Entity Relationship modeling from an ORM perspective: Part 3

Terry Halpin
Microsoft Corporation

Introduction

This article is the third in a series of articles dealing with Entity Relationship (ER) modeling from the perspective of Object Role Modeling (ORM). Part 1 provided a brief overview of the ER approach, and then covered the basics of the Barker ER notation. Part 2 completed the examination of the Barker ER notation by discussing verbalization, exclusion constraints, frequency constraints, subtyping and non-transferable relationships. Both parts compared the Barker notations with the corresponding ORM notations This article discusses the Information Engineering notation for ER, relating it to relevant ORM constructs.

The *Information Engineering* (IE) approach began with the work of Clive Finkelstein in Australia, and CACI in the UK, and was later adapted by James Martin. Different versions of IE exist, with no single standard. In one form or other, IE is supported by many data modeling tools, and is one of the most popular notations for database design.

Entity types, attributes and associations

In the IE approach, *entity types* are shown as named rectangles, as in Figure 1(a). A*ttributes* are often displayed in a compartment below the entity type name, as in Figure 1(b), but are sometimes displayed separately (e.g. bubble charts). Some versions support basic constraints on attributes (e.g. Ma/Op/Unique).

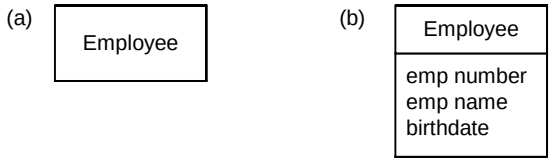

Figure 1 Typical IE notation for (a) entity type and (b) entity type with attributes

Relationships are typically restricted to binary associations only, which are shown as named lines connecting the entity types. As with the Barker notation, a half-line or line end corresponds to a role in ORM. Optionality and cardinality settings are indicated by annotating the line ends. To indicate that a role is *optional*, a circle "○" is placed at the other end of the line, signifying a minimum participation frequency of 0. To indicate that a role is *mandatory*, a stroke "|" is placed at the other end of the line, signifying a minimum participation frequency of 1. After experimenting with some different notations for a cardinality of "*many*", Finkelstein settled on the intuitive crow's foot symbol suggested by Dr. Gordon Everest.

In conjunction with a minimum frequency of 0 or 1, a stroke "|" is often used to indicate a maximum frequency of 1. With this arrangement, the combination "○|" indicates "*at most one*" and the combination "||" indicates "*exactly one*". This is the convention that I will use in this section. However different IE conventions exist. For example, some assume a maximum cardinality of 1 if no crows foot is used, and hence use just a single "|" for "exactly one". Clive Finkelstein uses the combination "○|" to mean "optional but will become mandatory", which is really a dynamic rather than static constraint—this can be combined with a crow's foot. Some conventions allow a crow's foot to mean the minimum (and hence maximum) frequency is many. So if you are using a version of IE, you should check which of these conventions applies.

Figure 2 shows a simple IE diagram and its equivalent ORM diagram. With IE, as you read an association from left to right, you verbalize the constraint symbols at the right-hand end. Here, each employee occupies exactly one (at least 1 and at most 1) room. Although inverse predicates are not always supported in IE, you can supply these yourself to obtain a verbalization in the other direction. For example:

"Each room is occupied by zero or more employees". As with the Barker notation, a plural form of the entity type name is introduced to deal with the many case.

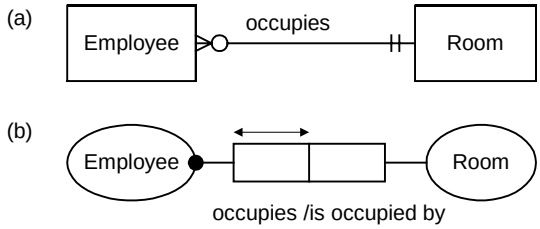

Figure 2 The IE diagram (a) is equivalent to the ORM diagram (b)

The IE notation is similar to the Barker notation in showing the maximum frequency of a role by marking the role at the other end. But unlike the Barker notation, the IE notation shows the optionality/mandatory setting at the other end as well. In this sense, IE is like UML (even though different symbols are used). As discussed in an earlier article, there are sixteen possible constraint patterns for optionality and cardinality on binary associations. Figure 3 shows eight cases in IE notation together with the equivalent cases in ORM.

