Appendix C: Entity Relationship Diagram for Electronic Resource Management

Nathan D. M. Robertson, Ivy Anderson, Adam Chandler, Sharon E. Farb, Timothy Jewell, Kimberly Parker, and Angela Riggio

Introduction

This document is an entity relationship diagram (ERD) for a system to manage e-resources. An ERD is a model that identifies the concepts or entities that exist in a system and the relationships between those entities. An ERD is often used as a way to visualize a relational database: each entity represents a database table, and the relationship lines represent the keys in one table that point to specific records in related tables. ERDs may also be more abstract, not necessarily capturing every table needed within a database, but serving to diagram the major concepts and relationships. This ERD is of the latter type, intended to present an abstract, theoretical view of the major entities and relationships needed for management of e-resources. It may assist the database design process for an ERM system, but does not identify every table that would be necessary for an e-resource management database.

This ERD should be examined in close consultation with other components of the *Report of the DLF Electronic Resource Management Initiative*, especially Appendix D (Data Element Dictionary) and Appendix E (Data Structure). The ERD presents a visual representation of e-resource management concepts and the relationships between them. The Data Element Dictionary identifies and defines the individual data elements that an e-resource management system must contain and manage, but leaves the relationship between the elements to be inferred by the reader. The Data Structure associates each data element with the entities and relationships defined in the ERD. Together, these three documents form a complete conceptual data model for e-resource management.

Understanding the Model

There are several different modeling systems for entity relationship diagramming. This ERD is presented in the "Information Engineering" style. Those unfamiliar with entity relationship diagramming or unfamiliar with this style of notation may wish to consult the following section to clarify the diagramming symbology.

ENTITIES

Entities are concepts within the data model. Each entity is represented by a box within the ERD. Entities are abstract concepts, each representing one or more instances of the concept in question. An entity might be considered a container that holds all of the instances of a particular thing in a system. Entities are equivalent to database tables in a relational database, with each row of the table representing an instance of that entity.

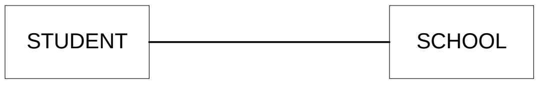
Remember that each entity represents a container for instances of the thing in question. The diagram below has an entity for *student* and another for *school.* This indicates that the system being modeled may contain one or more students and one or more schools.

STUDENT
SCHOOL

So far, no relationship between students and schools has been indicated.

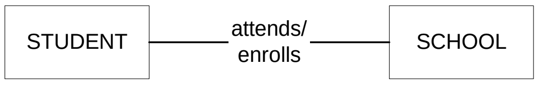
RELATIONSHIPS

Relationships are represented by lines between entities. Relationship lines indicate that each instance of an entity may have a relationship with instances of the connected entity, and vice versa.



The diagram above now indicates that students may have some relationship with schools. More specifically, there may be a relationship between a particular student (an instance of the student entity) and a particular school (an instance of the school entity).

If necessary, a relationship line may be labeled to define the relationship. In this case, one can infer that a student may attend a school, or that a school may enroll students. But if necessary, this relationship could be labeled for clarification:



Read the first relationship definition, *attends*, when tracing the relationship left to right or top to bottom. Read the second definition, *enrolls*, when tracing the relationship right to left or bottom to top.

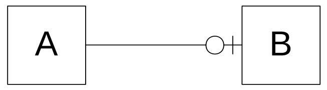
OPTIONALITY AND CARDINALITY

Symbols at the ends of the relationship lines indicate the *optionality* and the *cardinality* of each relationship. Optionality expresses whether the relationship is optional or mandatory. “Cardinality” expresses the maximum number of relationships.

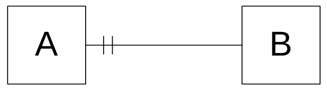
As a relationship line is followed from an entity to another, near the related entity two symbols will appear. The first of those is the optionality indicator. A circle (○) indicates that the relationship is optional—the minimum number of relationships between each instance of the first entity and instances of the related entity is zero. One can think of the circle as a zero, or a letter “O” for optional. A stroke (|) indicates that the relationship is mandatory—the minimum number of relationships between each instance of the first entity and instances of the related entity is one.

The second symbol indicates cardinality. A stroke (|) indicates that the maximum number of relationships is one. A crow's-foot (<) indicates that many such relationships between instances of the related entities might exist.

