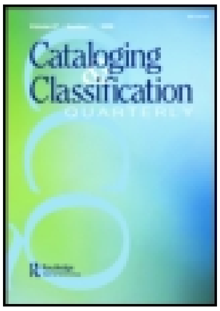

Cataloging & Classification Quarterly
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/wccq20

Item, Document, Carrier: An Object Oriented Approach

Norberto Manzanos [a]
[a] Instituto de Investigaciones en Humanidades y Ciencias Sociales (IdIHCS), FaHCE/UNLP-CONICET, La Plata, Argentina

Available online: 06 Jun 2012

To cite this article: Norberto Manzanos (2012): Item, Document, Carrier: An Object Oriented Approach, Cataloging & Classification Quarterly, DOI:10.1080/01639374.2012.681274

To link to this article: http://dx.doi.org/10.1080/01639374.2012.681274

Cataloging & Classification Quarterly, 00:1–17, 2012
Copyright © Taylor & Francis Group, LLC
ISSN: 0163-9374 print / 1544-4554 online
DOI: 10.1080/01639374.2012.681274

Item, Document, Carrier: An Object Oriented Approach

NORBERTO MANZANOS
Instituto de Investigaciones en Humanidades y Ciencias Sociales (IdIHCS), FaHCE/UNLP-CONICET, La Plata, Argentina

I discuss the concept of Item as stated by the International Federation of Library Associations and Institutions (IFLA) in the conceptual model Functional Requirements for Bibliographic Records (FRBR) and the object-oriented version of it (FRBR$_{OO}$). Using object-oriented modeling techniques I analyze the relationship of the Item with the Manifestation entity, the concept of Document*, and the physical object as a* Carrier *of a* Content*. A class scheme is proposed, not only as an implementation example, but as a way of clarifying some bibliographic concepts as well.*

KEYWORDS object orientation, object technology, objects paradigm, FRBR, conceptual models, FRBR$_{ER}$, FRBR$_{OO}$

INTRODUCTION

Establishing a conceptual framework for an object model based on Functional Requirements for Bibliographic Records (FRBR) has a double difficulty derived from the two disciplines involved—Computer Science and Library and Information Science (LIS). Neither of them provides a theoretical foundation that arises unanimously from each professional community. This fact is evidenced by the different names, even within the same language, given to each discipline.[1] The use of the term *paradigm* to refer to what is also called *object orientation* indicates the existence of different professional areas within computer science that presumably are based on different assumptions. But even standing within the object paradigm it is not possible to establish basic axioms: there are different assumptions that have

Received October 2011; revised January 2012; accepted March 2012.
The author thanks his colleague Fernando Gómez for help in reviewing and correcting this article.

Address correspondence to Norberto Manzanos, IdIHCS, FaHCE/UNLP-CONICET, La Plata, Buenos Aires 1900, Argentina. E-mail: nmanzanos@gmail.com

counterparts in different technical implementations. The inheritance mechanism, derived from logic of classes, which sometimes is referred to as an inherent part to object orientation, has at least three ways of being conceived: inheritance in the original sense, which follows the model of classical taxonomy, in which each class has a single superclass from which it inherits; multiple inheritance, in which a class inherits from more than one superclass; and prototypes, in which there is no inheritance at all. It is not even possible to apply the pragmatic criteria that usually define the discussions in Computer Science: efficiency, popularity of a language, the extent of its use and the size and persistence of its user communities, and so on; there are examples in all three cases of popular languages (Java, C++, and JavaScript), highly efficient products with large and very active professional communities. In addition, many times in real-world programming practice objects are considered as data structures, which can coexist with others. This is the case of the so-called hybrid languages such as PHP or Delphi. Although it lacks a theoretical foundation, this fourth group can compete with the other languages in terms of popularity, number of users in the community, and products.

In the case of LIS, we could rest in the principles of cataloging,[2] although the apparent existence of different paradigms in the discipline was noted.[3] But in the case of the FRBR family we face a similar scene to that of Computer Science: the models that inspired this work (mainly FRBR_{ER} and FRBR_{OO}) are based more on computing paradigms than on those principles: in FRBR_{ER}, the entity-relationship model, in FRBR_{OO}, the object orientation with multiple inheritance. In addition, we find another difficulty: the absence in such models of a clearly defined theoretical framework that might justify the technological choices. FRBR_{ER} does not define which are the traits that make one element of the bibliographic universe to be considered an entity or an attribute. It has been mentioned that names should be entities instead of attributes,[4] a need demonstrated by Functional Requirements for Subject Authority Data (FRSAD) and Functional Requirements for Authority Data (FRAD), where entities are defined for different types of names. Neither FRBR_{OO} nor International Committee for Documentation (CIDOC) Conceptual Reference Model (CRM), on which the first is based, make explicit what they mean by *object* and what is the reason for choosing multiple inheritance instead of single inheritance as a modeling technique.

The need for a uniform conceptual model for the bibliographic universe and the use of information technology as a tool necessarily result in another problem: if there is any agreement within the Computer Science community it is that *a priori* designs should be contrasted with reality. Flow charts, Unified Modeling Language (UML) diagrams, and Entity-Relationship (ER) schemes do not ensure the sustainability of a model that has not been observed in a real implementation and tested by domain experts. The cycles of the software are always iterative and circular: the observation and testing can lead to a modification of the model, which in turn can cause changes in the implementation. But in this case we are not dealing with a clearly delimited domain, as it is often the case with usual computer applications, but rather a domain of an extension that goes to infinity, as is the case of the bibliographic universe.

