Conceptual Modeling by Analogy and Metaphor

Karin K. Breitman, Simone D.J. Barbosa, Marco A. Casanova, Antonio L. Furtado
Departamento de Informática – Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de S. Vicente, 225 – Rio de Janeiro, Brazil – CEP 22451-900
{karin, simone, casanova, furtado}@inf.puc-rio.br

ABSTRACT
Metaphor is not merely a rhetorical device, characteristic of language alone, but rather a fundamental feature of the human conceptual system. A metaphor is understood by finding an analogy mapping between two domains. This paper argues that analogy mappings facilitate conceptual modeling by allowing the designer to reinterpret fragments of familiar conceptual models in other contexts. The contributions of the paper are expressed within the tradition of the Entity-Relation model.

Categories and Subject Descriptors
H.2.1 [**Database Management**]: Logical Design – *Data models, Schema and subschema*.

General Terms
Design, Standardization, Languages.

Keywords
Metaphor, analogy, entity-relationship model

1. INTRODUCTION
Metaphor is not merely a rhetorical device, characteristic of language alone. Lakoff and Johnson [11] argue that "the human conceptual system is fundamentally metaphorical in nature. The essence of metaphor is understanding and experiencing one kind of thing in terms of another." Holyoak and Thagard [10] (p. 220) argue that "metaphor uses the same mental processes as analogical thinking ... a metaphor is understood by finding an analogy mapping between the target domain (the topic of the metaphor) and the source domain. The degree to which an analogy is viewed as metaphorical will tend to increase the more remote the target and source domains are from each other."

In this paper, we claim that analogy mappings facilitate conceptual modeling by allowing the designer to reinterpret fragments of familiar conceptual models in other contexts. Specifically, we propose a discipline for database conceptual schema design, and Semantic Web ontologies as well, that we call conceptual modeling by analogy and metaphor.

The discipline is based on two simple ideas. First, a team of expert conceptual designers would build a standard repository of *source* conceptual models that cover commonly found conceptual design patterns and that are expressed in familiar terms. The source conceptual models will naturally contain fully formalized integrity constraints, as defined by the conceptual design experts. Second, naïve designers would then create new *target* conceptual models in other domains by defining analogy mappings with the source conceptual models in the repository. The target models will then borrow the structure and the integrity constraints from the source models by analogy – essentially a combination of a straightforward renaming process with consistency checking.

We introduce conceptual modeling by analogy and metaphor in the context of the familiar Entity-Relationship (ER) model [4]. We define a simple construct, ISLIKE, to specify analogy mappings. We adopt the weak entity construct as a running example since it is well-known, it has a concise description, and yet it looks sophisticated to most naïve designers.

Winston [9] is an early reference that describes a theory of analogy with applications to AI systems. Metaphors have been used to improve human-computer interface design [1,2,3], in particular, and software design, in general [8]. Goguen [5] describes a formalization of user interface design based on semiotic morphisms. The analogy mappings we describe throughout the paper are, in some sense, morphisms that preserve the structure and the integrity constraints of the source schemas.

2. Analogy Mappings in the ER Model
Holyoak and Thagard [6] introduce analogies as mappings between *target* domain and *source* domain. They argue that "the mapping can be used to enrich understanding of the target by generating new inferences, and it can lead to formation of a schema based on the relational structure common to the target and the source. ... In addition, analogy goes hand in hand with the formation of schemas – new categories that embrace both the source and the target, thus changing our understanding of both." Along these lines, but with a different purpose, we introduce analogy mappings as a way to generate new entity-relationship schemas from previously defined ones, considered to be design archetypes.

We start with simple auxiliary definitions. A *concept name* is an entity, relationship or attribute name and an *alphabet* is a finite set of concept names. Two concept names are *compatible* iff they are both entity names, or attribute names, or relationship names of the same arity. Given an entity-relationship schema **U**, the *alphabet* of **U**, denoted $\mathcal{A}(\mathbf{U})$, is the set of concept names of **U**.