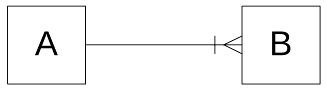
The following diagram indicates all of the possible combinations:



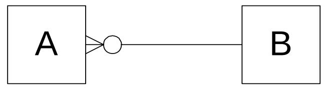
Each instance of A is related to a minimum of zero and a maximum of one instance of B



Each instance of B is related to a minimum of one and a maximum of one instance of A

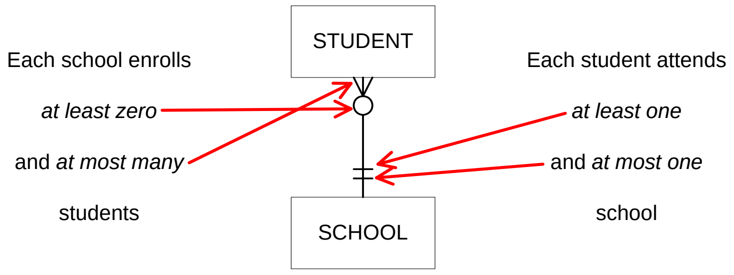


Each instance of A is related to a minimum of one and a maximum of many instances of B



Each instance of B is related to a minimum of zero and a maximum of many instances of A

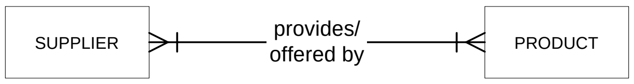
In our model, we wish to indicate that each school may enroll many students, or may not enroll any students at all. We also wish to indicate that each student attends exactly one school. The following diagram indicates this optionality and cardinality:



It is important to note that relationship optionality and cardinality constraints apply specifically to the system being modeled, not to all possible systems. According to the example modeled above, a school might not enroll any students—that relationship is optional. A school without students is not much of a school, and indeed if the system being modeled were a school system enrollment database, the relationship would probably be mandatory. However, if the system being modeled is an extracurricular honors program, there may be schools that have no students currently participating in the program. Consider the function of the system and consult the other documents in the data model to clarify modeling decisions.

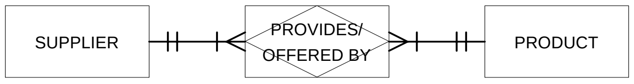
BRIDGE ENTITIES

When an instance of an entity may be related to multiple instances of another entity and vice versa, that is called a *many-to-many relationship.* In the example below, a supplier may provide many different products, and each type of product may be offered by many suppliers:



While this relationship model is perfectly valid, it cannot be translated directly into a relational database design. In a relational database, relationships are expressed by keys in a table column that point to the correct instance in the related table. A many-to-many relationship does not allow this relationship expression, because each record in each table might have to point to multiple records in the other table.

In order to build a relational database that captures this relationship, it is necessary to build a bridge between the two entities that uniquely expresses each relationship instance. This can be modeled in an ERD with a *bridge entity*, an entity box containing a diamond, which may replace the many-to-many relationship. (The diamond is used in other ER modeling systems to indicate relationships, or may be viewed as the joining—the bridge—of the many-to-many crow's-feet).

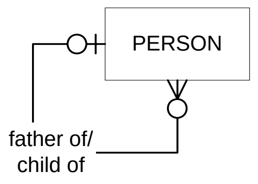


This diagram expresses the same relationship as the diagram above. Each instance of the *provides* bridge entity indicates that a certain supplier can provide a certain product.

In addition to explicitly depicting a relational database structure that can capture a many-to-many relationship, the bridge entity has an additional function in abstract entity-relationship modeling: A bridge entity may capture attributes that are specific to the relationship between instances of the bridged entities. In the supplier and product example, a product does not have an inherent cost; it only has a cost in relation to the supplier who sells it. Similarly, a supplier may not have a uniform delivery time; delivery times may vary in relation to the product being delivered. Any attributes that are dependent on the relationship would be associated with the relationship's bridge entity.

RECURSIVE RELATIONSHIPS

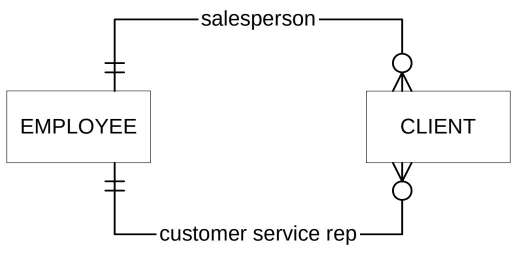
Instances of entities may have relationships with other instances of the same entity. These relationships may be drawn with relationship lines that begin and end connected to the same entity. Common occurrences of these recursive relationships include parent/child relationships:



The diagram above indicates that a person may be the father of zero or many persons, and that a person may have zero or one father. (Not every person's father will be recorded in the system, so the relationship is modeled as optional).