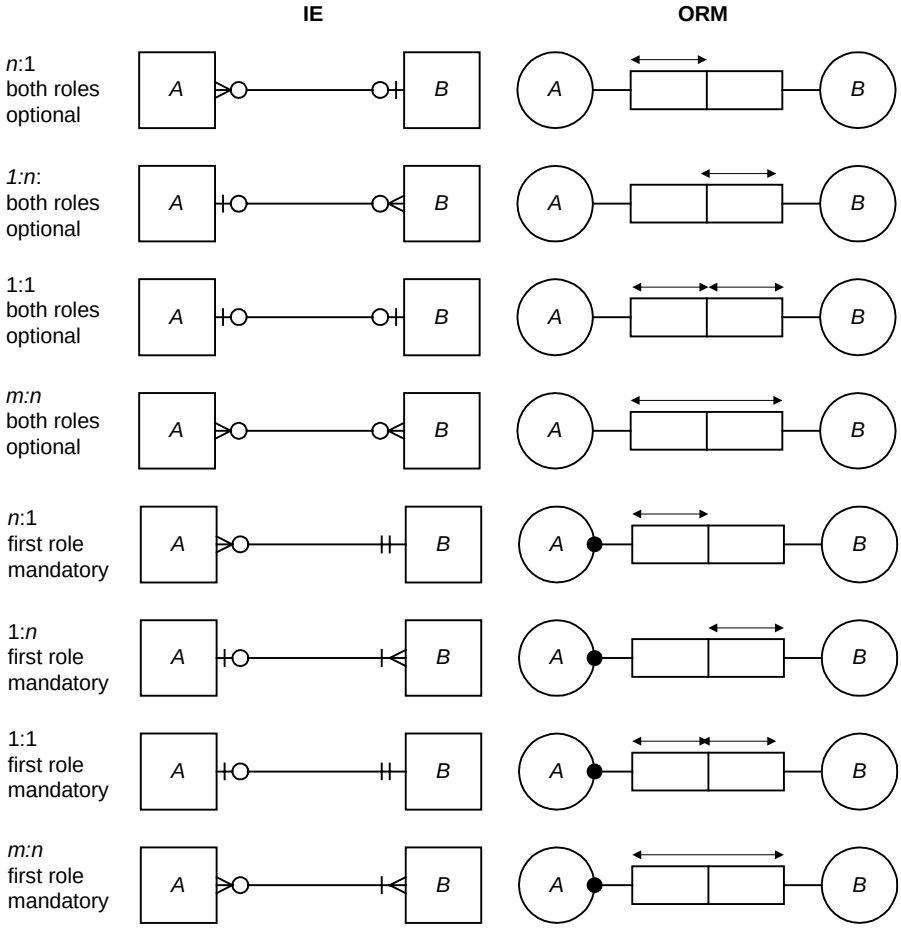

Figure 3 Some equivalent constraint patterns

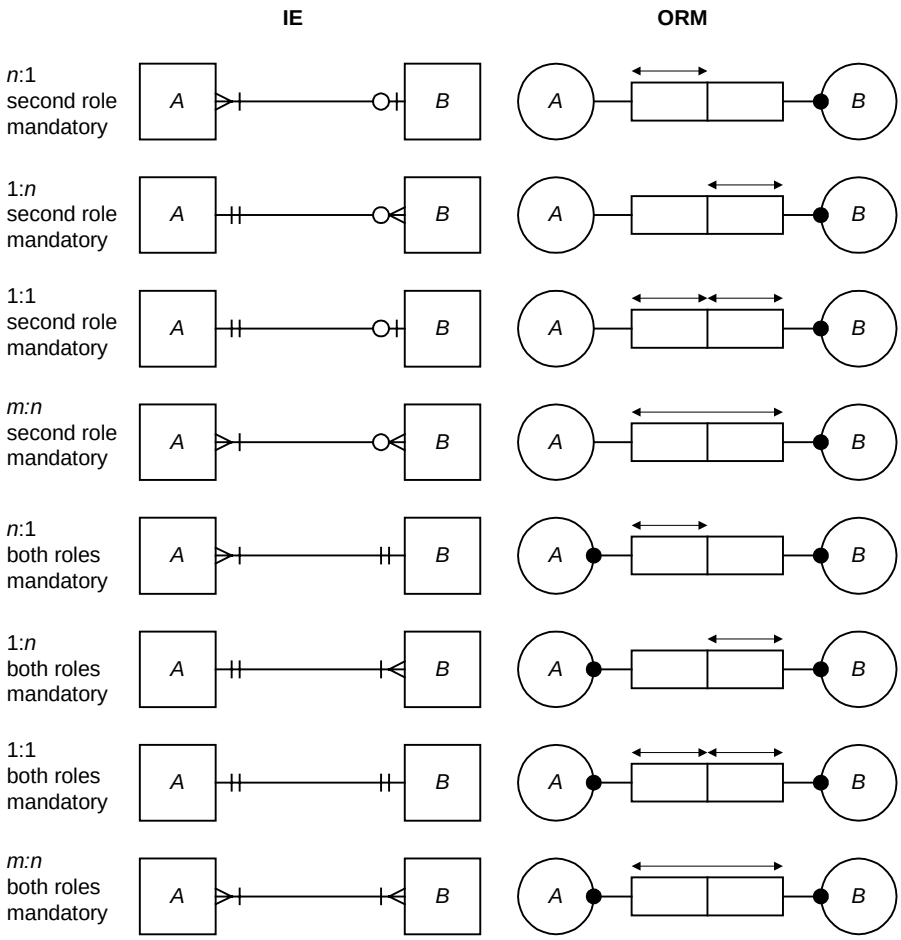

Figure 4 Other equivalent cases

The other eight cases are shown in Figure 4. An example using the different notation for IE used by Finkelstein is shown in Figure 5. Here the single bar on the left end of the association indicates that each computer is located in exactly one office. The circle, bar and crow's foot on the right end of the association collectively indicate that each office must eventually house one or more computers. This "optional becoming mandatory" constraint has no counterpart in ORM, and is not supported in most IE modeling tools.

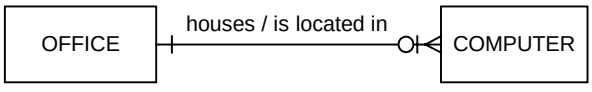

Figure 5 Finkelstein's notation for IE is different from ours

Some modeling tools support the IE notation for *n*:1, 1:*n* and 1:1 associations but not *m:n* (many to many) associations. For such tools, four of the sixteen cases