ERROL REVISITED: TOWARD A NATURAL LANGUAGE ORIENTED DATA DESCRIPTION AND MANIPULATION LANGUAGE

by

V.M. Markowitz

Technical Report #413

May 1986
ERROL REVISITED: TOWARD A NATURAL LANGUAGE ORIENTED DATA DESCRIPTION AND MANIPULATION LANGUAGE

Victor M. Markowitz
Technion - Israel Institute of Technology

ABSTRACT

The data description and data manipulation are two aspects of data modeling which are so interrelated that one would expect that a common language would cover both of them. Since most users understand and communicate about information in terms of text in natural language, it seems appropriate to base such a data language on natural language (NL). A suitable background for such a language is the Entity-Relationship Diagram (ERD) which can be made to reflect the surface structure of the NL sentence types associated with an Entity-Relationship schema. ERROL (Entity Relationship Role Oriented Language) is an ERD backed query language, founded on the linguistic aspect of the ERD. ERROL rules have linguistic analogs, such that the construction of complex expressions follow patterns of natural language sentence combination.

ERROL was originally based on a restricted ERD, its expression of NL quantifiers is set oriented, and it was designed as a query language, so that most of the data description and manipulation aspects have been overlooked. We propose to extend ERROL in all these aspects, both enhancing its NL orientation and making it a full sized data language.

Keywords- DDL, DML, ERD, ERROL, Natural Language, User Interface.
1. INTRODUCTION.

The proposal of the Entity-Relationship Model [Chen76], was motivated by the limitations of the relational model in modeling reality. The features of this model attempt to correspond to the intentional structures naturally occurring in information system applications: entities are qualified by their attributes and interactions between entities are expressed by relationships. Entity-relationship schemas are expressible in diagrammatic form, the Entity-Relationship Diagram (ERD).

The development of semantic models, the Entity-Relationship Model included, has been concerned mainly with the definition of rich structural parts and paid little attention to the data representation and data manipulation aspects of data modeling. As such, they cannot be considered as an alternative to the relational model. Instead, they can be thought of as high level user interfaces for the relational model. Such an approach achieves both the separation of the actual information from its logical data representation, and the inheritance of the existing theoretical work on the relational model.

In section 2 we present an extended ERD, defined as a directed graph, able to represent generalizations and aggregations. Our graphical approach allows an accurate and concise characterization of the ERD. In [MMR] we have investigated the significance of requiring from a relational database schema to comply with an entity-relationship structure, that is, to be representable by an ERD. Relational database schemas there consist of a traditional relational schema together with key and inclusion dependencies. Such a schema is said to be Entity-Relationship (ER) consistent, if it is either the translate of an ERD, or if it is possible to translate it into an ERD.

A data model consists of two parts [TL]: a set of generating rules, concerning the structure and constraint specifications, and a set of operations, defining the actions that can be performed within the data model. Both the structural specification and the operations attempt to reflect some real-world situation. They should be close to the way people naturally perceive the information. Such a view both favors and suggests
the definition of a data language drawing on analogies with the way people communicate, the natural language (NL). ERROL (Entity Relationship Role Oriented Language) [MR83] is a query language backed by ERD user interfaces, and is founded on the linguistic aspect of the ERD. ERROL rules have linguistic analogs, such that the construction of complex expressions follow patterns of natural language sentence combination. However, ERROL was limited by its basing on a restricted ERD, its expression of the NL quantifiers is set oriented, and it was originally designed as a query language, so that most of the data description and manipulation aspects have been overlooked. We propose to extend ERROL by taking into account a more powerful ERD, by enhancing its NL orientation, and by making it a full sized data language.

We perceive both data declaration and manipulation either as assertions or revocations of facts. We shall look into the meaning of different forms of assertions and revocations and their correspondence to classical data declaration and manipulation operations. The ERROL basic structure specification, presented in Section 3, is an immediate application of the linguistic interpretation given to the ERD. In Section 4 we review the combination rules of ERROL, and propose some extensions either backed by the increased modeling capability of the ERD, or motivated by our attempt to enhance the NL orientation of ERROL. The specification and use of virtual entity-sets and relationship-sets, is presented in Section 5. The data manipulation of ERROL, and the way constraints are specified in ERROL, both based on virtual entity-sets and relationship-sets, are also discussed in Section 5. While section 4 presents NL oriented extensions to ERROL, sections 3, and 5 deal with the data language capabilities added to ERROL. We close the paper by summarizing the results and drawing some conclusions.