MULTIPLE RELATIONSHIPS BETWEEN ENTITIES

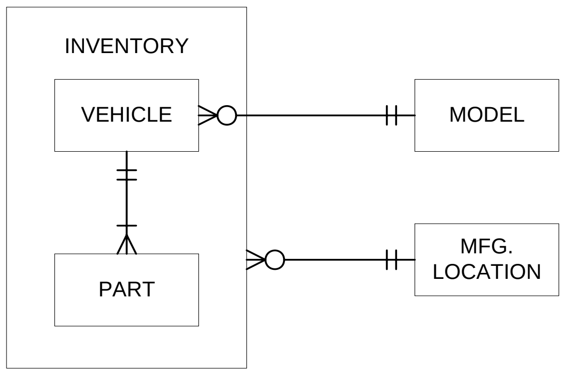
An entity may have multiple relationships with another entity. These are depicted in the ERD with multiple relationship lines connecting the two entities:



The diagram above indicates that an employee may be the salesperson assigned to zero or many clients, and an employee may be the customer service representative for zero or many clients. Each client has exactly one salesperson and exactly one customer service representative. Each client's salesperson may or may not be the same employee as the client's customer service representative; each relationship is treated independently.

ENTITY SUBTYPES

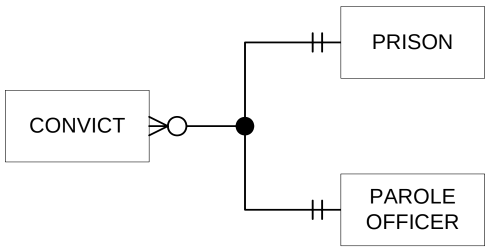
There are times when it is convenient to depict relationships that apply to an entire class of things, as well as depict relationships that apply only to certain types of the larger class. Entity subtypes accommodate these relationship depictions. In the ERD, entity subtypes may be depicted by entity boxes shown within larger entity boxes. The large entity represents the class, and the smaller boxes depict the subtypes.



The example above depicts an *inventory* entity, with the subtypes of *vehicle* and *part.* A vehicle has one or many parts, and every part is associated with one and only one kind of vehicle (according to this diagram, there are no interchangeable components). All items in inventory, whether they are vehicles or parts, have a manufacturing location, but only vehicles are of a particular model.

EXCLUSIVE-OR RELATIONSHIP

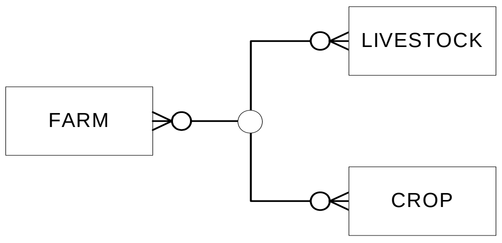
If an entity instance may have either one relationship or another, but not both, the constraint may be modeled with an *exclusive-or relationship,* represented as a tree with a solid dot where the tree branches. The entity attached to the trunk of the tree is the one subject to the exclusive-or constraint. No relationship is indicated between entities attached the branches.



The diagram above indicates that each convict is assigned to a prison, or to a parole officer, but not both. A prison may have zero or many convicts, a parole officer may have zero or many convicts, and there is no relationship between prisons and parole officers.

