouses, business intelligence, information systems and programming paradigms.