2. THE ENTITY-RELATIONSHIP DIAGRAM.

The Entity-Relationship Diagram we are presenting is an extension of the Entity-Relationship Diagram defined in [MMR]. Roughly, the extension refers to additional possibilities of expressing existence dependencies and to the inclusion of the aggregation abstraction, as represented by the new forms of ERD edges in (iii) and (vi) bellow.
together with the corresponding modifications of the ER constraints (ER0) to (ER7). The mapping of ER Diagrams into relational schemas is expected to be an adjustment of the analogous mapping of [MMR], section 4, capturing this extension.

An Entity-Relationship Schema, called the Entity-Relationship Diagram (ERD), is a finite labeled digraph, \( G_{ER} = (V, H) \), where

- \( V \) is the disjoint union of four subsets of vertices, \( S, A, E \) and \( R \): (i) \( S \) is the set of s-vertices; (ii) \( A \) is the set of a-vertices; (iii) \( E \) is the set of e-vertices; (iv) \( R \) is the set of r-vertices;

s-vertices and e-vertices are represented graphically by rectangles, a-vertices and r-vertices are represented graphically by circles and diamonds, respectively;

- \( H \) is the set of directed edges, where an edge can be of one of the following forms:
  - \( E_i \rightarrow A_j \);
  - \( E_i \rightarrow E_j \);
  - \( E_i \rightarrow R_j \);
  - \( R_i \rightarrow A_j \);
  - \( R_i \rightarrow E_j \);
  - \( R_i \rightarrow R_j \);
  - \( A_i \rightarrow S_j \).

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 \( R_i \);
- \( X_i \rightarrow X_j \) denotes a directed path from \( X_i \) to \( X_j \).

Intuitively, s-vertices, e-vertices, a-vertices and r-vertices represent value-sets, 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 value-set is a special kind of entity-set, without attributes, and grouping atomic values of a certain type. 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. 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 \( \text{role}_{ji} \), expressing the function it plays in the relationship.

Every ERD vertex is labeled by the name of the associated value-set, entity-set,
relationship-set, or attribute name; e-vertices, r-vertices and s-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, $E_i$, $Atr(E_i)$, is specified as the entity-identifier, $Id(E_i)$.

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;

$(E_i \rightarrow E_j)$ an entity may be dependent 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;
- the ID relationship expresses an identification relationship between an entity-set, called a 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;

$(E_i \rightarrow R_j)$ an entity may be dependent on the existence of a relationship, and the edge specifies that:
- every entity of $E_i$ is involved in some relationship of $R_j$;
- a relationship of $R_j$ may be regarded as a higher-level entity belonging to $E_i$; this kind of abstraction is called aggregation [LS]; the corresponding edge is labeled by an AG label and is called AG-edge;

$(R_i \rightarrow E_j)$ a relationship can exist only when the related entities also exist;

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

Cardinality constraints [TL] 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.

Generally value-sets are not represented in ERD digraphs and, alternatively, every edge
\( E_i \rightarrow A_j \) or \( R_i \rightarrow A_j \), is labeled by the combined type of the value-sets associated with \( A_j \).

The reduced ERD, RERD, is the digraph \( G'_{ER} = (V', H') \), where
\( V' = E \cup R \), and \( H' = H - \{ h \mid h \text{ is } E_i \rightarrow A_j \text{, or } R_i \rightarrow A_j \text{, of } H \} \).

A directed path in an ERD digraph is called an ISA-path if all the edges on the path are
ISA-edges; a directed path in an ERD digraph is called an ID-path if at least one edge on
the path is an ID-edge. An ISA-path from \( E^*_i \) to \( E^*_j \), is denoted \( E^*_i \mapsto ISA \mapsto E^*_j \).

An e-vertex \( E_k \) is said to be a link of e-vertices \( E_i \) and \( E_j \), iff \( E_i \mapsto ISA \mapsto E_k \) and
\( E_j \mapsto ISA \mapsto E_k \), and there is no e-vertex \( E_m \), such that \( E_i \mapsto ISA \mapsto E_m \), \( E_j \mapsto ISA \mapsto E_m \), and
\( E_m \mapsto ISA \mapsto E_k \).