INCLUSIVE-OR RELATIONSHIP

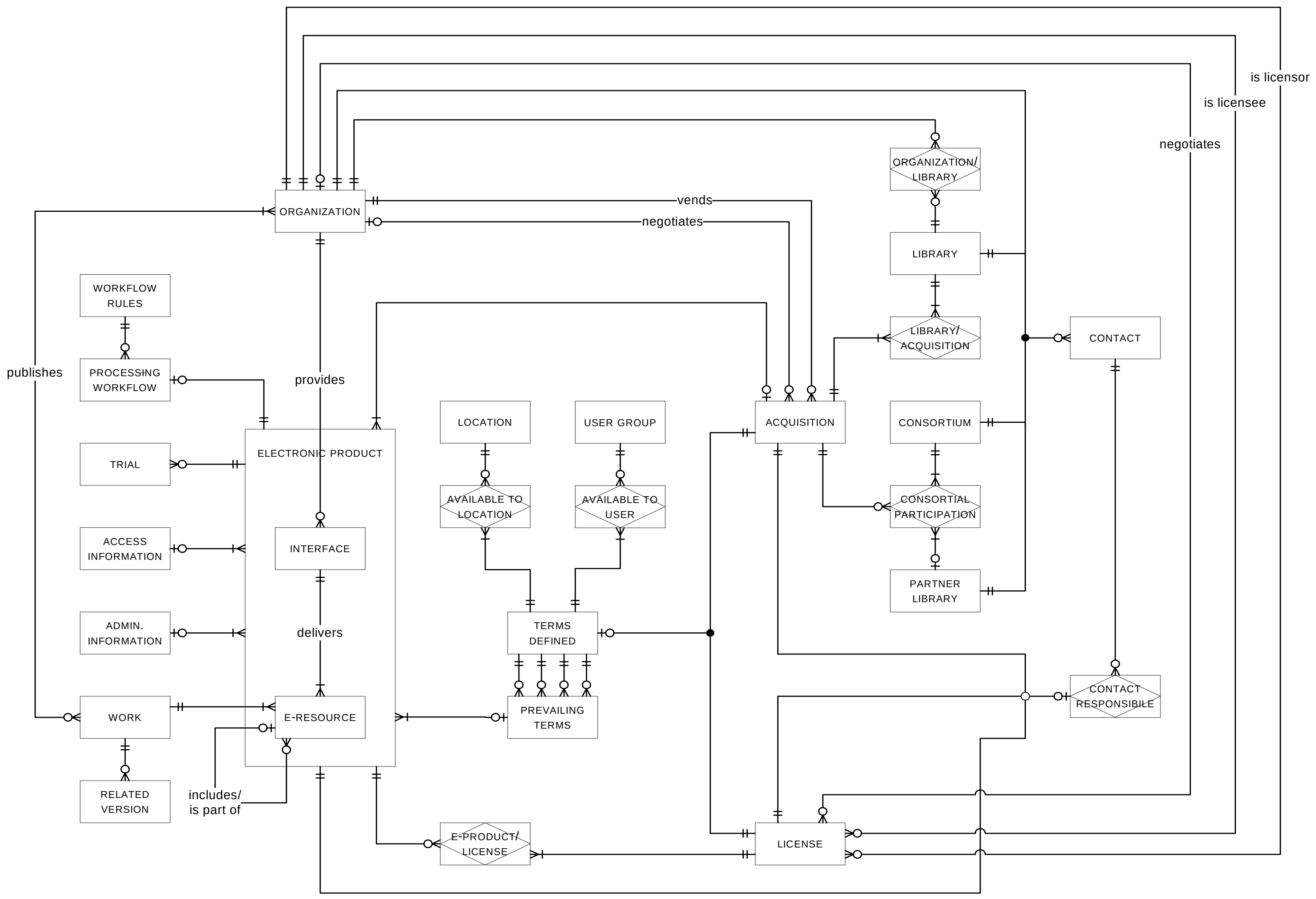
If an entity instance may have either one or more relationships, but must have at least one of the possible relationships, the constraint may be modeled with an *inclusive-or,* represented as a tree with a hollow dot where the tree branches. As with the exclusive-or constraint, the entity attached to the trunk of the tree is the one subject to the constraint, and the entities attached to the branches are not related to one another by this construction.



The diagram above indicates that a farm must have either livestock or a crop, or may have both. Any given type of livestock may belong on zero or many farms, and any given type of crop may be grown on zero or many farms.

Entity Relationship Diagram for Electronic Resource Management

The following diagram is the complete Entity Relationship Diagram for Electronic Resource Management. It presents an abstract, theoretical view of the major entities and relationships needed for management of e-resources:



is licensor
is licensee
negotiates
ORGANIZATION/ LIBRARY
ORGANIZATION
vends
negotiates
LIBRARY
WORKFLOW RULES
LIBRARY/ ACQUISITION
CONTACT
publishes
PROCESSING WORKFLOW
provides
ACQUISITION
ELECTRONIC PRODUCT
LOCATION
USER GROUP
CONSORTIUM
TRIAL
AVAILABLE TO LOCATION
AVAILABLE TO USER
CONSORTIAL PARTICIPATION
ACCESS INFORMATION
INTERFACE
PARTNER LIBRARY
ADMIN. INFORMATION
delivers
TERMS DEFINED
CONTACT RESPONSIBILE
WORK
E-RESOURCE
PREVAILING TERMS
RELATED VERSION
includes/ is part of
E-PRODUCT/ LICENSE
LICENSE

Additional Inheritance Constraints

Unlike other modeling systems, entity relationship diagrams (ERDs) do not explicitly depict situations in which entity instances are expected to inherit attributes or relationships of related entity instances.