For all the above, our proposal should necessarily be based on assumptions that are not intended to be universally valid, since they are not the foundation of any well-established knowledge. I am not suggesting that the proposed solution is the only one or the best. Rather, this work should be considered just a proposal for other implementations. A good object design needs conceptual clarity, so I will cover some theoretical aspects that justify the decisions made. Theoretical analysis could entail changes in models, not at the requirements level, but at the conceptual level, as a consequence of the mentioned characteristics of software lifecycle.

From the Computer Science point of view, the conceptual framework of this study is the object-orientation paradigm with simple inheritance and the use of design patterns.[5] As already mentioned, the inheritance mechanism itself has been criticized, particularly because of the so-called "diamond problem" in multiple inheritance, which is also present in single inheritance, generalized as the *common ancestor problem*.[6] Sometimes it is preferable to replace inheritance, single or multiple, by delegation. "Delegation is an implementation mechanism in which an object forwards or *delegates* a request to another object."[7] As discussed below, the use of delegation instead of inheritance is not just a technical shortcut or a way to simulate or replace multiple inheritance but, rather, a way to discover new entities. In object-oriented design it is not as important to establish a solid class hierarchy as it is to let it emerge from an analysis of the expected behavior objects must carry out; it is expected that this behavior shall be homologous to the domain entities in real world.

On the one hand, bibliographic catalogs are no longer mere collections of data because the information they carry travels beyond their original boundaries. On the other hand, the entities of the bibliographic universe are present in many other places besides the catalogs: one could say that almost everything that goes on the Web is a bibliographic entity. Besides, what has traditionally been understood by *user* is not the same as before: it is no longer possible to think of *user* as anything other than an Internet user, which at least in theory, is equal to human species. This new reality leads us to model the bibliographic universe with a technology that provides ways of conceiving our domain dynamically. In terms of object orientation this means thinking less about entities, attributes, and data and more on interaction ways and processes (i.e., behaviors).

It is not possible in the space of this article to address all FRBR issues in the light of object technology, so I have selected two topics that are located at the ends of the model: first, the Work entity and the problems posed by the attributes specified in FRBR_{ER}, and secondly, the relationship between *Item* entity and the traditional concepts of bibliographic item and document, as well as their relationships with the physical object.

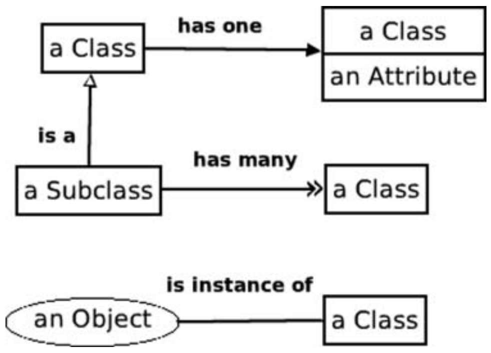

FIGURE 1 UML Graphics Used in This Article.

To simplify this exposition, I will use only literary and musical works as examples. I will use very simple class diagrams, detailed in Figure 1. Empty arrows mean an inheritance relationship; full arrows a composition relationship; double arrows mean a one-to-many cardinality. Only if it is necessary for the exposition I will include class attributes in a text box within the box representing the class. These diagrams can be interpreted as an entity-relationship, although this is not the idea, taking into account that inheritance is just a special kind of relationship.

THE ITEM IN FRBR$_{ER}$

Both FRBR$_{ER}$ and FRBR$_{OO}$ conceive the *Item* entity as equivalent to the physical object. FRBR$_{ER}$ defines *Item* as "a single exemplar of a manifestation that is 'the physical embodiment of an expression of a work.'"[8]

The item could be seen as a similar concept as that of document, but it has been argued that the notion of document is more extensive than the bibliographic item in the pre-FRBR sense of the word (i.e., in the sense of physical object).[9,10] It is clear that our culture places certain physical objects in a place of prominence. We recall, for example, the distinction made by Heidegger in his essay "The Origin of the Work of Art": things, tools, and artwork.[11]

Certain properties of the document, such as its itinerary among different collections, the places where it stayed, the incoming date, institutions or persons who have held it; all of these are not properties of physical objects, but only when considered as documents. While this itinerary impacts the physical object in the form of stamps, annotations, and so on each of these marks tells us more about the status of the object as collected, organized, and inventoried than about its physical history.[12]

FRBR$_{ER}$ clarifies that an *Item* can be a compound of multiple physical objects: the case of a multi-volume work. In the same way that a publishing house considers a multi-volume publication as merchandise like any other in all aspects except in their physical appearance, and to the extent that nobody believes they have read *Lord of the Rings* without having read all three volumes, we must conclude that we have a single manifestation exemplified by a single item. Should we think instead of a manifestation with three items? How then to consider the other sets of three volumes? A manifestation that always divides its items by three?

From the point of view of the library the *Item-Document* is a dual entity: while it may be formed by more than a physical object and this is shown, for example, in the inventory, it keeps its unity as a document: the call number is basically the same. Indeed, for FRBR_{ER}, although the opposite sometimes was said, the *Item* is not the physical object.

In addition, FRBR_{ER} conceived the item as an exemplar, a copy of a manifestation, implying that a manifestation is the embodiment of an expression in the form of multiple copies. This is faced with a problem already noted[13,14]: for some entities that should be considered works and expressions, the *Manifestation-Item* pair is not relevant: the case of visual arts—with the exception of arts like engraving—in which the physical object and the document match. Although the *Expression* entity can be abstracted, especially to represent other expressions than the original, as is the case of reproductions, repeating this abstraction process in relation to the *Manifestation-Item* is to force reality to fit the mold.

In another chapter of Taylor's book cited above,[15] Thurman notes another problem: for other types of materials such as archive documents, the triad of the more abstract FRBR entities makes no sense. An archive document does not (or does not necessarily) embody any expression of any work.