The entity-set, relationship-set and attribute compatibility, whose intuition is

\[
\begin{align*}
\text{SALARY} & \quad \text{NAME} \quad \text{ID} \\
\text{EMPLOYEE} & \quad \text{PERSON} \quad n \\
\text{LIVES} & \quad \text{STREET} \quad 1 \\
\text{WORK} & \quad \text{ASSIGN} \quad m \\
\text{DEPARTMENT} & \quad \text{PROJECT} \quad \text{NR}
\end{align*}
\]

\*Note: \* ASSIGN \rightarrow WORK \text{ 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.
straightforward, have the following ERD based definitions:

(i) two attributes, \( A_i \) and \( A_j \), are said to be compatible iff the \( a \)-vertices representing them are connected to the same \( s \)-vertices, or, alternatively, their respective incoming edges are identically labeled;

(ii) two entity-sets, \( E_i \) and \( E_j \), are said to be compatible iff the \( e \)-vertices representing them

(ii.1) have at least one link; or

(ii.2) either \( E_i \rightarrow ISA \rightarrow E_j \) or \( E_j \rightarrow ISA \rightarrow E_i \);

(iii) two relationship-sets, \( R_i \) and \( R_j \), are said to be compatible iff there is a one-to-one correspondence of compatible entity-sets between \( \text{Ent}(R_i) \) and \( \text{Ent}(R_j) \).

ERD digraphs obey the following constraints:

\( (\text{ER0}) \) for any \( a \)-vertex of \( V \), \( A_i \), \( \text{indegree}(A_i) = 1 \);

\( (\text{ER1}) \) for any \( e \)-vertex of \( V \), \( E_i \),

\( (\text{ER1.i}) \) it cannot have both ISA, ID and AG outgoing edges, and it can have at most one AG outgoing edge;

\( (\text{ER1.ii}) \) if \( E_i \) has at least one outgoing ID-edge, or \( \text{Ent}(E_i) = 0 \), then \( \text{Id}(E_i) \) must be not empty;

\( (\text{ER1.iii}) \) if \( E_i \) has AG or ISA outgoing edges, \( \text{Id}(E_i) \) must be empty; and

\( (\text{ER2}) \) for any \( r \)-vertex of \( V \), \( R_i \),

\( (\text{ER2.i}) \) \( \text{Ent}(R_i) \geq 2 \); and

\( (\text{ER2.ii}) \) it can have at most one AG ingoing edge;

\( (\text{ER3}) \) an ERD is a connected dag;

\( (\text{ER4}) \) for every edge \( R_i \rightarrow R_j \), let \( |\text{Ent}(R_m)| \geq |\text{Ent}(R_n)| \), where \( m = i/j \) and \( n = j/i \); there is one, and only one, subset of \( \text{Ent}(R_m), \text{Ent}(R_n) \), which has a one-to-one ISA path connection with \( \text{Ent}(R_m) \);

\( (\text{ER5}) \) two ISA paths having the same start \( e \)-vertex, must have the same end \( e \)-vertex;
(ER6) for every edge $E_i \rightarrow R_j$, which is not an AG-edge, there is an e-vertex $E_k$, s.t. $R_j \rightarrow E_k$ and $E_i \xrightarrow{ISA} E_k$.

(ER7) let $E_j$ be the end e-vertex of two ID/directed paths having the same start e-vertex/r-vertex;

(ER7.i) every $X_i \rightarrow E_j$ edge belonging to these paths, is labeled by the corresponding role of $E_j$; or

(ER7.ii) if roles are not used, two ID/directed paths having the same start e-vertex/r-vertex, must not have the same end e-vertex.

Constraints (ER0) to (ER7) define 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. Constraint (ER7) assures a minimal, that is, only when necessary, role labeling of the ERD digraphs.

3. BASIC STRUCTURE SPECIFICATION

Entity-Relationship Diagrams have the following linguistic interpretation [MR83] (see also [Chen83]). Usually the information to be represented refers to entities that either have properties or are related by some association. 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. Following the Closed World Assumption [Rei], the sentences represent only positive facts, while negative facts are considered as being implied by the lack of any positive counterpart within the database. The 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 in QTY.

DEPARTMENT is SUPPLIED with ITEM by SUPPLIER in QTY.

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 ERD in which the roles and vertex labels are chosen as mentioned above, represents the surface structure of the NL sentence types describing it (e.g. figure 2).

A relationship between entities may be regarded as a higher-level entity; this kind of abstraction is called aggregation. An aggregate entity-set is denoted by NL nouns, possibly derived from the verb (gerund) characterizing the corresponding relationship-set.