This data model assumes the following relationship inheritance constraints:

• An e-resource may inherit its interface's relationship with an acquisition, administrative information, or access information. Alternatively, the e-resource may have a separate acquisition, administrative information, and/or access information from its interface.

• An e-resource must inherit its interface's relationship or relationships with licenses (and with the terms defined by the interface's licenses—although different terms from a different license may prevail for the e-resource).

• An e-resource may inherit its parent e-resource's relationship with an acquisition, administrative information, or access information. Alternatively, the child e-resource may have a separate acquisition, administrative information, and/or access information from its parent.

• An e-resource must inherit its parent e-resource's relationship or relationships with licenses (and with the terms defined by the parent's licenses—although different terms from a different license may prevail for the child e-resource).

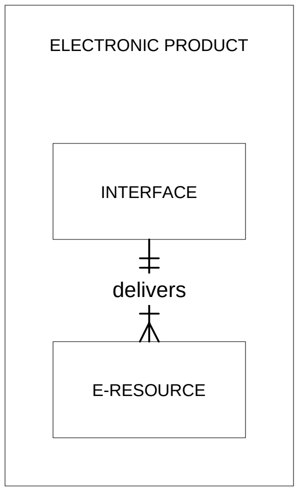
Explanation of Complex Relationships or Concepts

Although the overall diagram is large and complex, most of the entities and relationships depicted in the ERD are relatively straightforward, and can be fully understood in consultation with the definitions provided in Appendix D (Data Element Dictionary) and Appendix E (Data Structure).

Several of the relationships depicted merit further explanation.

ELECTRONIC PRODUCT AND ITS SUBTYPES

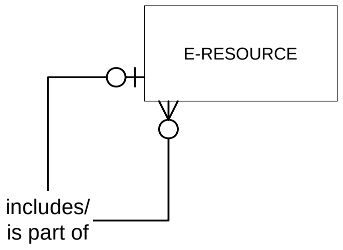
The *electronic product* entity represents electronic things that may be acquired, licensed, or managed. Electronic product encompasses interfaces, e-resource packages, packages of packages, and individual e-resource titles.



Many attributes and relationships are shared by all electronic products, regardless of whether the electronic product is an interface, package, or individual title. For example, all types of electronic products may be licensed, may be acquired, may undergo a product trial, etc. On the other hand, interfaces and e-resources do have some unique characteristics and in certain cases need to be handled in distinct ways. The electronic product entity and its subtypes accommodate the depiction of these relationships.

E-RESOURCE RECURSIVE RELATIONSHIP

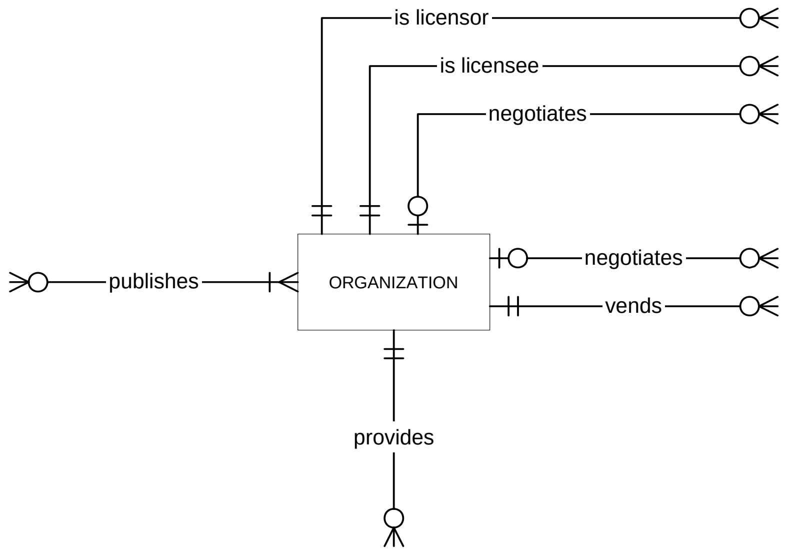
The *e-resource* entity represents e-resource packages, packages of packages, and individual e-resource titles. Any given e-resource title may be a standalone product, or may be a part of an e-resource package. An e-resource package could in turn be part of a larger package. These possible relationships are accommodated with the optional recursive relationship for e-resources:



THE ORGANIZATION ENTITY

The *organization* entity represents any business, vendor, provider, publisher, licensor, etc. with which a library does business related to electronic products. Generally, one records the same attributes of an organization, such as name and address, regardless of the role the organization plays. Furthermore, a single organization frequently fills several roles in the e-resource management environment, functioning as vendor, licensor, publisher, and/or provider.