If we want to apply object-oriented modeling techniques to these concepts, defining a class for each of them, we will not find one definite scheme. If we part from a *Physical Object* class where we can locate all that has to do with physical traits of a material object (size, shape, weight, color, etc.) we would find different class hierarchies for each of the referred situations. For traditional library materials (books, serials, music, movies) we should consider the relationship as a composition: an *Item/Document* consists of one or more *Physical Objects*. If we have visual arts in mind, we should consider the *Item* class as a subclass of *Physical Object*, but this class, as an embodiment of an expression, is also a *Manifestation*. Finally, to represent an archive document, we should consider a *Document* class, with no relation with work, expression, manifestation, item (WEMI) entities, as a subclass of *Physical Object* (Figure 2).

THE ITEM IN FRBR_{OO}

Not being a general ontology, but a conceptual bibliographic model, FRBR_{er} does not require the definition of a physical object entity; but FRBR_{oo},

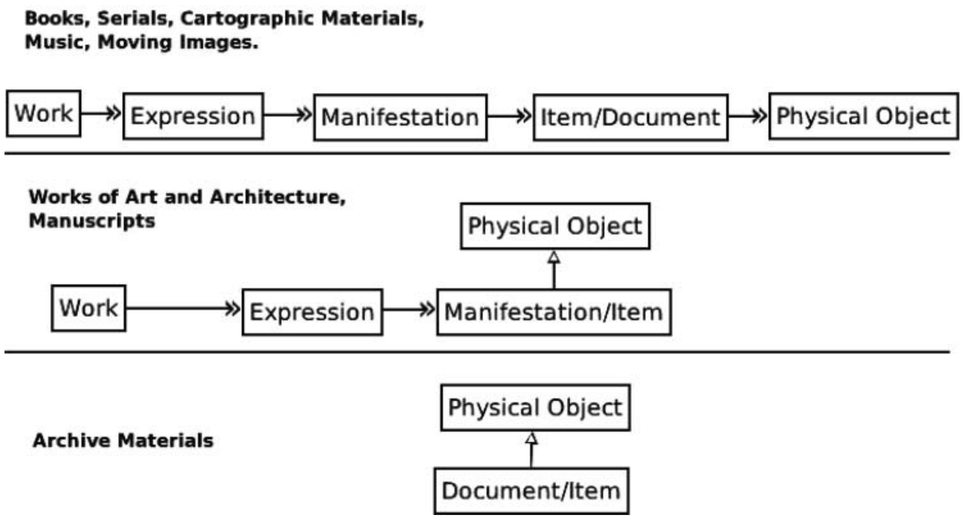

FIGURE 2 Different Class Schemes for Item/Document.

being coupled to an ontology such as CRM, whose domain is museology, in which the physical object plays a central role, needs to establish the relationship between both classes. FRBR_{OO} class *F5 Item* "comprises physical objects (printed books, scores, CDs, DVDs, CD-ROMS, etc.) that carry a *F24 Publication Expression* . . ."[16] It is a subclass of *E84 Information Carrier*, which in turn, skipping other intermediate classes, is a subclass of *E19 Physical Object*. This is in sharp contrast with FRBR_{ER}, in which, as we have seen, the item is not a physical object but a composite of several physical objects.

Instead, FRBR_{OO} solves another problem pointed out—the identity between *Manifestation* and *Item* in visual arts and manuscripts, splitting the *Manifestation* entity into two: *F3 Manifestation Product Type* and *F4 Manifestation Singleton*. Class *F3 Manifestation Product Type* "comprises the definitions of publication products. An instance of *F3 Manifestation Product Type* is the 'species', and all copies of a given publication are 'specimens' of it," which matches FRBR_{ER}'s *Manifestation* entity. Class *F4 Manifestation Singleton* "comprises physical objects that each carry an instance of *F2 Expression*, and that were produced as unique object."

Two classes in different hierarchies where FRBR_{ER} sees one entity, is an anomaly evidenced by their very names. If we have manifestation product types, on the one hand, and manifestation singletons, on the other, it is clear that both are "types" of manifestation, so they should share behavior in some way, which is usually accomplished by setting an inheritance relationship. An implementation that pretends to carry out this scheme should duplicate all the common behavior for both "types" of manifestation. Clearly what is missing here is what in object technology is called an *abstract class,* a class that cannot have instances, but which serves to provide a common

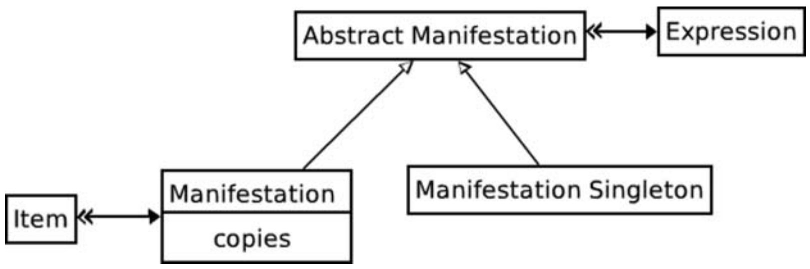

FIGURE 3 The Class Abstract Manifestation and Its Subclasses.

behavior for its subclasses. These subclasses are concrete and they have the responsibility of creating concrete instances.

Figure 3 shows the basic outline: the abstract class *Abstract Manifestation* possesses the common behavior, which should at least ensure that any manifestation is the embodiment of an expression, regardless of whether it represents a mass production of copies or a single copy. These particular behaviors are represented by the classes *Manifestation Singleton* and *Manifestation*.[17] To illustrate this I placed the attribute *copies* in class *Manifestation*. In terms of behavior, this means that instances of class *Manifestation* have the responsibility of "knowing" the amount of copies (items) produced.

