UNIFYING ENTITY-RELATIONSHIP AND UNIVERSAL RELATION APPROACHES TO USER INTERFACES

by

V.M. Markowitz and J.A. Makowsky

Technical Report #412

May 1986
UNIFYING ENTITY-RELATIONSHIP AND UNIVERSAL RELATION APPROACHES TO USER INTERFACES

Victor M. Markowitz and Johann A. Makowsky
Technion - Israel Institute of Technology

ABSTRACT

We propose to unite the Entity-Relationship (ER) and Universal Relation (UR) approaches to user interfaces. UR schemas are difficult to design, mainly because most of the explicit structure of the schema is implied (concealed) by the attribute names, and are not able to communicate with the users. Unlike UR schemas, ER diagrams are easy to understand and use, and back techniques of paraphrasing the interpretation given by the system to the user's query. We present an ER oriented methodology for the design of UR schemas. The proposed mapping of ER diagrams to UR schemas solves the problem of multiple role representation by UR schemas. A UR schema is said to be ER-consistent, either if it is the translate of an ER diagram, or if it is possible to translate it back into an ER diagram. An algorithm is presented which decides if a UR schema is ER-consistent.

Automatic or guided navigation is important for users with partial knowledge of the database structure. Such users could assume the existence of some relationship-set among certain entity-sets. The main concern of the UR interfaces is to find the most plausible derivation suitin such a relationship-set. The core of the UR methodology is the determination of (maximal) objects, on which these derivations are based; we show that the ER oriented design, embodied by ER-consistency, simplifies their designation.

The user interface we are proposing combines the ER modeling and communication capabilities with the UR methodology for automatic derivation of virtual relationships. Such an interface could back user-system interaction in an effective way by cooperating in solving ambiguities and disparities between its beliefs and those of its users.

Categories and Subject Descriptors: H.2.1 [Database Management]: Logical design; H.2.3 [Database Management]:Languages; H.2.4 [Database Management]:Systems.

General Terms: Algorithms, Human Factors, Languages.

Additional Key Words and Phrases: dependency, entity-relationship diagram, entity-relationship consistency, natural language, relational model, query, universal relation, user interface.

Authors' address: Computer Science Department, Technion - Israel Institute of Technology, Haifa 32000, Israel.
BITNET: victor@TECHSEL, janos@TECHSEL.
I. INTRODUCTION.

The main objectives of the Relation Model [3] were logical data independence and communicability, as the dominant characteristics of a data model easy to understand, use and communicate about. The relational model attains these objectives with respect to uninterpreted data, but fails to provide a suitable framework to deal with interpreted data, that is information. The relational model user works in terms of data representations which hide most of the structure of the modeled environment. The communicability principle is further damaged in the Universal Relation Model (cf. [14]), which is supposed to achieve the user independence from the logical structure of a database, by concealing the partition of a relational schema into relations. Universal Relation Schemas consist of unnamed sets of attributes, called objects, together with key dependencies. Relation names are eliminated and most of the explicit structure of a Universal Relation Schema is implied by the attribute names. Consequently, the attribute naming becomes a critical problem and the requirement to know and understand the relations structure is replaced, for the user, by the requirement to manage a names structure, which surely is a poor substitute for a semantic structure.

The Universal Relation Model is based on so called assumptions which have been often criticized as being unreasonable in modeling the real world [1]. Subsequent to some of these objections the Universal Relation concepts underwent various changes (a survey is provided by [14]). Notable in these developments is the increasing use of Entity-Relationship oriented concepts. However these concepts are used only informally and in a limited way. The Universal Relation Model originally was not able to represent multiple relationships between attributes. More precisely, it was based on the unique role assumption [9], which requires from an attribute to always represent both a same class of entities and a same role for that class. Recently [7] has proposed to extend the model to capture the multiple role playing of attributes. The extension, which consists of the introduction, in addition to the Universal Relation Schema, of a role graph, is incompletely specified, restrictive and unnecessarily complex.

The features of the Entity-Relationship Diagram (ERD) (introduced in [2] and extended in [6]) attempt to

Unifying ER and UR approaches to user interfaces
correspond to the intentional structures naturally occurring in information system applications; they reflect a natural, although somehow limited, view of the world: entities are qualified by their attributes and interactions between entities are expressed by relationships. We present an Entity-Relationship oriented methodology for the design of Universal Relation Schemas. We claim that only Universal Relation Schemas corresponding to Entity-Relationship Diagrams make sense. A Universal Relation Schema is said to be Entity-Relationship (ER) consistent, either if it is the translate of an Entity-Relationship Diagram, or if it is possible to translate it back into an Entity-Relationship Diagram. We present an algorithm which decides if a schema is ER-consistent. Moreover, the mapping of Entity-Relationship Diagrams to Universal Relation Schemas we are proposing naturally solves the problem of multiple relationship representation by Universal Relation Schemas.

ER-consistency allows a direct relation of ER and UR approaches to user interfaces. Users with partial knowledge of the database structure (ERD) could assume the existence of some relationship-set among certain entity-sets. When the derivation of such a virtual relationship-set is not explicitly given, a default, wishfully the most plausible, derivation suiting the virtual relationship-set, has to be found automatically. The use of Universal Relation concepts to back an automatic navigation within an Entity-Relationship Diagram, has been proposed for the first time in [15]; the associated Entity-Relationship Diagram to Universal Relation Schema mapping is informal and involves only restricted diagrams; and the problem of default ERD connections is investigated only in a limited way.