Given two alphabets S and $\mathcal{T}$, an *analogy mapping* from S into $\mathcal{T}$ is a one-to-one function $\alpha: S \rightarrow \mathcal{T}$ such that, for any $s \in S$, if $\alpha(s)$ is defined, then s and $\alpha(s)$ are compatible. Since alphabets are finite, we may conveniently indicate that $\alpha(s)=t$ by a triple of the form *"t* `isLike` *s"*.

CIKM'07, November 6-8, 2007, Lisboa, Portugal.

Note that we require that α be neither total nor surjective. If we assume that α is surjective, then we might drop the compatibility requirement: a concept name s in $\mathcal{V}$ will be interpreted as an entity, relationship or attribute name depending on whether $\alpha^{-1}(s)$ is as an entity, relationship or attribute name, respectively.

Given an entity-relationship schema **S**, an alphabet $\mathcal{T}$ and an analogy mapping $\alpha: \mathcal{A}(\mathbf{S}) \rightarrow \mathcal{T}$, we say that an entity-relationship schema **T** is *created by analogy with* **S** *using* $\mathcal{T}$ *and* α iff

1) $\mathcal{A}(\mathbf{T})=\mathcal{T}$, i.e., the alphabet of **T** is $\mathcal{T}$.
2) *(structural analogy)*
   a) If s is an attribute of t in **S** and $\alpha(s)$ is defined, then $\alpha(t)$ is defined and $\alpha(s)$ is an attribute of $\alpha(t)$ in **T**.
   b) If s is an n-ary relationship over $e_1,..., e_n$ in **S** and $\alpha(s)$ is defined, then $\alpha(e_1),...,\alpha(e_n)$ are defined and $\alpha(s)$ is an n-ary relationship over $\alpha(e_1),...,\alpha(e_n)$ in **T.**
3) *(integrity constraint analogy)* For each integrity constraint I of **S**, if α is defined for all concept names that occur in I, then the sentence obtained from I by replacing each concept name s of $\mathcal{A}(\mathbf{S})$ that occurs in I by $\alpha(s)$ is an integrity constraint of **T**, and these are the only constraints of **T**.

We call **S** and **T** the *source* and the *target* entity-relationship schemes of $\mathcal{T}$ and α. We stress that **T** is created from **S** by using the concepts names in $\mathcal{T}$ and by borrowing the structure and the constraints from **S** via the analogy mapping α. Therefore, α should not be viewed as a structure preserving morphism (Goguen 1999) between **S** and **T**, but rather as a morphism between the alphabets $\mathcal{A}(\mathbf{S})$ and $\mathcal{T}$ that helps construct the target entity-relationship schema **T** from the source schema **S**.

As an example, consider the weak entity construct, typically defined as follows. Let E and W be two entity schemes, and R be a binary relationship between W and E. Suppose that R is n-1, with W on the n side, and total on W. Furthermore, assume that E has a key K and that W has a *discriminating attribute* A such that:

- (Discriminating Attribute Property) Given any consistent state σ of E, W and R, for any $d,d' \in \sigma(W)$, for any $e \in \sigma(E)$, if $(d,e) \in \sigma(R)$ and $(d',e) \in \sigma(R)$ and $d[A]=d'[A]$, then $d=d'$

Usually, one teaches the weak entity construct by resorting to an archetypal schema, **EMPLOYEE**, defined as follows. **EMPLOYEE** models employees and their dependents, with the help of two entity schemes, `Emp` and `Dep`, and a binary relationship `isDepOf`. Assume that `isDepOf` relates a dependent to the employee he or she is a dependent of. Also, assume that `Emp` has a key, `EmpNo`, and that `Dep` has an attribute `DepNo` that satisfies the following property:

- (Discriminating Attribute Property for **EMPLOYEE**) The attribute `DepNo` distinguishes the dependents of the same employee (but not dependents of different employees)

Then, we say that `Dep` is a *weak entity* of `Emp`, with discriminating attribute `DepNo` and relationship `isDepOf`. Note that the Discriminating Attribute Property for **EMPLOYEE** is intuitively equivalent to the formal definition of the Discriminating Attribute Property, but it is much simpler to understand.