An entity belonging to an entity-set is determined by its properties that is, by the values of the attributes characterizing the respective entity-set. The attributes of an entity-set are grouped in a attribute-declaration having the following form:

role attribute,

.....

role attribute;

Fig.2 ERD Reflecting the NL Surface Structure.
An entity-set is specified as follows:

```
assert entity-set attribute-declaration.
```

One of the attributes is chosen for the external identification (not necessarily unique) of the entity-set; it is marked by the role IDENTIFIED. For example,

```
assert ITEM IDENTIFIED by NUMBER,
    NAMED by NAME,
    HAVING TYPE,
    COLORED by COLOR.
```

A property-specification is the data specification correspondent of the attribute-declaration, and has the following form:

```
id role attribute val,
role attribute val;
```

where 'id' is an identifier value and 'val' is an attribute value. A property specification is said to be partial, if it does not consist of all the attributes of the corresponding entity-set; otherwise it is total. Total property specifications back assertions of single entities, which have the following form:

```
assert entity-set property-specification;
```

For example,

```
assert ITEM IDENTIFIED by '123',
    NAMED 'TOY',
    HAVING TYPE 'T',
    COLORED 'RED'.
```

Generalization views a set of types as one generic type; specialization is the corresponding opposite abstraction process [TL]. Entity-set specialization is specified as follows:

```
assert entity-set is a entity-set [attribute-declaration]
```

For example:

```
assert MANAGER is a EMPLOYEE.
```

No identifier attribute is defined for a specialization entity-set (constraint ER1.iii).
The specification of a relationship-set has the following form:

\[ \text{assert relationship-set is entity-set role} \]

... entity-set (role);

[attribute-declaration].

For example, the SUPPLY relation above is defined as follows:

\[ \text{assert SUPPLY is DEPARTMENT SUPPLIED by} \]
\[ \text{SUPPLIER(SUPPLIES)} \]
\[ \text{ITEM(SUPPLIED)}; \]
\[ \text{HAVING QTY.} \]

Note that the structural definition of relationship-sets, makes possible to view, and to refer, any relationship-set as an aggregate entity-set.

It is assumed that the structure specifications embody the enforcement of the constraints (ER0) to (ER7), introduced in section 2, characterizing proper ER structures.

4. ERROL COMBINATION RULES

ERROL takes advantage of the possibilities posed by the relationship-set denotations based on NL sentence types; by using constructs similar to the NL sentence combination. The following combination patterns are used (a detailed presentation may be found in \[\text{Win}\]):

(i) relativization, meaning the connection of a sequence of sentences such that any two neighboring sentences are chained by a relativizer, such as that or which, on an object term. There are two main forms:

   (i.a) restrictive relativization, where the connecting object term is raised from an object position in the first sentence, to the subject position in the second sentence; the relativizer need not be present; for instance: "department requesting item supplied by supplier";

   (i.b) non restrictive relativization, where the connecting relativizer is not standing for the subject of the second sentence in a chair pair; for instance, the above example may be reformulated as: "item requested by department to which
some supplier supplies this item";

(ii) **coordination**, meaning the connection of several sentences with the help of the logical connectives *and* and *or*, for instance: "item requested by department *and* supplied by some supplier".

(iii) **correlation** of the various appearances of a same noun in different component elementary sentences, with the help of determiners such as: this, that, etc; an implicit correlation is provided by textual contiguity.

An elementary statement of the form \((A \varnothing B)\) or \((A \varnothing k)\) where \(A\) and \(B\) denote compatible attributes, \(k\) is a constant, and \(\varnothing\) is a comparison operator, is called *restriction*. Similarly, a statement of the form \((\text{set } E_i \Theta \text{set } E_j)\), where \(E_i\) and \(E_j\) denote compatible entity-sets, and \(\Theta\) is a set comparator, is called *set-restriction*. Let \(R_i\) denote a relationship-set, \(\text{Ent}(R_i) = \{E_{j_1}, \ldots, E_{j_p}\}\); henceforth we shall denote such a relationship-set by one of the forms: \(E_{j_m} \text{role}_{ij_m} \cdots E_{j_p}\), where \(\text{role}_{ij_m}\) is the role of \(E_{j_m}\) in \(R_i\), and \(E_{j_m} \cdots E_{j_p}\) is any permutation of \(\text{Ent}(R_i)\).

The relationship-set denotations and restrictions are combined, by relativization and coordination, into complex ERROL sentences, analogous to the NL complex sentence types. Correlation within ERROL sentences are assured by identifiers called *correlators*. The correlators fill in ERROL the referencing task of the NL determiners mentioned above, and are easily deduced from the NL correlation. For instance "department requesting some item supplied to this department", becomes DEPARTMENT \(D\) REQUESTING ITEM SUPPLIED to DEPARTMENT \(D\).