The core of the UR methodology is the specification or determination of (maximal) objects; we show that ER-consistency simplifies their designation. Universal Relation Interfaces have been frequently criticized as being unreliable because of their inability to communicate with the users; the users pose abbreviated queries and the system cannot guarantee the user that its interpretation complies with the user's intended meaning. According to the defenders of the Universal Relation approach [14], what lacks is a technique of paraphrasing the interpretation given by the system to the user's query. The ERD provides the proper background for such techniques. Moreover, the ERD relieves the user of talking in terms of the logical structure of the database, and enables him, instead, to communicate in terms of the information structure of

Unifying ER and UR approaches to user interfaces
the database.

ER-consistency has been introduced in [6] for regular relational schemas. The experience of [6] has certainly influenced us, but otherwise any similarity between [6] and the present paper is only of an expository nature.

In section 2 we give the definition of ER Diagrams; ER Diagrams are defined as directed graphs, their modeling power is extended and a precise characterization of proper diagrams is provided. Next, in section 3, we review some concepts of the Universal Relation Model necessary to our presentation and propose a graphical representation for Universal Relation Schemas. The problem of multiple relationship representation by Universal Relation Schemas and a discussion of a recent attempt to solve it are also part of section 3. In section 4 we present the mapping of Entity-Relationship Diagrams to Universal Relation Schemas and the characterization of these schemas. In section 5 we propose a reverse mapping from Universal Relation Schemas to Entity-Relationship Diagrams and investigate when and whether this mapping is possible. In section 6 we investigate what is the ER correspondent of specifying non basic objects in an Universal Relation Schema and its relation to the problem of finding a default connection among any set of e-vertices in an Entity-Relationship Diagram. While sections 4 and 5 deal with user interface modeling aspects, section 6 discusses user interface communication aspects. We close the paper by summarizing the results and drawing some conclusions.

II. THE ENTITY-RELATIONSHIP DIAGRAM.

The Entity-Relationship Diagram we are presenting below is an extension of the Entity-Relationship Diagram of [6]. Roughly, the extension refers to the definition of the virtual vertex, and to the additional possibilities of expressing existence dependencies as represented by the new form of ERD edge in (iii) below, together with the corresponding modifications of the ER constraints (ER0) to (ER6).

An Entity-Relationship Diagram (ERD) is a finite labeled digraph $G_{ER} = (V, H)$, where $V$ is the disjoint union of three subsets of vertices, $A$, $E$ and $R$:

Unifying ER and UR approaches to user interfaces
(i) A is the set of a-vertices; (ii) E is the set of e-vertices; (iii) R is the set of r-vertices; e-vertices, a-vertices and r-vertices are represented graphically by rectangles, circles and diamonds, respectively;

H is the set of directed edges, where an edge can be of one of the following forms:

(i) $E_i \rightarrow A_j$ ; (ii) $E_i \rightarrow E_j$ ; (iii) $R_i \rightarrow R_j$ ; (iv) $R_i \rightarrow A_j$ ; (v) $R_i \rightarrow E_j$.

We shall use the following notations:

- $\text{Attr}(X_i) = \{ A_j | X_i \rightarrow A_j \in H \}$, where $X_i$ is either $E_i$ or $R_i$;
- $\text{Ent}(X_i) = \{ E_j | X_i \rightarrow E_j \in H \}$, where $X_i$ is either $E_i$ or $R_i$;
- $\text{Rel}(E_i) = \{ R_j | R_j \rightarrow E_i \in H \}$; and
- $X_i \rightarrow X_j$ denotes a directed path from $X_i$ to $X_j$.

Intuitively, e-vertices, a-vertices and r-vertices represent entity-sets, attributes of entity-sets or relationship-sets, and relationship-sets, respectively. An entity-set groups entities of a same type, where the entity-type is perceived as the sharing of a same set-of attributes. A relationship represents the interaction between several entities, and relationships of the same type are grouped in a relationship-set. A relationship-set can have attributes, just like an entity-set. A value-set is a special kind of entity-set, without attributes, and grouping atomic values of a certain type. An attribute is associated with one or several value-sets, and provides an interpretation for that combination of value-sets in the context of an entity-set or relationship-set. An entity-set, $E_i$, in a relationship-set, $R_j$, may have a role, denoted $\text{role}_j$, expressing the function it plays in the relationship.

Every ERD vertex is labeled by the name of the associated entity-set, relationship-set, or attribute name; e-vertices and r-vertices are uniquely identified by their labels globally, while a-vertices are uniquely identified by their labels only locally, within the set of a-vertices connected to some e-vertex/r-vertex. A subset of the attributes associated with an entity-set, $E_i$, $\text{Attr}(E_i)$, is specified as the entity-identifier, $\text{Id}(E_i)$.

ERD edges specify existence constraints:

$(X_i \rightarrow A_j)$ where $X_i$ is either an e-vertex or an r-vertex, express the fact that an attribute value is meaningful only as characterizing an entity or relationship;

Unifying ER and UR approaches to user interfaces
(E_i → E_j) an entity depend on on the existence of other entities, through ISA or ID relationships:

- the ISA relationship expresses a subset relationship between two entity-sets; the corresponding edge is labeled by an ISA label and is called ISA-edge; ISA-edges represent generalization hierarchies; for \( E_i \rightarrow ISA \rightarrow E_j \), \( E_i \) is said to be a specialization of \( E_j \), and \( E_j \) is said to be a generalization of \( E_i \);

- the ID relationship expresses an identification relationship between an entity-set, called weak entity-set, which cannot be identified by its own attributes, but has to be identified by its relationship(s) with other entity-sets; the corresponding edge is labeled by an ID label and is called ID-edge;

(R_i → E_j) a relationship can exist only when the related entities also exist;

(R_i → R_j) a relationship can be dependent on the existence of related relationships.