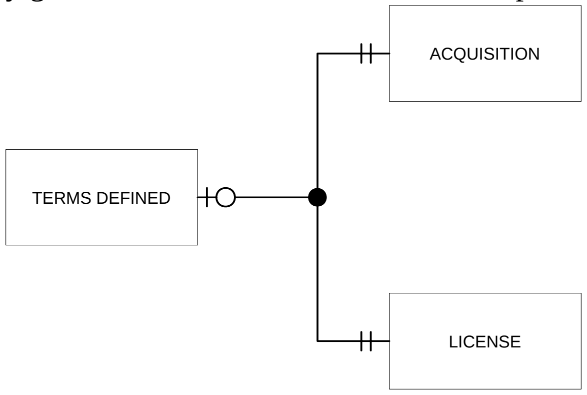
Rather than create separate entities for each role, which may contain duplicate instances of one another, the model contains the single organization entity, with the role or roles of the organization dependent on the relationship or relationships in which the organization participates:



For definitions of the roles indicated by these relationships, see Appendix D (Data Element Dictionary) and Appendix E (Data Structure).

TERMS DEFINED AND PREVAILING TERMS

Each license and each acquisition defines (or may define) a set of associated rights and restrictions on the usage and management of the electronic products covered by the license or acquisition. In some cases, certain terms are defined in licenses, while in other agreements those same term concepts might be defined in the business agreement. To accommodate the uncertainty of where certain rights, responsibilities, and other terms might be defined in any given situation, we have modeled a separate *terms defined* entity:

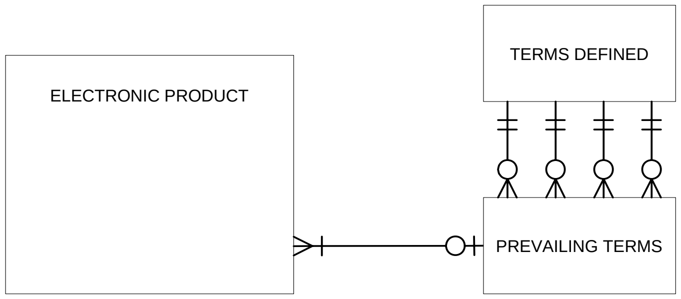


The terms defined entity is able to record all possible rights, responsibilities, and other terms, including terms that are typically defined in licenses (such as “interlibrary loan” or “scholarly sharing” permissions), as well as terms that are typically defined in acquisitions or business terms (such as “number of concurrent users”).

Each acquisition may define at most one terms defined instance. Each license may define at most one terms defined instance. Each terms defined instance is defined either by one acquisition or by one license; this relationship is modeled using the exclusive-or relationship.

Use, rights, and restrictions for any given e-resource may be governed by multiple agreements, both legal license terms and negotiated acquisition terms. Each acquisition defines one set of terms, and each license defines one set of terms. Through each electronic product's relationship with an acquisition and its relationship with one or more licenses, one can find all of the terms defined instances that apply to each e-resource.

It may occur that among several terms defined instances that apply to a given electronic product, some of the individual terms are in conflict; this is especially likely in situations where multiple license agreements cover a single electronic product. In such cases it is necessary to determine which of the conflicting terms prevail for the electronic product in question. This is modeled with the *prevailing terms* entity, and its relationships with the terms defined entity:



The prevailing terms entity represents all possible terms, just as the terms defined entity does, but for each term merely points to the terms defined entity that prevails. The four relationship lines depicted in the drawing actually represent many relationships: one for each defined term. So while each electronic product may be governed by many different acquisitions and licenses that define terms, each electronic product is governed by at most one complete set of prevailing terms.

Conclusions

This entity relationship diagram depicts the major concepts and relationships needed for managing e-resources. It is neither a complete data model depicting every necessary relational database table, nor is it meant to be a proscriptive design for implementations of ERM systems. Alternate models may capture the necessary attributes and relationships. Hopefully this document, along with the other components of the *Report of the DLF Electronic Resource Management Initiative* will assist developers with envisioning the complexity of the environment that an ERM system must address, and ensure that crucial relationships and features will be included in ERM products.