Entity-set and attribute names, roles and correlators are the ERROL identifiers; they appear written with upper case letters; correlators always follow an entity-set name, are written with upper case letters and are underlined. *Comments*, may be placed anywhere within an ERROL sentence, with the purpose of including the lacking words of its NL sentence type correspondent; comments appear written with lower case letters, and

Relativizers, such as *that*, *which*, etc., are used in non restrictive relativization; they appear written with lower case letters and are underlined. A restriction \((A = k)\), where \(A\)
is an attribute and \( k \) is a constant, may be written as \((A k)\). Whenever \( A \) is an identifier attribute, or the attribute role is unambiguous, the above restriction may be abbreviated to \((k)\). For instance, \texttt{ITEM HAVING NUMBER='5'} may be abbreviated to \texttt{ITEM '5'} and \texttt{ITEM COLORED by COLOR='RED'} may be abbreviated to \texttt{ITEM COLORED 'RED'}.

Properties of aggregate entity-sets can be embedded in the denotion of the corresponding, abstracted, relationship-set. For example, \texttt{QTY OF SUPPLY} may be alternatively be expressed as

\[
\text{SUPPLIER SUPPLIES ITEM to DEPARTMENT in QTY or QTY in which SUPPLIER SUPPLIES ITEM to DEPARTMENT.}
\]

We present below the rules of constructing ERROL sentences; the extensions we are proposing are marked by (+).

4.1 Omission.

Let \( R_i \) denote a relationship-set, \( \text{Ent}(R_i) = \{E_{i_1}, \ldots, E_{i_k}\} \); any proper subset of \( \text{Ent}(R_i) \), \( \{E_{i_m}, \ldots, E_{i_p}\} \), defines either a virtual relationship-set, or an entity-set, \( X_v \), \( \text{Ent}(X_v) = \{E_{i_m}, \ldots, E_{i_p}\} \). 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 the derived entity-set, or relationship-set, \( X_v \). For example,

\[
\text{DEPARTMENT SUPPLIED with ITEM,}
\]

is derived, by omission, from \text{DEPARTMENT SUPPLIED by SUPPLIER with ITEM}.

4.2 Substitution.

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 \( m = 1, \ldots, k \), either \( E_{i_m} \equiv E_{j_m} \) or \( E_{i_m} -\text{ISA} \rightarrow E_{j_m} \). A substitution is trivial if for every \( m = 1, \ldots, k \), \( E_{i_m} \equiv E_{j_m} \). Let \( R_i \) denote a relationship-set, \( \text{Ent}(R_i) = \{E_{i_1}, \ldots, E_{i_k}\} \); any non trivial substitution of \( \text{Ent}(R_i) \), \( \{E_{i_1}, \ldots, E_{i_k}\} \), defines a virtual relationship-set \( R_j \), \( \text{Ent}(R_j) = \{E_{j_1}, \ldots, E_{j_k}\} \). For example, let \( R_i \) be \texttt{EMPLOYEE WORKS in DEPARTMENT}, and let \texttt{MANAGER ISA EMPLOYEE} be a specialization; then
MANAGER WORKS in DEPARTMENT is obtained from $R_i$ by substituting EMPLOYEE with MANAGER. A single entity-set non-trivial substitution expresses the attribute inheritance in a generalization hierarchy. For instance, let SALARY be an attribute of EMPLOYEE; then

SALARY of MANAGER is obtained by substituting EMPLOYEE by MANAGER.

4.3 Complement.

The ERROL complement corresponds to the Closed World Assumption [Rei] negation, that is, referring facts whose positive counterparts are not explicitly present in the database. Let $R_i$ be a relationship-set denoted by $E_{ijm} role_{jm} \cdots E_{j}$; then the negation of $R_i$, is denoted as

$$E_{ijm} \neg role_{jm} \cdots E_{j}.$$  

For example,

DEPARTMENT not REQUESTING ITEM, denotes a relationship-set associating all the departments and items which exist currently in the database and which are not associated by the relationship-set DEPARTMENT REQUESTING ITEM.