But this hierarchy does not take into account the relationship between the entities *Document* and *Item*, nor the relationship of them with the physical object, nor can it include documents that are not part of the WEMI chain. Furthermore, if we want to keep compatibility with CRM we should use multiple inheritance, so that both *Item* and *Manifestation Singleton* inherit from *Physical Object*, keeping in the latter its inheritance with *Abstract Manifestation*.

And one last objection. Let us consider an example of a document as a physical object. Suppose a documentary collection that contains a recording on a CD with music performed only once. It would be clearly a *Manifestation Singleton*, for it embodies an expression of a work with no mass production involved. But as a physical object, it has the same features as a commercial CD, which in this case would be an item of a manifestation. If we wanted to represent the physical characteristics of the CD, beyond the common characteristics of any physical object, for example its duration, it would not be simple. We would be forced to repeat this behavior in both classes—*Manifestation Singleton* and *Item*. This would force the programmer to many branches (*if* statements) for addressing each particular case; that is, each type of physical object: it would not make sense to talk about the duration of a book. If the system incorporates new kinds of physical objects, which is very likely considering the speed of technological leaps, we should revisit the long list of *if*s. The same would apply to certain characteristics of books, such as the number of pages. And the program code for these branches should be repeated in each of the classes. An object designer would find that the solution for addressing these situations is to reify the physical object, that is, define classes for books, CDs, and so on, so that when facing any change it will not be necessary to modify the existing code, but just add new classes instead. But since both *Manifestation Singleton* and *Item* are physical objects, the only way is that these classes inherit from both. But this makes no sense, because a CD is either an *Item* or a *Manifestation Singleton*. So the only remaining possibility is to duplicate each class of *Physical Object*. According to the given example, a class *CD-Item* and another *CD-Manifestation Singleton* would be needed and all the specific behavior of the CD should be repeated on both. Clearly, because of that explosion of classes, this is not a good solution.

This need for reifying the physical object regardless of its intellectual content, just as physical object, must not be seen as an abuse of description or as an intention to merely keep compatibility with cataloging rules. As noted by Yee[18] "the paging of a book," that is, a feature of the physical object, "can be with some frequency the only reliable clue that two items are significantly different" and taking into account that certain types of users, bibliographers, "might be interested in physical evidence of the printing and publishing history of a work," while scholars "need to find a particular edition because they have a citation to a particular page number," the differences between physical objects can have a correlate in significant documentary differences that could be needed for organization and retrieval.

(RE) INTRODUCING THE DOCUMENT

As noted earlier, an item is not necessarily a single physical object. In the case of multi-volume works, all of them are information carriers of the same kind (e.g., books). But a CD consists of at least three different physical objects: the disc itself, the booklet and the plastic case. They are physical objects with different properties, which may suffer various alterations that perform different bibliographic functions: only if the disc is missing or damaged the work is no longer accessible; if the booklet is missing the work is still accessible but some referential information is lost; the box, instead, is easily replaceable.

If the item is not so much a specialized physical object, but a specific use and meaning we give to the physical object inside the bibliographic universe then we should consider both, item and physical object, as two separate entities that collaborate with each other. What lightens the mechanism of inheritance is delegation: an object of class *Item* contains a collection of physical objects, that is, objects of class *Physical Object*, or, in entity-relationship terms, the *Item* entity has a one-to-many relationship with the entity *Physical Object*. The behavior of the item as a physical object is delegated to its physical objects.

In addition, this would extend the use of *Physical Object* entities to non-bibliographic physical objects that may be related in some way with the bibliographic system.[19] But without going so far, this method allows the reification of many entities of the bibliographic universe without falling into an explosion of classes. A Book as a physical object (i.e., a volume), can "know" its number of pages, a CD object its duration, and so on. Any of them, as any *Physical Object*, can answer its dimensions (Figure 4). These examples show where the *Manifestation's* attributes *dimensions of the carrier* and *extent of carrier* are located in the object model. To show how other attributes can be depicted, I have added two classes: *Material* and *Merchandise* for attributes *physical medium* and *terms of availability.* Any question of a *manifestation* that has to do with the lifecycle of the material ("would you still show your content in ten years?") would be redirected to the *item*, the *item* would redirect it to its *physical object*, and finally the *physical object* would make the question to its *material*, the object that finally returns the answer. A similar chain could be established when dealing with thes *manifestation* considering it as merchandise. Similar abstraction processes can be made with the other attributes.

In this scheme we have lost the distinction between *Item* and *Manifestation Singleton*. And yet we cannot overcome the objection concerning archive documents. If we consider that:

- a document may be a copy of a manifestation that embodies an expression that realizes a work (an *Item* entity as in FRBR$_{ER}$)
- a document may be a manifestation of an expression produced only once (in the sense of FRBR$_{OO}$ *F4 Manifestation Singleton*)
- a document is a physical object that may or may not participate in the WEMI chain (archival documents, legal documents, historical or archaeological documents, etc.)

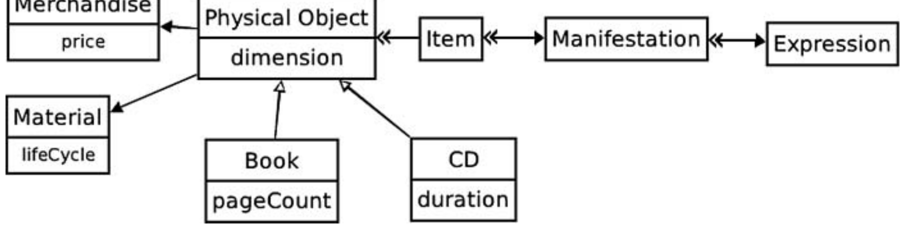

FIGURE 4 Physical Object as an Abstract Class and Some Concrete Subclasses.