We may now, for example, create an entity-relationship schema **BOOKS** (see Fig. 1a – top part) by analogy with **EMPLOYEE** using

- the alphabet $\mathcal{B}$ = {`Book`, `Edition`, `isEdOf`, `ISBN`, `EdNo`}
- the analogy mapping α defined as:

```
Book       isLike      Emp
Edition    isLike      Dep
isEdOf     isLike      isDepOf
ISBN       isLike      EmpNo
EdNo       isLike      DepNo
```

BOOKS will then have integrity constraints analogous to those of **EMPLOYEE:**

- `Book` is an entity scheme with key `ISBN`
- `Edition` is an entity schema with an attribute `EdNo` that satisfies the Discriminating Attribute Property
- `isEdOf` is an n-1 binary relationship, associating `Edition` with `Book`, that is total on `Edition`

We emphasize that none of the integrity constraints of **BOOKS** needs to be explicitly defined, being a consequence of defining that **BOOKS** is created by analogy with **EMPLOYEE** using $\mathcal{B}$ and α. Furthermore, we claim that the conceptual burden of understanding what the Discriminating Attribute Property means in **BOOKS** is alleviated since it follows by analogy with the Discriminating Attribute Property for **EMPLOYEE**.

The argument also applies to the behavioral aspects of weak entities. We postulate that, in the **EMPLOYEE** schema, deleting an employee propagates to the deletion of his or her dependents and that the insertion of a new dependent is blocked if the corresponding employee does not exist, or else the discriminating attribute is not properly defined (for the new dependent). Therefore, any schema declared to be like **EMPLOYEE** will inherit such behavior.

3. Metaphor Mappings in the ER Model

Recall from Section 1 that Holyoak and Thagard [6] (p. 220) argue that "metaphor uses the same mental processes as analogical thinking... a metaphor is understood by finding a mapping between the target domain (the topic of the metaphor) and the source domain. The degree to which an analogy is viewed as metaphorical will tend to increase the more remote the target and source domains are from each other."

We adopt a more pragmatic position and introduce the notion of a *metaphor mapping* simply as a partial analogy mapping. However, we reinterpret how target schemas are induced from source schemas.

Let **S** be an entity-relationship schema, $\mathcal{T}$ be an alphabet and $\mu: \mathcal{A}(\mathbf{S}) \rightarrow \mathcal{T}$ *be* a metaphor mapping. Assume that $\mathcal{A}(\mathbf{S}) \cap \mathcal{T} = \emptyset$. We say that an entity relationship schema **T** is *created by metaphorical analogy with* **S** *using* $\mathcal{T}$ *and* μ iff **T** is created by analogy with **S** using $\mathcal{U}$ and β, where

1) $\mathcal{U} = \mathcal{T} \cup \{s \in \mathcal{A}(\mathbf{S}) \,/\, \mu(s) \text{ is undefined}\}$
   ($\mathcal{U}$, the alphabet of **T**, is $\mathcal{T}$ together with the concept names in $\mathcal{A}(\mathbf{S})$ for which $\mu(s)$ is undefined)

2) $\beta: \mathcal{A}(\mathbf{S}) \rightarrow \mathcal{U}$ is the analogy mapping defined as

$\beta(s)=\mu(s)$ if μ is defined for s

$\beta(s)=s$ if μ is undefined for s

Intuitively, the construction of the new entity-relationship schema **T** by metaphorical analogy with **S** using $\mathcal{T}$ and μ is a process that:

- borrows part of the original alphabet of **S** – those concept names for which μ is undefined (the reason why we require that $\mathcal{A}(\mathbf{S}) \cap \mathcal{T} = \emptyset$)
- expands the metaphor mapping μ to a total analogy mapping β by taking $\beta(s)=s$ when μ is undefined for s
- copies the structure and the integrity constraints from **S** using the expanded alphabet and the expanded analogy mapping

As an example, suppose that we start with

- an alphabet C with just two entity names, `Book` and `Edition`, and an attribute name, `ISBN`
- an analogy mapping μ defined as follows:

```
Book    isLike  Emp
Edition isLike  Dep
ISBN    isLike  EmpNo
```

Note that μ is defined just for some of the concepts of **EMPLOYEE**. Then, we may create the entity-relationship schema **BOOKS2** by metaphorical analogy with **EMPLOYEE** using C and μ as follows. First, we add to C the attribute `DepNo` and the relationship name `isDepOf`, for which μ is undefined, and expand μ to these new concept names:

`DepNo` in $\mathcal{A}$(**EMPLOYEE**) `isLike DepNo` in C

`isDepOf` in $\mathcal{A}$(**EMPLOYEE**) `isLike isDepOf` in C

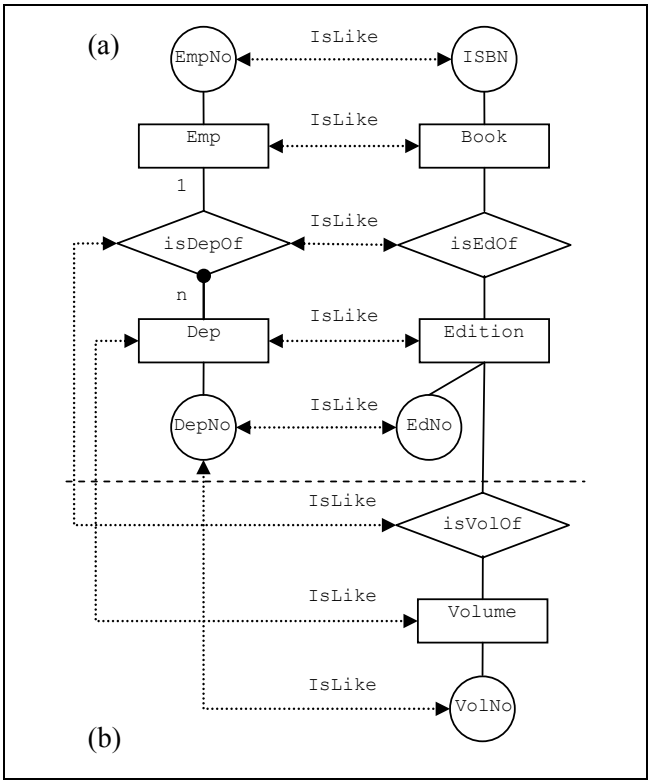

Fig. 1. Example of analogy mapping.

Note that μ is now total on the alphabet of **EMPLOYEE** and is the identity for the concept names `DepNo` and `isDepOf` added to C. Then, we define that **BOOKS2** is the entity-relationship schema created by analogy with **EMPLOYEE** using (the expanded alphabet) C and (the expanded analogy mapping) μ. In some sense, we completed the alphabet C until it has concepts like those in the alphabet of **EMPLOYEE,** and extended the analogy mapping μ until it is defined for all concepts of **EMPLOYEE**.

As a second example, we may define **BOOKS3** using an alphabet $\mathcal{D}$ containing just `Book` and `ISBN`, and the analogy mapping γ defined as follows:

```
Book  isLike  Emp
ISBN  isLike  EmpNo
```

Then, we may assert that **BOOKS3** is created by metaphorical analogy with **EMPLOYEE** using $\mathcal{D}$ and γ, which implies that **BOOKS3** has:

- a new entity scheme `Dep`, which implicitly is like `Dep` of **EMPLOYEE**
- a new attribute `DepNo` of `Dep`, which implicitly is like the attribute `DepNo` of **EMPLOYEE**
- a new relationship scheme `isDepOf` between `Book` and `Dep`, which implicitly is like `isDepOf` of **EMPLOYEE**
- integrity constraints analogous to those of **EMPLOYEE**

We stress that we use the concept of metaphor as a partially specified analogy mapping, and introduce the construction of a new target entity-relationship schema by metaphor as a process that borrows the structure and the integrity constraints from the source schema, as well as part of the original alphabet – a feature not present in pure analogy.