4.4 Coordination.

Let $R_i$ and $R_k$ be two relationship-sets denoted by $E_{ijm} role_{jm} \cdots E_{j}$, and $E_{ijm} role_{kn} \cdots E_{h}$, respectively, and such that $E_{ijm} = E_{ih}$; their coordination, centered on $E_{ijm}$, is denoted

$$E_{ijm} role_{jm} \cdots E_{j} \text{ and/or } role_{kn} \cdots E_{h}.$$  

For example, let ITEM REQUESTED by DEPARTMENT and ITEM SUPPLIED to DEPARTMENT denote two relationship-sets; then "department requesting an item and supplied with this item", which is their combination by and coordination, centered on DEPARTMENT, is expressed as

DEPARTMENT REQUESTING ITEM I and SUPPLIED with ITEM I.
4.5 Relativization.

Let $R_i$ and $R_k$ be two relationship-sets denoted by $E_i \text{ role}_{i_m} \cdots E_i$, and $E_k \text{ role}_{k_n} \cdots E_k$, respectively, and such that $E_i = E_k$; their restrictive relativization on $E_i$ is denoted as

$$E_i \text{ role}_{i_m} \cdots E_i \text{ role}_{i_p} \cdots E_k$$

For example, let ITEM REQUESTED by DEPARTMENT and ITEM SUPPLIED to DEPARTMENT denote two relationship-sets; then "department requesting an item supplied by some supplier to it", represents their combination by relativization and is expressed as

$$\text{DEPARTMENT } D \text{ REQUESTING } \text{ITEM SUPPLIED by } \text{SUPPLIER to DEPARTMENT } D.$$  

Chaining by relativization may be applied in a similar way when restrictions are involved. For example,

$$\text{DEPARTMENT REQUESTING } \text{ITEM in QTY > 10},$$

denotes the chaining of the relation DEPARTMENT REQUESTING ITEM in QTY, with the restriction QTY > 10, and

$$\text{DEPARTMENT REQUESTING } \text{ITEM in QTY > QTY SUPPLIED of ITEM to SUPPLIER},$$

denotes the chaining of DEPARTMENT REQUESTING ITEM in QTY, and ITEM SUPPLIED by SUPPLIER in QTY, respectively, by restriction QTY > QTY.

4.6 Modifiers *.

The NL modifiers [no] more than, [no] less than and exactly [as] are used with a counting sense; they correspond to, and may alternatively be expressed by, regular EROL count expressions. Thus, for example, more than $k$ $E_i$ corresponds to, count $E_i > k$, exactly $k$ $E_i$ to count $E_i = k$, and so on. For example, "department requesting more than $k$ items", is expressed as

$$\text{DEPARTMENT REQUESTING count ITEM > k, or}$$

$$(+) \text{ DEPARTMENT REQUESTING more than k ITEM}.$$  

Similarly, "department requesting more items than supplied by some supplier", is expressed as
4.7 Universal-Quantifier Expressions.

The NL universal quantifiers all, every, at least, only, at most, replace the set-restrictions of the form ($E_i \Theta set E_i$). Thus, all $E_i$ and every $E_i$ correspond to set $E_i$ contains set $E_i$, all and only $E_i$ to set $E_i$ equal set $E_i$, at most $E_i$ to set $E_i$ in set $E_i$, and at least $E_i$ to set $E_i$ contains set $E_i$.

A set-restriction, or the corresponding quantifier expression, can chain, just like an ordinary restriction, two sentences. For example, "department requesting at least the items supplied by some supplier to this department", is expressed as

DEPARTMENT $D$ REQUESTING set ITEM contains

set ITEM SUPPLIED by some SUPPLIER to DEPARTMENT $D$, or

(+) DEPARTMENT $D$ REQUESTING at least the ITEMS

SUPPLIED by some SUPPLIER to DEPARTMENT $D$, denoting the chaining by relativization of two relations, DEPARTMENT REQUESTING ITEM, and ITEM SUPPLIED by SUPPLIER to DEPARTMENT respectively, through the set-restriction set ITEM contains set ITEM, respectively the universal quantifier expression at least ITEM.

5. VIRTUAL ENTITY AND RELATIONSHIP SETS.

We return to the structure specification by presenting assertions involving ERROL sentences. An ERROL sentence specifies either a virtual entity-set or a virtual relationship-set.

5.1 Virtual Relationship-Set.

Virtual relationship-sets can be explicitly asserted by associating them with an ERROL sentence:

assert relationship-set is entity-set role.
entity-set (role);
such that ERROL sentence.