cannot be directly represented. In this situation, you can model the *m:n* cases indirectly by introducing an "*intersection entity type*" with mandatory *n*:1 associations to the original entity types. For example, the *m:n* case with both roles optional may be handled by introducing the object type *C* as shown in Figure 6 for both IE and ORM. In ORM, *C* is a *co-referenced* object type, and the transformation is an instance of a flatten/coreference equivalence.

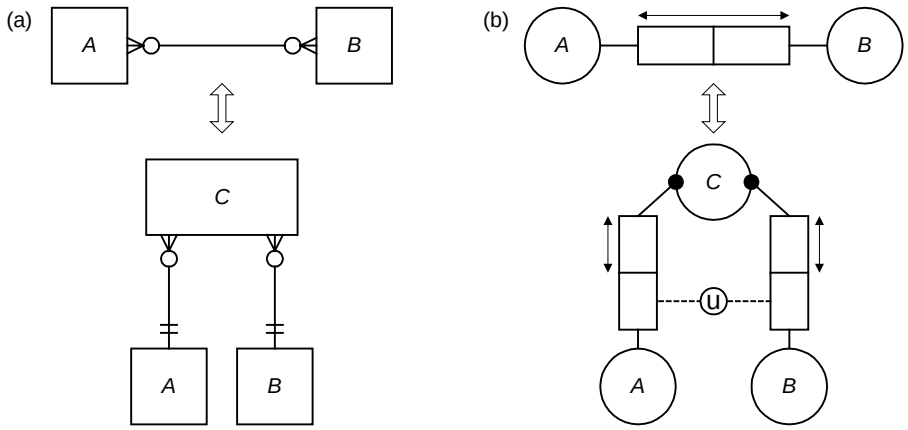

Figure 6 An *m:n* association remodeled as an entity type with *n*:1 associations

As an example, the *m:n* association Person plays Sport can be transformed into the mandatory *n*:1 associations: Play is by Person; Play is of Sport. However such a transformation is often very unnatural, especially if nothing else is recorded about the co-referenced object type. So any truly conceptual approach must allow *m:n* associations to be modeled directly.