A directed path in an ERD is called an ISA-path if all the edges on the path are ISA-edges; a directed path in an ERD is called an ID-path if at least one edge on the path is an ID-edge; and a directed path in an ERD is called an R-path if the start vertex of the path is an r-vertex and all the other vertices on the path are e-vertices. An ISA-path (ID-path) from \( E_i \) to \( E_j \), is denoted \( E_i \rightarrow ISA \rightarrow E_j \) (\( E_i \rightarrow ID \rightarrow E_j \)).

Two relationship-sets, \( R_i \) and \( R_j \), of an ERD \( G_{ER} \), are said to be independent, if neither \( R_i \rightarrow R_j \), nor \( R_j \rightarrow R_i \), are edges of \( G_{ER} \); two independent relationship-sets, \( R_i \) and \( R_j \), of an ERD \( G_{ER} \), are said to be disjoint, if \( Ent(R_i) \cap Ent(R_j) = \emptyset \).

Cardinality constraints are restrictions on the minimum and maximum number of entities from a given entity-set, that can be related, in the context of some relationship-set, to a specific combination of entities from other entity-sets. Every edge \( R_i \rightarrow E_j \) is labeled as follows: if it corresponds to a maximum cardinality of one, it is labeled with 1, and is called one-labeled; if it corresponds to a maximum cardinality greater than one, it is labeled with \( n \), and is called many-labeled; we shall assume that at least one outgoing edge of every r-vertex is many labeled.

Unifying ER and UR approaches to user interfaces
Note:
ASSIGN \rightarrow WORK means that an employee is assigned to projects only if he works in some department, and only in the departments he works in.

Fig. 1 Entity-Relationship Diagram example.

ERD digraphs obey the following constraints:

(ER0) for any a-vertex of V, A_i, indegree(A_i)=1;
(ER1) for any e-vertex of V, E_i,
  (ER1.i) it cannot have both ISA and ID outgoing edges;
  (ER1.ii) if E_i has outgoing ID-edges, or Ent(E_i)=0, then Id(E_i) must be not empty;
  (ER1.iii) if E_i has ISA outgoing edges, then Id(E_i) must be empty;
  (ER1.iv) for every E_i, Atr(E_i)\neq \emptyset and all its roles are distinct;
(ER2) for any r-vertex of V, R_i, Ent(R_i)≥2;
(ER3) an ERD is a simple, connected dag;
(ER4) for every edge R_i \rightarrow R_j:
  (ER4.i) |Ent(R_i)| > |Ent(R_j)|;
  (ER4.ii) for every E_m \in Ent(R_j)

Unifying ER and UR approaches to user interfaces
\[ \exists E_k \in \text{Ent}(R_i) \text{ such that either } E_k \rightarrow \text{ISA} \rightarrow E_m \text{ or } E_k = E_m, \text{ and} \]

if \( R_j \rightarrow E_m \) is one-labeled, then \( R_i \rightarrow E_k \) is also one-labeled;

(ER5) two ISA paths having the same start e-vertex, must have the same end e-vertex;

(ER6) two ID-paths (R-paths) having the same start e-vertex (r-vertex), must not have the same end e-vertex.

Constraints (ER0) to (ER6) define a certain kind of proper ER structures. An attribute characterizes a single entity-set or relationship-set, therefore an a-vertex is connected by a single edge to a single e-vertex or r-vertex, as expressed by (ER0). Constraint (ER3) above guarantees that directed cycles do not exist; thus, for instance, an entity-set will neither be defined as depending on identification on itself, nor be defined as a proper subset of itself. Constraint (ER5) above guarantees entity-set compatibility. More about the meaning and motivation of these constraints may be found in [6].

An extended ERD, XERD, of an ERD \( G_{ER} \), includes, in addition to the ERD, virtual e-vertices and r-vertices. Virtual e-vertices and r-vertices are graphically represented by double line rectangles and diamonds, respectively. Intuitively, virtual e-vertices (r-vertices) represent entity-sets (relationship-sets) derivable by the composition of other entity-sets and/or relationship-sets. The derivation of virtual entity and relationship-sets is out of the scope of this paper; we only note that a virtual vertex not associated with a derivation, could match several virtual entity or relationship-sets. Extended ERDs has to obey the additional constraint (ER7):

(ER7.i) a virtual e-vertex, \( E_i \), has no incident ID-edges, and has at least one outgoing ISA-edge; and

(ER7.ii) for any virtual r-vertex, \( R_i \), \( \exists R_j \), such that \( R_i \rightarrow R_j \).

The reduced ERD, RERD, of an ERD (XERD) \( G_{ER} \), is the subgraph of \( G_{ER} \), \( G_{ER}' = (V', H') \), where

\[ V' = E \cup R, \text{ and } H' = H - \{ X_i \rightarrow A_j \} - \{ R_i \rightarrow E_j | \exists R_k, s.t. R_i \rightarrow R_k \text{ and } R_k \rightarrow E_j \}. \]

An ERD, \( G_{ER} = (V, H) \), can be associated with an ERD hypergraph, whose set of vertices is \( E \), and whose set of hyperedges is \( \{ \text{Ent}(R_j) | R_j \in V \} \). An ERD is said to be loop-free if its associated hypergraph is acyclic (for hypergraph acyclicity see [5]).