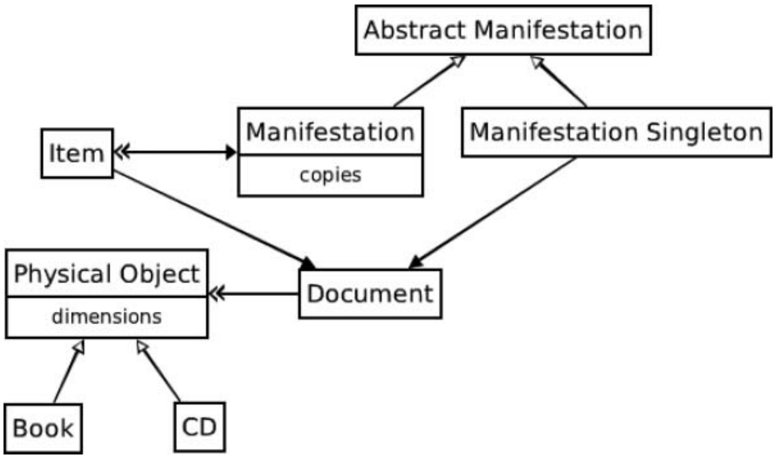

FIGURE 5 Document, Item, and Carrier.

then it is clear that in the above scheme the refication of the document is needed, considering it as a larger entity than the FRBR *Item*, which may or may not match it. And given that any scheme that involves inheritance shall sacrifice some aspect, the only solution is delegation. But it is not just a technical-implementative solution; the analysis led us to the clarification of which features these entities share and which ones distinguish them: what in a first naive approach appeared as identity (an item is a document, a document is a physical object) now expresses conceptual differences that arose after applying object design techniques. The *Item* retains its particularity of being a single copy of a manifestation, but delegates the behavior related to its documentary aspect to a *Document* object. The *Manifestation Singleton*, as an *Abstract Manifestation* retains its characteristic of being the embodiment of an expression, but embodied only once, and its behavior as document is delegated to a *Document* object. A document can consist of one or more physical objects, so all the behavior of the *Document* as "thing" is delegated to objects of *Physical Object* class (Figure 5). The scheme allows us to define specific classes for particular physical objects, such as *Book* or *CD* without duplicating code and without an explosion of classes.

WHERE IS THE CONTENT?

I have covered some aspects concerning the carrier, but where is the content in this scheme? If we have in mind the definition of *Expression* given by FRBR, "The intellectual or artistic realization of a work in the form of alpha-numeric, musical, or choreographic notation, sound, image, object,

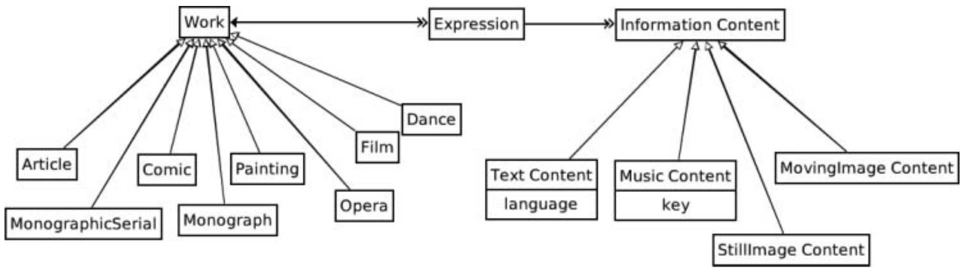

FIGURE 6 Work Subclasses and Content Types as Classes.

movement, etc., or any combination of such forms," we must conclude that the content is inside this entity. Following object technology principles, we have to discard the idea of the content as an attribute of the expression and prefer a reification of the concept "content." Having a class *Content* would let any expression delegate its behavior related with content to instances of this class. Figure 6 shows in a reduced and simplified way some of the classes resulting from this conclusion. Instead of "types of Work," or "types of Expression," we have "types of Content," following Yee's categories of pure content[20,21] and Delsey's categories of work.[22] The scheme shows how some attributes of the *Work*, as *language* and *key* are now attributes of specific classes of content: text content is bound to be in a certain language, therefore it is the *Text Content* object that "knows" the language of the work and expression. The "mixed works" issue can be solved considering the relationship between expression and content as a one-to-many relationship. Put in terms of object technology, an *Expression* object can collaborate with more than one *Content* object. This allows all combinations, but does not force them to be established a priori. An *Opera*, for example, is a work that is realized by an expression that includes two distinct categories of content. All behavior associated with musical works is not in the work nor in the expression, but in the content; pure musical works and any artistic combination that includes music will delegate everything related to its musical aspect to the corresponding *Content* object, independently of the class hierarchy in which they are. And everything in the system that has to do with music content will be said only once, a must of all programming paradigms.

Figure 6 also shows that concrete examples of *Works*, as *Opera, Song, Article*, *Monographic Serial*, and so on are considered classes of their own. That way we can isolate the behavior of each type of work, which varies much from each other, while keeping the names established by usage. There is no firmly established taxonomy to locate traditional bibliographic entities such as Monograph, Article, Instrumental piece, or Opera, as has been repeatedly said[23,24] Perhaps more research is needed on the documentary types