The correspondence is accomplished through correlators; every (entity-set, correlator) pair involved in the virtual relationship-set must appear at least once within the ERROL sentence. For example, let two relationship-sets be denoted by EMPLOYEE EMPLOYED by DEPARTMENT, and DEPARTMENT MANAGED by EMPLOYEE, respectively; then the virtual relationship-set EMPLOYEE MANAGING EMPLOYEE, is defined as

```
assert MANAGER is EMPLOYEE X MANAGES
      EMPLOYEE Y (MANAGED)
```
such that EMPLOYEE X MANAGES DEPARTMENT EMPLOYING EMPLOYEE Y.

5.2 Virtual Entity-Set.

Virtual entity-sets are asserted in the same way as the virtual relationship-sets; additionally they embody a specialization relation (specialization was introduced in section 3 for basic entity-sets). Unlike for virtual relationship-sets, here the correspondence is with the first entity-set of the ERROL sentence, implicitly taken as the generalization entity-set. Virtual entity-sets are defined as follows:

```
assert entity-set is a ERROL sentence.
```
For example,

```
assert RED ITEM is a ITEM COLORED 'RED'.
```

5.3 Queries.

A query is a virtual relationship-set or entity-set, defined temporarily for retrieval purposes. In queries the key word get replaces the key word assert, and no new roles are specified. For example,

```
get EMPLOYEE X EMPLOYEE Y
```
such that EMPLOYEE X MANAGES DEPARTMENT EMPLOYING EMPLOYEE Y.
5.4 Nesting.

Nesting in ERROL is accomplished by embedding definitions of virtual relationship-sets within an ERROL sentences. Embedded ERROL sentences begin with one of the determiners such that or that, and end with ";" or ".". Such definitions are local, therefore instead of roles pseudo-roles are used to assert the presence of a virtual relationship-set. As pseudo-roles are used either the general purpose verb HAVE, or any other verb put between apostrophes. For example, ne 3

SUPPLIER S HAVING ITEM that is

REQUESTED by a DEPARTMENT SUPPLIED by SUPPLIER S;

explicitly refers to a virtual relationship-set associating suppliers and items, such that the items are requested by departments supplied by these suppliers.

For complex universal quantifier expressions the ERROL nesting reflects the nested form of the NL expression. For instance, "supplier that supplies every department that is requesting some item, and supplies every department with every item the department requests", is expressed as

SUPPLIER S SUPPLYING set DEPARTMENT D that is

SUPPLIED by SUPPLIER S with set ITEM contains

set ITEM REQUESTED by DEPARTMENT D; contains

set DEPARTMENT REQUESTING some ITEM, or

(+ ) SUPPLIER SUPPLYING every DEPARTMENT D that REQUESTS some ITEM;

with every ITEM that is REQUESTED by DEPARTMENT D.

5.5 Updates.

Some forms of assertions and revocations correspond to update data manipulations. We stress that information, rather than data, structures are referred, therefore there are only updates which are meaningful on the information structure level. All these manipulations embody the enforcement of the existence constraints specified by the ERD edges.
5.5.1 *Entity Oriented Data Manipulations.*

The assertion corresponding to the insertion of a single entity has been presented in section 3. The following form of assertion corresponds to the modification of entities:

```
assert ERROL sentence defining virtual entity-set is property-specification;
```

where the property specification may be either total or partial. For example,

```
assert ITEM COLORED "RED";

is COLORED "BLUE".
```

The dual, revocation, form of the preceding statement is:

```
revoke ERROL sentence defining virtual entity-set.
```

For example,

```
revoke ITEM NAMED "TOY".
```

In both assertion and revocation, the modification, respectively deletion, refers to the subset of entities corresponding to the mentioned virtual entity-set.

5.5.2 *Relationship Oriented Data Manipulations.*

The following form of assertion corresponds to the insertion of one or several relationships:

```
assert ERROL sentence defining virtual entity-set; role ERROL sentence defining virtual entity-set.
```

For example:

```
assert DEPARTMENT NAMED "D"; SUPPLIED

by SUPPLIER "123";

with ITEM NAMED "TOY";

in QTY "5".
```

The dual, revocation, form of the above statement is similar. For example,

```
revoke DEPARTMENT NAMED "D";

SUPPLIED by SUPPLIER "123"

with ITEM NAMED "TOY".
```

Notice that relationships may be either inserted or deleted, but not modified.
5.6 Constraint Specification.

Virtual entity-sets and relationship-sets are defined not only for derivation, but also for integrity constraint specification purposes. Integrity constraints specify conditions that must be satisfied by the database. We shall mainly refer to constraints specifying restrictions on the mappings represented by virtual relationship-sets, between entity-sets. Constraints express some real-world semantics by restricting the possible associations of the involved entities. Constraints are expressed in the following way:

```assert constraint ERROL sentence defining virtual relationship-set;
```

where all the entity-sets involved in the relationship-set are universally quantified. We shall illustrate below, how constraints are expressed.