Unifying ER and UR approaches to user interfaces
III. THE UNIVERSAL RELATION MODEL.

From the various Universal Relation Models that have been defined recently, we have chosen the version presented in [7] and [9], as being the most representative and the least restricted.

A Universal Relation Schema (URS) is a finite set of unnamed sets of attributes, called objects. Some of the objects are basic (associations, in [9]); intuitively basic objects represent nondecomposable facts, while nonbasic objects represent meaningful, but decomposable, facts, spanning over several connected basic objects.

Keys, not mentioned in [9], keep their usual relational meaning. A set of objects is said to be key based [13], if whenever XA and XYA are objects and A is not in a key of XA, then A is not in a key of XYA.

We propose the following graphical representation of an URS. Given an URS, O, the associated object graph (OG) is the digraph $G_O = (V, E)$, where $V = O$ and $O_i \rightarrow O_j$ is an edge of E iff $O_i \subset O_j$, and there is no $O_k$ in $O$ s.t. $O_j \subset O_k \subset O_i$. An example of an OG digraph is given in figure 2.

We shall use the following notations:

- $O_i \rightarrow O_j$ denotes a directed path from $O_i$ to $O_j$;
- $Out(O_i) = \{O_j \mid O_i \rightarrow O_j \in G_O\}$; and
- $CK_i = \bigcup \{K_j \mid O_j \in Out(O_i)\}$.

Fig. 2. Containment Graph example.

Unifying ER and UR approaches to user interfaces
Whenever there is a dipath, in $G_0$, from $O_j$ to $O_i$, object $O_i$ is said to be a subobject of $O_j$, and $O_j$ is said to be a superobject of $O_i$; if the length of the dipath is one, $O_i$ is said to be a maximal subobject of $O_j$, and $O_j$ is said to be a minimal superobject of $O_i$. An object corresponding to a vertex with indegree 0 is said to be maximal.

Generalization (ISA) hierarchies are captured by URS as follows: an object $O_i$ having $K_i = CK_j$, and such that for every object $O_j \in \text{Out}(O_i)$, $K_i = K_j$, is said to be a specialization (s-object) of any object $O_j \in \text{Out}(O_i)$, while the corresponding objects $O_j$ are said to be generalizations (g-objects) of $O_i$.

A universal relation database state consists of a set of base relations, one for every basic object of the URS; a relation corresponding to a non basic object, $O_i$, is defined as the join of all the relations associated with basic objects contained in $O_i$:

$$r^*(O_i) = \bigwedge \{r(X) \mid X \text{ is a basic object and } X \subset O_i\}.$$  

Legal states have to satisfy the containment condition: let $X$ and $XY$ be two objects and $r$ and $r'$ be their respective associated relations; then $r'[X] \subseteq r$.

Let $X$ denote any set of attributes; the relation to be associated with $X$ is defined to be the union, of projections onto $X$, of the relations associated with all the minimal superobjects of $X$ [9]:

$$r(X) = \bigcup_{o_i \in O_X \subseteq O} \prod_{x} r(O_i).$$

A set of objects, $O$, is said to be integral [9], iff $O$ is closed under nonempty intersection, that is, the intersection of any pair of objects is either empty, or equals another object. When an object set is integral, any set of attributes is associated with the relation corresponding to its unique minimal superobject [7], whenever such a superobject exists.

The Universal Relation Model originally was not able to represent multiple relationships between attributes. More precisely, it was based on the unique role assumption [9], which requires from an attribute to always represent both a same class of entities and a same role for that class. A possibility to overcome this restriction is to rename attributes. As noted in [7], attribute renaming loses the relationship between attributes representing a same entity in different roles. Instead, [7] proposes to extend the model to capture the
multiple role playing of attributes. The extension consists on the schema level of an additional role graph \( R\)-graph \( G_R = (U, E) \), where \( U \) is the set of all the attributes and \( A_i \rightarrow A_j \) is an edge of \( E \) iff \( A_i \) and \( A_j \) are role related, that is, are domain compatible and if any \( A_j \) instance is known to be always also an \( A_i \) instance.

Note that the role relationship refers, in fact, to generalization hierarchies, the role relationships are based on inclusion dependencies which are added to an already existing inclusion constraint, the containment condition, and the role relationship is defined only for single attributes which are also, tacitly, assumed to be keys of basic objects. Our main objection, however, is that the extension makes the Universal Relation more complex (without achieving generality, as noted in the conclusion of [7]), and unnecessarily so, as we shall show in the next sections.

IV. MAPPING ER DIAGRAMS INTO UNIVERSAL RELATION SCHEMAS.

In this section we present the mapping of Entity-Relationship Diagrams to Universal Relation Schemas and the characterization of these schemas.

In an RERD every e-vertex is connected only to r-vertices representing independent relationship-sets. Let \( E_i \) be an e-vertex of an ERD, \( G_{ER} \); the set of independent r-vertices of \( E_i \), \( Relin(E_i) \), is defined as:

\[
Relin(E_i) = \{ R_j \mid E_i \rightarrow R_j \text{ in } \text{RERD} \}
\]

**Definition.** A unique-role ERD (ERuD) is an ERD in which for every e-vertex, \( E_i \), for any \( R_k, R_j \in Relin(E_i), k \neq j, \text{Ent}(R_k) \cap \text{Ent}(R_j) = E_i \).