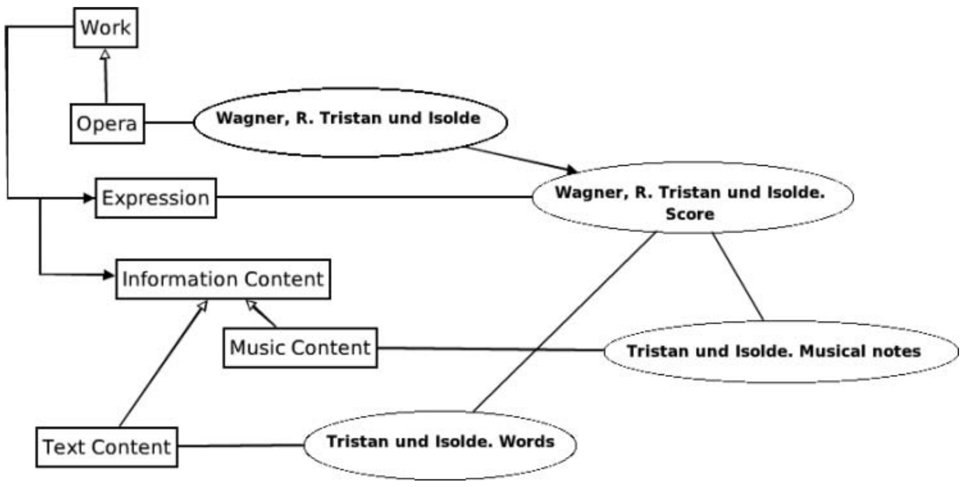
Work
Opera
Wagner, R. Tristan und Isolde
Expression
Wagner, R. Tristan und Isolde. Score
Information Content
Music Content
Tristan und Isolde. Musical notes
Text Content
Tristan und Isolde. Words

FIGURE 7 Example of Several Content Types in One Work.

issue, a topic where object orientation, which seeks to define interaction ways instead of data, could be a great help.

Figure 7 provides examples of specific works, that is, instances of existing classes. Expressed in FRBR notation, this scheme would be

- w1 Wagner, R. Tristan und Isolde
- - e1 Wagner, R. Tristan und Isolde. Score
- - - c1 Tristan und Isolde. Sound
- - - c2 Tristan und Isolde. Words

in which the letter *c* refers to the *Content* entity.

ARCHIVE DOCUMENTS

While in the scheme shown in Figure 5 documents that do not participate in the WEMI chain may be included, it is precisely that chain that allows access to the document content through the *Content* object, as we have just seen, since this object is located in the *Expression* class. It would be expected for an object that represents an archive document to access its content, whether it holds referential information or the digital full content. Since we have not used inheritance to express this relationship, the solution is readily available: simply through the definition of a subclass of *Document* that holds a *Content* object. While a message for getting the content in a Document object is delegated to the Expression object, which in turn delegates it to its *Content* object, objects of this new class, say, *Archive Document*, would

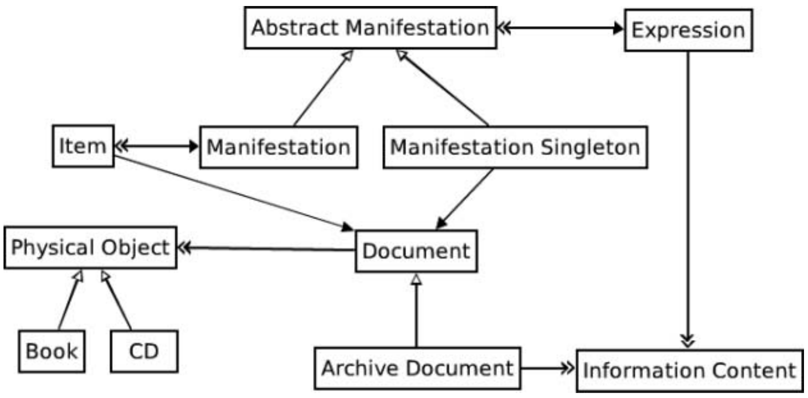

FIGURE 8 The Class Archive Document and Its Relationship with Class Information Content.

delegate the dispatching of any message related with the content to its own *Content* object[25] (Figure 8).

In other words, the reification of the concept of information content makes it possible for any object in a system that has to deal with any kind of information content to access it using delegation.

The scheme, which has been simplified for the purposes of this exposition, shows similarity to that given by Delsey to conceptualize the *Anglo-American Cataloguing Rules*, Second Edition (AACR2),[26] now framed in the outline of FRBR:

> [An] *Item* is a document or set of documents... [which] is an object that comprises intellectual and/or artistic content and is conceived, produced, and/or issued as an entity [and] may contain *Document parts* [which] is a physically separate component of a document. *Document* and *Document part* consist of *Content* [which] may (or may not) "contain" one or more *Content Part* [which] is an individual component of the intellectual or artistic content of a *Document* or *document part*. *Content* and *Content Part* are set as one or more *Infixion* [which] is the formatting of intellectual or artistic content [and] is stored on one or more *Physical Carrier* [which] is a physical medium in which data, sound, images, etc. are stored.

An aspect directly related to the *Item* entity has been omitted to simplify the exposition: the relationship between the *Manifestation* entity and the ideal physical object it represents. A manifestation, in the sense of *F3 Manifestation Product Type*, does not represent a particular physical object, but a kind of ideal physical object, a template of which each copy is presumably an identical exemplar.[27] This feature has also been noted by Renear and Choi in their analysis of whether the manifestation is concrete or abstract.[28] Figure 9 depicts the classes needed for addressing this issue. A *Physical*

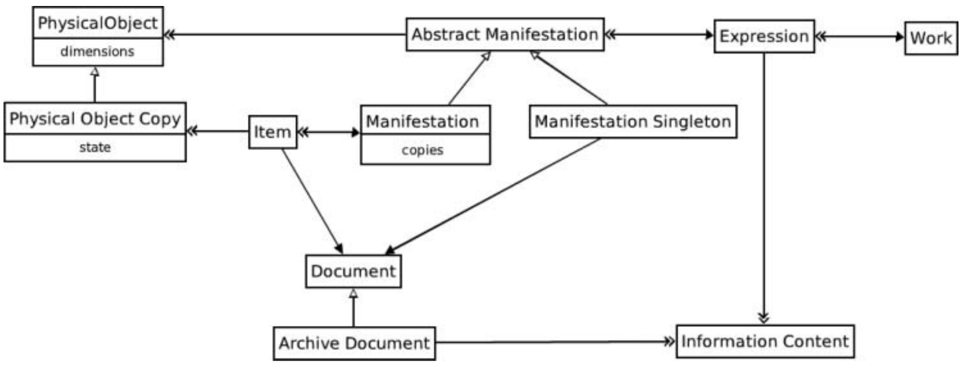

FIGURE 9 The Complete Class Scheme.