Advanced constraints and subtyping

Some versions of IE support an *exclusive-or constraint*, shown as a black dot connected to the alternatives. Figure 7(a) depicts the situation where each employee is allocated a bus pass or parking bay, but not both. The equivalent ORM schema is shown in Figure 7(b). Unlike ORM, IE does not support an inclusive-or constraint. Nor does it support exclusion constraints over multi-role sequences.

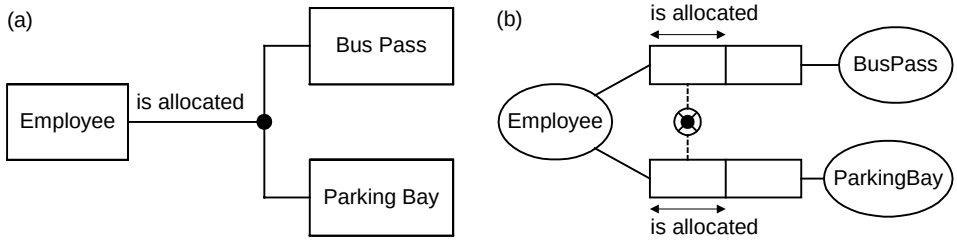

Figure 7 An exclusive-or constraint in (a) IE and (b) ORM

Subtyping schemes for IE vary. Sometimes Euler diagrams are used, adding a blank compartment if needed for "Other". Sometimes directed acyclic graphs are used, possibly including subtype relationship names and optionality/cardinality constraints. Figure 8 show three different subtyping notations for partitioning Patient into the subtypes MalePatient and FemalePatient. There is no formal support for subtype definitions, and no provision for context-dependent reference. Multiple inheritance may or may not be supported, depending on the version.

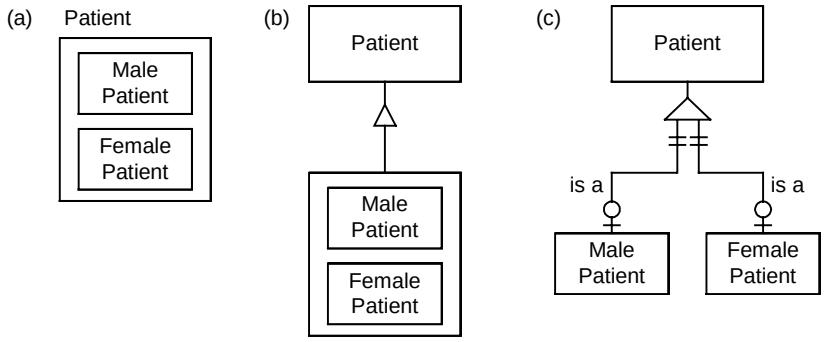

Figure 8 Some different IE notations for subtyping

Conclusion

Although far less expressive than ORM, IE does a good job of covering basic constraints. If you ever need to specify a model in IE notation, I suggest you first do the model in ORM, then map it to IE and make a note of any rules that can't be expressed there diagrammatically. Often referred to as "the father of information engineering", Clive Finkelstein is an amiable Aussie who is still actively engaged in the information engineering discipline. He developed a set of modeling procedures to go with the notation, extended IE to Enterprise Engineering (EE), and recently began applying data modeling to work in XML (Extensible Markup Language). An overview of IE by Clive can be found in [1], and details on his recent work are given in [2]. For a look at the IE approach used by James Martin, see [3]. Martin's more recent books tend to use the UML notation instead. In practice however, IE is used far more extensively for database design than is UML, which is mostly used for object-oriented code design.

Next issue

The next article in this series discusses the IDEF1X notation.

References

1. Finkelstein, C. 1998, 'Information engineering methodology', *Handbook on Architectures of Information Systems*, eds. P. Bernus, K. Mertins & G. Schmidt, Springer-Verlag, Berlin, Germany, pp. 405-427.
2. Finkelstein, C. & Aiken, P. 2000, *Building Corporate Portals with XML*, McGraw-Hill, New York.
3. Martin, J. 1993, *Principles of Object Oriented Analysis and Design*, Prentice Hall, Englewood Cliffs, NJ.