The ERD of figure 1, section II, for instance, is an ERuD. Note that loop-freeness is a sufficient, but not necessary, condition for an ERD to be ERuD. We present below a mapping of ERD into ERuD; the mapping transforms multiple roles of an entity-set into multiple specializations of that entity-set.
Mapping ERD into ERuD (Tu).

Input: an ERD, $G_{ER} = (V, H)$;

Output: an ERuD;

for every e-vertex $E_i$ with $|\text{Relin}(E_i)| > 1$,

for every r-vertex $R_m \in \text{Relin}(E_i),$

for every r-vertex $R_j \in \text{Relin}(E_i)$, s.t. $|\text{Ent}(R_m) \cap \text{Ent}(R_j)| > 1$ do:

(i) add e-vertex $E'_i$ and a-vertex $A_{ij}$, labeled by $\text{role}_{ij}$, connected as follows:

(i.a) $E'_i \text{ISA} \rightarrow E_i$

(i.b) $E'_i \rightarrow A_{ij}$

(ii) for every r-vertex $R_k \in \text{Rel}(E_i)$, s.t. $R_k R_j$ or $R_k \rightarrow R_j$ do:

(ii.a) remove $R_k \rightarrow E_i$

(ii.b) add $R_k \rightarrow E'_i$, labeled by the cardinality of $R_k \rightarrow E_i$.

An exemplification of the Tu mapping is given in figure 3.

Proposition 4.1

In an ERuD, $R_i \rightarrow R_j$ iff for every $E_m \in \text{Ent}(R_j)$ $\exists E_k \in \text{Ent}(R_i)$ such that either $E_k \text{ISA} \rightarrow E_m$ or $E_k = E_m$.

Mapping ERuD into Universal Relation Schemas (T).

Input: $G_{ER} = (V, H)$, an ERuD as defined above;

Output: $(O, K)$, the URS interpreting $G_{ER}$;

(0) initially $O, K$ are empty;

(1) rename all the a-vertices in $G_{ER}$ by prefixing every a-vertex label by the associated e-vertex/r-vertex label, so that they become globally unique within $G_{ER}$.

Unifying ER and UR approaches to user interfaces
An exemplification of the above mapping is given in figure 4, for the ERD digraph of figure 1, section II.

(2) compute the key set of a-vertices, as follows:

(2.i) for every e-vertex $E_i$, $\text{Key}(E_i) = \text{Id}(E_i) \cup \{\text{Key}(E_j) | E_i \rightarrow E_j\}$;

(2.ii) for every r-vertex $R_i$, $\text{Key}(R_i) = \bigcup \{\text{Key}(E_j) | R_i \rightarrow E_j \text{ is many-labeled} \}$;

(3) for every e-vertex/r-vertex $X_i$:

(3.i) define object $O_i$:

$$O_i := \text{At}(X_i) \cup \{\text{At}(A_j) | X_i \rightarrow X_j\};$$

(3.ii) $K_i := \text{Key}(X_i)$;

(3.iii) $O := O \cup O_i$;

(3.iv) $K := K \cup (K_i \rightarrow O_i)$;

An exemplification of the above mapping is given in figure 4, for the ERD digraph of figure 1, section II.

Unifying ER and UR approaches to user interfaces
Theorem 4.2

Mappings $T_u$ and $T$ are total, and any ERD, $G_{ER}$, complying with constraints (ER0) to (ER6), is translatable to an URS, $(O, K)$, by $T_u o T$.

Proposition 4.3

Let $(O, K)$ be the URS translate of the ERuD $G_{ER}$; and let $G_O$ be its associated OG;

(i) $G_O$ is isomorphic to the RERD $G'_{ER}=(H', V')$;

(ii) an edge of $G_O, O_i \rightarrow O_j$, corresponds to an edge of $G_{ER}$ which is;

- ID-labeled iff $K_j$ is not included in $CK_i$;
- ISA-labeled iff $K_i = CK_i = K_j$;
- many-labeled iff $K_i \subseteq CK_i, K_j \subseteq K_i$;
- one-labeled iff $K_i \subseteq CK_i, K_j$ not included in $K_i$.

Objects (keys are underlined):

- $(CITY . NAME)$
- $(STREET . NAME, CITY . NAME)$
- $(PERSON . ID, PERSON . NAME, STREET . NAME, CITY . NAME)$
- $(PERSON . ID, PERSON . NAME)$
- $(PERSON . ID, PERSON . NAME, EMPLOYEE . SALARY)$
- $(PERSON . ID, PERSON . NAME, EMPLOYEE . SALARY, DEPARTMENT . NR)$
- $(DEPARTMENT . NR)$
- $(PROJECT . NR, PERSON . ID, DEPARTMENT . NR, EMPLOYEE . SALARY, PERSON . NAME)$
- $(PROJECT . NR)$

Fig 4 Translation of ERuD digraph of Fig. 1.

Unifying ER and UR approaches to user interfaces
Proof:

(i) a vertex of $G'_{ER}$ is either an e-vertex or an r-vertex; by translation $T$ (step 3), to every such vertex corresponds exactly one object which, in $G_O$, is associated with exactly one vertex; let $X_i \rightarrow Y_j$ be an edge of $G'_{ER}$, such that $X_i$ and $Y_j$ correspond, by translation $T$, to objects $O_i$ and $O_j$ respectively; then $O_j \subseteq O_i$ and there is no $O_k$ in $O$ s.t. $O_j \subseteq O_k \subseteq O_i$, which is represented in $G_O$ by edge $O_i \rightarrow O_j$;

(ii) the proof is based directly on the mapping $T$.