Object Copy is a subclass of *Physical Object* that has the behavior concerning the physical object as a particular entity (e.g., the particular state of the object). Everything that has to do with the physical object as a general entity is located in the superclass (e.g., the dimensions of the object).

Figure 10 completes the scheme by showing some concrete subclasses for physical objects, works, and content types. The idea behind these classes, which I cannot cover here in detail, is to keep the terminology established by usage; thus concrete subclasses (the leaves of the tree in the figure) represents things and concepts that used to be called by the same names used the system (the name of the classes): *Book, CD, Text, Music, Opera, Novel, Article,* and so on.

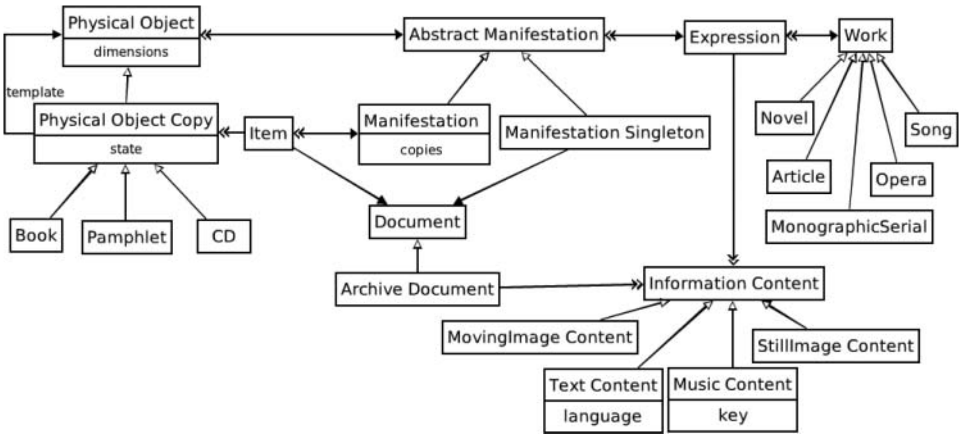

FIGURE 10 The Complete Scheme Showing Some Concrete Examples.

CONCLUSION

I believe that the scheme presented here is sufficient to express the benefits of object orientation and design patterns for the analysis of the FRBR model analysis not only because it can provide inspiration for application developers but because it can lead to conceptual rethinking that could contribute to increase our understanding of some theoretical aspects.

I have also tried to incorporate the perspective of the user, which is equivalent to that of the whole human species, and as such requires an analysis of all the different cultural patterns and worldviews that come into play in all practices involving bibliographic entities. I am aware that much research is needed in this field to establish a firm direction, so that any appeal to the "user perspective" is necessarily conjectural and provisional. However, it is essential to resort to that user's perspective, as conceptual models, "can be validated only by agreement of a group of participants who actually need such a model."[29]

This article has shown some of the most important traits of Opus, an object framework programed in the Smalltalk language. It has been the basis for different applications: a migration from traditional databases to object-oriented databases[30,31] and a research on FRBRization of traditional catalogs.[32] While in the first case, objects were persisted in an object-oriented database, in the second the objects were dynamically created in memory. One important idea behind object modeling is that models are not determined by persistence; then a change in the persistence scheme will not impact the model. I am aware that this approach may seem strange to most librarians and to many programmers also. Clearly this topic is outside the scope of this article so I cannot dwell on it.

Perhaps the most problematic issue is the sustainability of the models and the implementation proposals, simply because we do not know what changes in the bibliographic universe will be introduced by the technology of the future. Given this, we can only have confidence that arriving at general abstractions that in turn allow us to locate individual entities in a model homologous to what we observe in the bibliographic reality is a minimum guarantee that bibliographic entities that could appear in the future will find their place.

NOTES

1. *Bibliotecología*, *Biblioteconomía*, *Ciencias de la Documentación*, are some of the names used in Spanish for what, according to English usage, I call here *Library and Information Science* (LIS), which also supports variants in this language such as *Library Science* or *Librarianship*.

2. IFLA Cataloguing Section and IFLA Meetings of Experts on an International Cataloguing Code, "Statement of International Cataloguing Principles," 2009, http://www.ifla.org/publications/statement-of-international-cataloguing-principles

3. Birger Hjorland, "How to define a scientific term such as 'A Work"' (presented at the American Society for Information Science and Technology Annual Meeting, Providence, Rhode Island, 2006), http://www.iva.dk/bh/lifeboat_ko/CONCEPTS/work.htm

4. Tony Gill, "Is that a Reference Model in Your Pocket...? The CIDOC-CRM & IFLA FRBR" (presented at the "Ready to Wear: Metadata Standards to Suit Your Project: An RLG-CIMI Forum, Mountain View, California, 2003), http://www.rlg.org/events/metadata2003/gill.ppt, 28.

5. Erich Gamma, *Design Patterns: Elements of Reusable Object-Oriented Software* (Reading, MA: Addison-Wesley, 1995).

6. E. Truyen et al., "A Generalization and Solution to the Common Ancestor Dilemma Problem in Delegation-Based Object Systems," in *Dynamic Aspects Workshop (DAW04)* (2004), 6.