**Cardinality** constraints are restrictions on the minimum and the maximum number of entities of a given type, that can be associated, within a relationship-set, to entities of other types. Let a relationship-set be denoted by

```DEPARTMENT SUPPLIED by SUPPLIER with ITEM;
```

then a maximum cardinality constraint could be "every department that is supplied, is supplied by at most (max) ten suppliers", which is expressed as

```(+ assert constraint every DEPARTMENT that is SUPPLIED;
    is SUPPLIED by no more than 10 SUPPLIER;
```

with the set-expression equivalent

```set DEPARTMENT that is SUPPLIED; contained in
set DEPARTMENT that is SUPPLIED by count SUPPLIER <= 10.
```

Minimum cardinality constraints are similarly expressed.

The well known functional dependency is a special case of the cardinality constraint. For example, "every department that is supplied, is supplied by at most one supplier" which expresses the functional dependency `DEPARTMENT -> SUPPLIER`, is expressed as

```(+ assert constraint every DEPARTMENT that is SUPPLIED;
    is SUPPLIED by at most 1 SUPPLIER.
```
Constraints involving logical implication are similarly expressed. For example, "every supplier supplying department A, also supplies department B" may be expressed as

\[
(+) \text{assert constraint DEPARTMENT 'B' SUPPLIED by every SUPPLIER that SUPPLIES DEPARTMENT 'A'};
\]

with set-expression equivalent

\[
\text{assert C is set SUPPLIER that SUPPLIES DEPARTMENT 'A'; contained in set SUPPLIER that SUPPLIES DEPARTMENT 'B'}.
\]

6. CONCLUSION.

We have presented an extension of ERROL, both enhancing its NL orientation and making it a full sized data language. The basic structure specification is an immediate application of the linguistic aspect of the ERD. The combination rules extensions are backed mainly by the increased modeling capability of the ERD we have adopted. Both the data manipulation capabilities and the constraint assertion are based on the specification and use of virtual entity-sets and relationship-sets.

Unlike other data description and manipulation languages that use NL like expressions as "syntactic sugar", the extended ERROL outlined in this paper has many features of the Natural Language (NL). The need for such a data language is extensively explained in [DN]. The ERROL expressions are intended to be close to NL expressions, thereby easy to understand and to formulate. The utilization of ERROL does not require the learning of just another artificial language, but rather the observation of restrictions enforced upon a language already known to the user. The ERROL constructs refer to structure specifications, retrieval and update operations, and some forms of schema restructuring, all known from the field of data manipulation. However, manipulations over ERD are meant to represent certain types of real-world information manipulations, thus ensuring a close relationship between this level of data modeling and the natural structures as perceived by users.

The semantics of ERROL have been expressed with the help of an entity-relationship
oriented relational algebra [MR84]. The common NL analogies of both this algebra and ERROL made straightforward the use of the algebraic operators in the semantic definition of ERROL. The extensions presented in section 4 are directly translatable into regular ERROL expressions, so that they do not pose additional semantic definition problems. As for the update oriented assertion and revocation statements, we point out that information, rather than data, structures are referred, and all these manipulations embody the enforcement of the existence constraints specified by the ERD edges.

The extended ERROL allows to view in a unitary manner the naturally interrelated, description and manipulation sides of data modeling. We believe that our approach both favors a better understanding of the involved semantic problems, and represents a better communication medium for data modeling.

The ERD has been acknowledged as a good user interface, mainly due to its capacity to provide effective communication between users and relational databases ([MM], [MMR]). 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. ERROL is based on the specification of connections among entity-sets, expressing virtual relationship-sets. For a given set of entity-sets, there are, in general, several possible connections, and the required one must be explicitly specified. A user should not be compelled to know the whole structure of a database. Users with partial knowledge of the ERD could assume the existence of some relationship-set among certain entity-sets. The derivation of such a virtual relationship-set could be found, either by default, or by cooperating with the user in solving ambiguities and disparities in the sense of [MM].

Acknowledgements. Yoav Raz contributed both to the design of the original version of ERROL, and to the development of several ERROL oriented user interfaces. Reuven Cohen was the main contributor to the first ERROL implementation and offered many useful suggestions to improve ERROL. Johann Makowsky was the driving force behind [MMR] and [MM], which gave formal substance to the concept of ER oriented user interfaces.
REFERENCES