The following theorem provides the characterization of the URS translate of an ERD.

Theorem 4.4

Let $(O,K)$ be the URS translate of an ERD by mapping $T_w o T$. Then $O$ is a set of basic objects which is key-based, and the corresponding $OG$ digraph, $G_O$, has the following properties:

(OG1) $G_O$ is a simple, connected dag;

(OG2) two directed paths having a same start vertex representing an s-object, must meet in a vertex representing an g-object.

Proof:

(i) key-basing is guaranteed by step 2 of $T$, together with constraint (ER4.ii);

(ii) since the OG digraph associated with $(O,K)$, $G_O$, and the corresponding RERD digraph are isomorphic (proposition 4.3(i)), $G_O$ inherits the acyclicity property (ER3) of the respective ERD digraph;

(iii) let $O_i$ be an s-object and suppose $O_i \rightarrow O_j$ and $O_i \rightarrow O_k$; $O_i$ can be the translate only of an e-vertex, and both OG edges correspond to ERD ISA-edges (proposition 4.3(ii)); as above, the OG graph inherits the property (ER5) of the corresponding ERD digraph, which corresponds to property (OG2).
V. ENTITY-RELATIONSHIP CONSISTENCY FOR UNIVERSAL RELATION SCHEMAS.

Now we shall investigate the question of the range of the translation \( T \), and when and whether a reverse translation \( T^R \) is defined. Let \( (O,K) \) be the translate URS of the ERuD \( G_{ER} \). We have seen above that the object set, \( O \), generated by translation \( T \) is key-based, and that the corresponding OG, \( G_O \), is isomorphic to a subgraph of the source RERD \( G'_{ER} \). Given an URS \( (O,K) \), our procedure derives a candidate ERuD from \( G_O \), using the information provided by the key structure, and verifies whether it is a proper ERD.

**Mapping Universal Relation Schemas into ER Diagrams (\( T^R \)).**

Input: URS \( (O,K) \);

Output: either an ERuD, \( G_{ER} \), or a fail message;

1. construct the OG, \( G_O \), associated with the input URS;

2. initialize candidate ERD \( G_{ER} = (V_{ER}, H_{ER}) \); \( V_{ER} := V_O \), and \( H_{ER} := H_D \);

3. label every edge in \( H_{ER}, X_i \rightarrow Y_j \), corresponding to edge \( O_i \rightarrow O_j \), in \( G_O \), as follows:

   (ID-label) \( K_i \) not included in \( CK_i \);

   (ISA-label) \( K_i = CK_i = K_j \);

   (many-label) \( K_i \subseteq CK_i \), and \( K_j \subseteq K_i \);

   (one-label) \( K_i \subseteq CK_i, K_j \) not included in \( K_j \);

4. split \( V_{ER} \) into two disjunctive sets of e-vertices and r-vertices, where vertex X is an r-vertex iff every outgoing edge is either many- or one-labeled;

5. complete candidate ERD:

   for every e-vertex \( E_i \) corresponding to object \( O_i \):

   \[ Atr(E_i) = O_i \cdot \{O_j | O_i \rightarrow O_j \} \]; and

   \[ Id(E_i) = Key(O_i) \cdot \{Key(O_j) | O_i \rightarrow O_j \} \];

   for every r-vertex \( R_i \) corresponding to object \( O_i \):

Unifying ER and UR approaches to user interfaces
for every edge \( R_i \rightarrow R_j \),

for every e-vertex \( E_k \in \text{Ent}(R_j) \),

if there is no e-vertex \( E_m \in \text{Ent}(R_i) \), s.t. \( E_m \rightarrow E_k \);

add \( R_i \rightarrow E_k \) to \( H \), with cardinality established as in (3) above;

(7) verify that the resulting ERD, \( G_{ER} \), is proper, that is, constraints (ER0) to (ER6) hold.

Several remarks concerning \( TR \) are in order:

(i) step (2) of \( TR \) is based on proposition 4.3(i); step (3) of \( TR \) is based on proposition 4.3(ii); step (6) of \( TR \) is based on proposition 4.1;

(ii) both \( T \) and \( TR \) mappings consider only basic objects; the mapping of non basic objects is left for the next section;

(iii) the failure of \( TR \) does not mean necessarily that the URS has no associated ERD, but could be the result of a specific key choosing; such failures could be limited by extending the input set of keys with several, rather than one, candidate keys for every object;

(iv) the reverse mapping of \( Tu \), from ERuD to ERD, which would complete the reverse mapping of URS to ERD, is straightforward.

Entity-Relationship Consistency for Universal Relation Schemas is defined as follows.

Definition. A Universal Relation Schema \((Q,K)\) is said to be \(ER\)-consistent if there is a mapping \( S \) from the set of URSs to the set of ERDs, such that \((T_u o T) o S (Q,K)\) and \((Q,K)\) are equivalent, up to a renaming of attributes.

Theorem 5.1

Let \((Q,K)\) be a Universal Relation Schema. Procedure \( TR \), taking \((Q,K)\) as input, succeeds iff \((Q,K)\) is \(ER\)-consistent.
Sketch of the proof: The if direction follows from the specification (Input/Output of $T^R$) and correctness of $T^R$ (whose proof is out of the scope of this paper). For the other direction, it suffices to show that $T^R$ succeeds on $(T_{\alpha} \circ T) o S (O,K)$ rather than on $(O,K)$. The latter is true because $S (O,K)$ is an ERuD and therefore satisfies constraints (ER0) to (ER6).