4. Schema Expansion by Analogy and Metaphor Mappings

We may promote a database conceptual design strategy that favors the definition of conceptual schemas by gradually incorporating predefined conceptual schema fragments through the use of analogy or metaphor mappings.

Let **B** and **S** be two entity-relationship schemas, $\mathcal{T}$ be an alphabet and $\alpha: \mathcal{A}(\mathbf{S}) \rightarrow \mathcal{T}$ be an analogy mapping. Assume that any concept name $s \in \mathcal{A}(\mathbf{B}) \cap \mathcal{T}$ is consistently understood in both alphabets, that is, s is an entity name in $\mathcal{A}$(**B**) iff s is an entity name in $\mathcal{T}$, and similarly for attribute names and relationship names. Then, we say that an entity-relationship schema **T** *expands* **B** *by (metaphorical) analogy with* **S** *using* $\mathcal{T}$ *and* α iff **T** is the union of **B** and the entity-relationship schema created by (metaphorical) analogy with **S** using $\mathcal{T}$ and α. We call **B**, **S** and **T** the *base*, the *source* and the *target* entity-relationship schemes of the analogy mapping α.

This notion of expansion by analogy permits the definition of new entity-relationship schemas by repeatedly using analogy mappings. For example, consider again an entity-relationship schema **BOOKS** created by analogy with **EMPLOYEE** using the alphabet `Book`, `Edition`, `isEdOf`, `ISBN`, `EdNo` and the analogy mapping α defined as follows:

```
Book        isLike  Emp
Edition     isLike  Dep
```

```
isEdOf      isLike  isDepOf
ISBN        isLike  EmpNo
EdNo        isLike  DepNo
```

We may define another entity-relationship schema **BOOKS4** that captures that book volumes are weak entities with respect to book editions (see Fig. 1b – bottom part). **BOOKS4** is created by expanding **BOOKS** again by analogy with **EMPLOYEE**, but using the alphabet $\mathcal{I}$ consisting of `Edition`, `Volume`, `isVolOf`, `EdNo`, and `VolNo`, and the analogy mapping ν defined as follows:

```
Edition     isLike  Emp
Volume      isLike  Dep
isVolOf     isLike  isDepOf
EdNo        isLike  EmpNo
VolNo       isLike  DepNo
```

Note that $\mathcal{I}$ overlaps (on purpose) with the alphabet of **BOOKS**. However, this overlapping does not create problems because `Edition` and `EdNo` are interpreted as an entity name and an attribute name, respectively, in both alphabets (although not explicitly indicated in the above example).

5. Analogy Mappings Applied to Second Order Integrity Constraints

The examples we described in Section 4 transfer the integrity constraints of **EMPLOYEE** to **BOOKS** and **BOOKS4** simply by renaming the concept names used in such constraints, as dictated by the analogy mappings. We may fruitfully extend this simple renaming principle to higher order constraints thereby covering more interesting examples.

Consider, for example, an entity-relationship schema **EMPLOYEE2**, with three entity schemes, `Emp`, `FullTime`, and `PartTime`. Assume that `FullTime` and `PartTime` are mutually exclusive, total specializations of `Emp`, which is usually formalized as follows:

E1. `Emp = FullTime ∪ PartTime`
E2. `FullTime ∩ PartTime = ∅`

Analogy mappings will then be able to generate similar specialization hierarchies where an entity set is specialized into two mutually exclusive sets.