7. Gamma, *Design Patterns*, 360.

8. *Functional Requirements for Bibliographic Records.* Final Report. Approved by the Standing Committee of the Cataloguing Section on September 1997 as amended and corrected through February 2009. http://www.ifla.org/files/cataloguing/frbr/frbr_2008.pdf, 23.

9. Richard P. Smiraglia, *The Nature of "a Work": Implications for the Organization of Knowledge* (Lanham, MD: Scarecrow Press, 2001), 6.

10. Norberto Manzanos, "El impacto de FRBR en Argentina: Implementación de un modelo de objetos basados en FRBR, CRM y FRBRoo en CAICYT-CONICET." [The Impact of FRBR in Argentina: Implementation of a Model Based on FRBR-ER, CRM and FRBRoo at CAICYT-CONICET], Conference Paper, 2007, http://eprints.rclis.org/handle/10760/11007, 401–403.

11. Martin Heidegger, "El origen de la obra de arte," in *Caminos de bosque*, trans. Helena Cortés and Arturo Leyte (Madrid: Alianza, 1995).

12. Manzanos, "El impacto de FRBR en Argentina," 401–403.

13. Martha Baca and Sherman Clarke, "FRBR and Works of Art, Architecture, and Material Culture," in *Understanding FRBR: What It Is and How It Will Affect Our Retrieval Tools*, ed. Arlene Taylor (Westport, CT: Libraries Unlimited, 2007).

14. Mary Lynette Larsgaard, "FRBR and Cartographic Materials," in *Understanding FRBR: What It Is and How It Will Affect Our Retrieval Tools*, ed. Arlene G. Taylor (Westport, CT: FRBR and Cartographic Materials, 2007).

15. Alexander Thurman, "FRBR and Archival Materials," in *Understanding FRBR: What It Is and How It Will Affect Our Retrieval Tools*, ed. Arlene Taylor (Westport, CT: Libraries Unlimited, 2007).

16. International Working Group on FRBR and CIDOC CRM Harmonisation, "FRBR Object-Oriented Definition and Mapping to FRBRer (Version 1.0.1)," 40.

17. The splitting of AbstractManifestation in two subclasses, Manifestation and ManifestationSingleton depicts the pattern Template.

18. Martha Yee, "Manifestations and Near-Equivalents: Theory, with Special Attention to Moving-Images Materials," *University of California. Postprints* (January 1, 1994), 234–235.

19. For example, if an application aims to represent a library as a kind of virtual world, it should include not only physical objects that carry bibliographic content, but others that do not as well (desks, chairs, etc). The inventory book, if there is one, would be an example of a book (in the sense of physical object) that is not a carrier of a *Work*.

20. Martha Yee, "Musical Works on OCLC, or, What if OCLC Were Actually to Become a Catalog?," *Music Reference Services Quarterly* 31, no. 3/4 (2001): 237–295.

21. Martha Yee, "Lubetzky's Work Principle," in *The Future of Cataloging: Insights from the Lubetzky Symposium*, eds. Robert L Maxwell and Tschera Harkness Connell (Chicago: ALA Editions, 2000), 85.

22. Tom Delsey, "Back to the Future," *ALA Preconference: RDA, FRBR, and FRAD: Making the Connection* (2009), 5.

23. John Helmer, "Cataloging, Economics, and the Experience of Works" (University of California, Los Angeles, 1987), http://www.uoregon.edu/~jhelmer/Helmer%20JF,%20Cataloging,%20Economics%20and%20the%20Experience%20of%20Works.pdf

24. Elaine Svenonius, *The Intellectual Foundation of Information Organization* (Cambridge, MA: MIT Press, 2000), 111–114.

25. In this case the pattern Bridge was applied: Document (Abstraction), Archive Document (Refined Abstraction), Physical Object (Implementor), Book, CD, and so on (Concrete Implementors).

26. Tom Delsey, "The logical structure of the Anglo-American cataloguing rules: drafted for the Joint Steering Committee for revision of AACR" (Ottawa: National Library of Canada, 1998).

27. Norberto Manzanos et al., "Modelo de descripción documental basado en el paradigma de objetos," in *Proceedings VI Congreso del Capítulo Español de ISKO*, ed. José Antonio Frías Montoya and Críspulo Travieso Rodríguez (Salamanca, Spain: 2003), http://eprints.rclis.org/handle/10760/8875

28. Allen H Renear and Yunseon Choi, "Modeling Our Understanding, Understanding Our Models: The Case of Inheritance in FRBR," Conference Paper, January 1, 2006, http://eprints.rclis.org/archive/00008158/, 5.

29. Patrick Le Boeuf, "What is a Conceptual Model" (presented at the "De la conception à la survie: comment documenter et conserver les productions du spectacle multimédia?," Paris, 2003), 5.

30. Manzanos, "El impacto de FRBR en Argentina."

31. Manzanos, N. and A. M. Flores, "Opus: Sistema de registro bibliográfico basado en FRBR. I Encuentro Nacional de Catalogadores. Biblioteca Nacional 2008." Presented at the I Encuentro Nacional de Catalogadores. Biblioteca Nacional 2008, Buenos Aires. (n.d.).

32. Norberto Manzanos, "Ferberización de una base de datos bibliográfica" (presented at the I Jornada de Intercambio y Reflexión acerca de la Investigación en Bibliotecología, La Plata: Facultad de Humanidades y Ciencias de la Educación de la Universidad Nacional de La Plata, 2010), http://jornadabibliotecologia.fahce.unlp.edu.ar/jornada-2010/manzanos