ER-consistency is a mandatory condition for UR schemas to reflect real-world situations. However, whether an URS is right or not can be decided only by the designer himself, and in this matter the reverse mapping $T^R$ could be of help. We illustrate the importance of such a verification with the following example, taken from [7] (the keys are our addition): let \{P,OP,PD\}, be a set of basic objects, with keys P, P, and PD, respectively; P stands for product, O for owner and D for dealer - products exist, owners own them, and dealers stock them; let P, P, and PD be the respective keys; then $T^R (O,K)$ is:

![Diagram](image)

which is, at least, disputable. Note that a different choosing of keys will make the translation only slightly different.

VI. THE USER INTERFACE.

In the previous sections we have dealt only with the basic objects of a Universal Relation Schema. In this section we shall complete the URS design by investigating how non basic objects are defined, or determined, and their role in backing user interfaces. The Universal Relation Model is supposed to free the user from knowing the logical structure of a database, that is, the partition of a relational schema into relations. Users of UR interfaces pose abbreviated queries, mentioning only the attributes they are interested in. Let $X$ denote such a set of attributes; the relation to be associated with $X$ is defined to be the union, of projections onto $X$, of the relations associated with all the minimal superobjects of $X$. Recall that a non basic object, $O_i$, is associated with the join of all the relations associated with basic objects contained in $O_i$. Non basic objects are either explicitly asserted [9], or defined over maximal sets of basic objects whose associated relations join losslessly [8]. This latter approach is based on the believe that a join makes sense if and only if it is lossless. Both approaches limit the range of possible connections and try to find the single comp-

Unifying ER and UR approaches to user interfaces
bination yielding derived associations. This combination is conjectured to be the intuitively correct one. For explicitly asserted non basic objects, a unique derivation is assured by object set integrity.

The Universal Relation Model has been frequently criticized (e.g. [1]), as being unreasonable in its modeling capabilities and unreliable as a user interface. Thus, although the choosing of the non basic objects seems to be so critical in the attempt of capturing the accurate intuition, the meaning of non basic objects is inherently obscure. It is also not clear how a designer could decide the structure of non basic objects. The main objection to the Universal Relation Interfaces is their inability of communicating with the users, that is, their inability to guarantee the user that their interpretation complies with the users' intended meaning. What lacks, apparently, is a technique of paraphrasing the interpretation given by the system to the user's query [14].

The ERD provides the proper background for such techniques. To begin with, the ERD relieves the user of talking in terms of the logical structure of the database, which he certainly does not need to know, but enables him, instead, to communicate in terms of the information structure of the database.

Entity-Relationship Diagrams have the following linguistic interpretation [10]. Since relationships model facts expressed by natural language (NL) major (stand alone) declarative sentences, every relationship-set may be denoted by the sentence type of these sentences. These sentence types are required to be elementary, that is, to consist of a single verb and one or several nouns, where each noun denotes an entity-set. Moreover, a relationship-set may be denoted by all the different paraphrases of the associated sentence type. A SUPPLY relationship-set, for instance, could be denoted by ITEM is SUPPLIED to DEPARTMENT by SUPPLIER, DEPARTMENT is SUPPLIED with ITEM by SUPPLIER, and so on. What differentiates the paraphrases is which of the terms is in the subject position, while the ordering of the terms in the object position is irrelevant. The verb of the paraphrase in which a specific entity-set is in the subject position, characterizes the role of that entity-set in the relationship-set. An RERD in which the roles and e-vertex labels are chosen as mentioned above, represents the surface structure of the associated NL sentence types.

The linguistic interpretation of ERDs promotes ER languages having NL like syntax and a natural semantic interpretation. For instance, ERROL [10] has constructs analogous to the following NL combination
patterns:
- relativization, meaning the connection of a sequence of sentences such that any two neighbor sentences are chained by a relativizer, such as that or which, on an object term;
- coordination, meaning the connection of several sentences with the help of the logical connectives and and or; and
- correlation of the various appearances of a same noun in different component sentences of NL complex sentences, with the help of determiners such as this, that, etc, or by textual contiguity.

ER oriented languages are based on the specification of virtual entity-sets or relationship-sets. Virtual entity and relationship-sets are representable by virtual ERD vertices. Note that, generally, various entity or relationship-sets can be associated with a given virtual vertex. Automatic or guided navigation within an ERD is particularly important for users with partial knowledge of the ERD. Such users could assume the existence of some relationship-set among certain entity-sets. Then, one would be interested to find both the virtual vertex matching the respective relationship-set, and the suitable derivation, that is,

given \( E_i, \ldots E_k \),

(i) \( \text{find } R_j, \text{ s.t. } \text{Ent}(R_j) = \{E_i, \ldots E_k\} \); and

(ii) \( \text{determine the associated derivation giving } R_j \).

In general several mutually independent derived relationship-sets, \( \{R_{j_1}, \ldots, R_{j_k}\} \), are found in the stage (i), above.

Let \( X_i \) be a virtual vertex of the XERD \( G_{ER} \); the subgraph rooted in \( X_i \) and consisting of \( X_i \) and all the vertices \( X_j \), such that \( X_i \rightarrow X_j \), together with all the edges connecting them, is called the subgraph induced by \( X_i \) (e.g. figure 5).