A better approach would be to introduce an entity-relationship schema **EMPLOYEE3**, with three entity schemes, `Emp`, `FullTime`, and `PartTime`, and a set

E3. $E = \{$`FullTime`, `PartTime`$\}$

and define second order constraints as follows:

E4. `Emp` $= \bigcup_{S \in E} S$

E5. $(\forall R \in E)(\forall S \in E)(R \neq S \Rightarrow R \cap S = \varnothing)$

This more elaborate formalization of mutual exclusion may indeed be remapped to other contexts, if we generalize alphabet and analogy mappings to consider sets of concepts, such as *E*. For example, we may define an alphabet $\mathcal{F}$, with five entity names, `Cat`, `Eletronics`, `Computers`, `Cameras`, `Phones`, and a set

E6. $C = \{$`Eletronics`, `Computers`, `Cameras`, `Phones`$\}$

and define an analogy mapping δ as follows:

```
Cat  isLike  Emp
C    isLike  E
```

By appropriately extending the concepts of the previous sections, we may create an entity-relationship schema **CATALOGUE** by analogy with **EMPLOYEE3** using $\mathcal{F}$ and δ. **CATALOGUE** will have the following two constraints, generated from E4 and E5 by replacing `Emp` by `Cat` and *E* by *C*:

E7. `Cat` $= \bigcup_{S \in C} S$

E8. $(\forall R \in C)(\forall S \in C)(R \neq S \Rightarrow R \cap S = \varnothing)$

indicating that `Cat` is specialized into mutually exclusive classes `Eletronics`, `Computers`, `Cameras`, and `Phones`, in view of the definition of *C* in E6.

6. CONCLUSION

We argued in favor of a database conceptual schema discipline that explores analogy mappings to reuse the structure and integrity constraints of conceptual models, stored in a repository. As a proof-of-concept, we implemented a knowledge base in SWI Prolog that captures the design strategy described in this paper.

7. ACKNOWLEDGMENTS

This work is partially supported by CNPq under grants 550250/2005-0, 551241/2005-5 and 311794/2006-8.

8. REFERENCES

[1] Barbosa, S.D.J.; Souza, C.S. (2001) "Extending software though metaphors and metonymies". *Knowledge-Based Systems*, 14, Elsevier Science, pp. 15-27.

[2] Blackwell, A.F. (2006) "The reification of metaphor as a design tool". *ACM Trans. on Computer-Human Interaction*, Vol. 13, Issue 4 (Dec. 2006), pp. 490-530.

[3] Catarci, T.; Costabile, M.F.; Matera, M. (1996) "Which Metaphor for Which Database?". In: Proc. HCI'95 Conf. on People and Computers. Huddersfield, UK, pp. 151-165.

[4] Chen, P. P-S. (1976) "The entity-relationship model—toward a unified view of data". *ACM Transactions on Database Systems*, Vol. 1, Issue 1 (March 1976), p. 9–36.

[5] Goguen, J. (1999) "An introduction to algebraic semiotics, with applications to user interface design". In: Nehaniv, C (ed.) *Computation for Metaphors, Analogy and Agents*. LNAI, Vol. 1562. Springer, pp. 242-291.

[6] Holyoak, K.J.; Thagard, P. (1995) *Mental Leaps – Analogy in Creative Thought*. The MIT Press, Cambridge, MA, USA.

[7] Lakoff, G.; Johnson, M. (1980) *Metaphors We Live By*. The University of Chicago Press, Chicago, 1980.

[8] Lippert, M.; Schmolitzky, A.; Züllighoven, H. (2003) "Metaphor Design Spaces". In: Proc. 4th Int. Conf. Extreme Programming and Agile Processes in Software Engineering, XP 2003. Genova, Italy, May 25-29, 2003.

[9] Winston, P.H. (1980) "Learning and reasoning by analogy". *Communications of the ACM*, 23, pp. 689-703.