Virtual vertices are specified either explicitly, or indirectly, by determining their induced subgraphs, as the maximal subgraphs satisfying some constraint(s), such as join losslessness, or loop-freeness. The virtual vertex designations above, emulate the two approaches to the specification of non basic objects. The meaning and motivation of these approaches may be found in the papers on Universal Relation interfaces, such as

Unifying ER and UR approaches to user interfaces
as [7] or [14]. Once the virtual ERD vertices are either specified or determined, the ERD to URS mapping, $T$, is completed with the mapping of these vertices, exactly as for regular vertices. Obviously, all the characteristics of the $T$ mapping, presented in section IV, are preserved. Note that properties analogous to URS acyclicity and object integrity, can be stated directly over the ERD.

Given several relationship-sets, there are many ways of combining them in order to derive a new relationship-set. From a linguistic viewpoint, the derivation of a new relationship-set corresponds to the combination of the associated NL sentence types into a new, complex, sentence type. We contend that every derived relationship-set is *meaningful* as long as it corresponds to such a complex NL sentence type.

Two straightforward derivations are as follows:

(i) let $R_i$ denote a relationship-set, $\text{Ent}(R_i) = \{E_{i_1}, \ldots, E_{i_k}\}$; any proper subset of $\text{Ent}(R_i)$, $\{E_{j_1}, \ldots, E_{j_m}\}$, defines a virtual entity or relationship-set $X_j$, $\text{Ent}(X_j) = \{E_{j_1}, \ldots, E_{j_m}\}$; on the linguistic level, the omission of one or several object terms from the sentence type associated with $R_i$, implies a new sentence type, describing $X_j$;

(ii) let $\{E_{i_1}, \ldots, E_{i_k}\}$, be a set of entity-sets; $\{E_{j_1}, \ldots, E_{j_k}\}$, is said to be a specialization substitution, or simply substitution, of $\{E_{i_1}, \ldots, E_{i_k}\}$, if for every $m=1..k$, either $E_{i_m} = E_{j_m}$, or $E_{i_m} \overset{\text{ISA}}{\longrightarrow} E_{j_m}$; a substitution is trivial if for every $m=1..k$, $E_{i_m} = E_{j_m}$; let $R_i$ denote a relationship-set, $\text{Ent}(R_i) = \{E_{i_1}, \ldots, E_{i_k}\}$; any non trivial substitution of $\text{Ent}(R_i)$, $\{E_{j_1}, \ldots, E_{j_k}\}$, defines a virtual relationship-set $R_j$, $\text{Ent}(R_j) = \{E_{j_1}, \ldots, E_{j_k}\}$; a single entity-set non trivial substitution expresses the attribute inheritance in a generalization hierarchy.

For example, DEPARTMENT SUPPLIED with ITEM is derived from DEPARTMENT SUPPLIED by SUPPLIER with ITEM, by omission; MANAGER WORKS in DEPARTMENT is obtained from
EMPLOYEE WORKS in DEPARTMENT, by substituting EMPLOYEE with MANAGER, provided MANAGER \(\rightarrow\text{ISA} \rightarrow\text{EMPLOYEE}\); and SALARY of MANAGER is obtained by substituting EMPLOYEE with MANAGER, provided MANAGER \(\rightarrow\text{ISA} \rightarrow\text{EMPLOYEE}\), and SALARY is an attribute of EMPLOYEE.

The UR approach concerning the default derivation to be associated with a virtual vertex, is to choose the derivation corresponding to the relativization composition of all the involved NL sentence types. Note that the sharing of at least one common noun by these sentences is assured by the connectivity of the subgraph induced by the virtual vertex. Such a combination corresponds to the natural join of the relations associated with the involved relationship-sets [11]. It is impossible to guarantee that this interpretation agrees with the user's intended meaning. However, the least a user interface should do is to paraphrase its interpretation of the user's query, and the ERD is just the proper background for this. The construction of the NL sentence types associated with derived relationship-sets is paramount both to the designer of the interface and, later on, to the user-interface communication.

VII. CONCLUSION.

We have proposed to unify the Entity-Relationship and Universal Relation approaches to user interfaces. Recently ER oriented concepts have been employed increasingly within the Universal Relation Model, but only informally and in a limited way. Our paper has formalized this usage. We have introduced the concept of ER-consistency for Universal Relational Schemas; ER-consistency embodies the employment of the ER-oriented design principles. Our claim is that only ER-consistent Universal Relation Schemas make sense.

We have presented two mappings; the first defines the Universal Relation interpretation of ERD digraphs; the second attempts to find the ERD digraph corresponding to a Universal Relation Schema. We have characterized the URS-translatable ERD, and defined the directly URS-translatable ERD, ERuD. As a byproduct, the ERD to URS mapping solves the problem of multiple role representation by Universal Relation Schemas.

Users with partial knowledge of the ERD could assume the existence of some relationship-set among
certain entity-sets. Finding the derivation suiting the users' meant relationship-set is the core of the UR theory. We have shown that ER-consistency simplifies the designation of the (maximal) objects backing the UR methodology of determining such derivations. Moreover, unlike UR interfaces, the ERD is adequate to provide effective communication between users and relational databases ([6], [10]), due to its linguistic aspect. The ERD relieves the user of talking in terms of the logical structure of the database, and enables him, instead, to communicate in terms of the information structure of the database.

The unified user interface we are proposing combines the ERD modeling and communication capabilities with the UR methodology for automatic derivation of virtual relationships. Such an interface could back user-system interaction in an effective way through its capability of cooperating in solving either ambiguities, or disparities, between its beliefs and those of its users.
REFERENCES.


Unifying ER and UR approaches to user interfaces