International Review of Research in Open and Distributed Learning

Learning Objects: Resources For Distance Education Worldwide

Stephen Downes

Volume 2, numéro 1, juillet 2001

URI : https://id.erudit.org/iderudit/1073120ar
DOI : https://doi.org/10.19173/irrodl.v2i1.32

Aller au sommaire du numéro

Éditeur(s)

Athabasca University Press (AU Press)

ISSN

1492-3831 (numérique)

Découvrir la revue

Citer cet article

Downes, S. (2001). Learning Objects: Resources For Distance Education Worldwide. *International Review of Research in Open and Distributed Learning*, *2*(1), 1–35. https://doi.org/10.19173/irrodl.v2i1.32

Résumé de l'article

This article discusses the topic of learning objects in three parts. First, it identifies a need for learning objects and describes their essential components based on this need. Second, drawing on concepts from recent developments in computer science, it describes learning objects from a theoretical perspective. Finally, it describes learning objects in practice, first as they are created or generated by content authors, and second, as they are displayed or used by students and other client groups.

International Review of Research in Open and Distance Learning ©
ISSN: 1492-3831
Vol. 2, No. 1 (2001)

Learning Objects: Resources For Distance Education Worldwide

Stephen Downes

Abstract

This article discusses the topic of learning objects in three parts. First, it identifies a need for learning objects and describes their essential components based on this need. Second, drawing on concepts from recent developments in computer science, it describes learning objects from a theoretical perspective. Finally, it describes learning objects in practice, first as they are created or generated by content authors, and second, as they are displayed or used by students and other client groups.

The Need for and Nature of Learning Objects

Some Assumptions and a Premise

Before launching directly into a discussion of learning objects, it is important to examine some assumptions and a premise. The first assumption is that there are thousands of colleges and universities, each of which teaches, for example, a course in introductory trigonometry. Each such trigonometry course in each of these institutions describes, for example, the sine wave function. Moreover, because the properties of sine wave functions remains constant from institution to institution, we can assume that each institution's description of sine wave functions is more or less the same as other institutions'. What we have, then, are thousands of similar descriptions of sine wave functions. Now suppose that each of these institutions decided to put its "Introductory Trigonometry" course online. This is no stretch; the International Data Corporation estimates that 84% of four-year colleges will offer courses online by 2002 (Council for Higher Education Accreditation, 1999). The result will be thousands of similar descriptions of sine wave functions available online.

Now for the premise: the world does not need thousands of similar descriptions of sine wave functions available online. Rather, what the world needs is one, or maybe a dozen at most, descriptions of sine wave functions available online. The reasons are manifest. If some educational content, such as a description of sine wave functions, is available online, then it is available worldwide. Even if only one such piece of educational content were created, it could be accessed by

each of the thousands of educational institutions teaching the same material. Moreover, educational content is not inexpensive to produce. Even a plain web page, authored by a mathematics professor, can cost hundreds of dollars. Include graphics and a little animation and the price is double. Add an interactive exercise and the price is quadrupled.

Suppose that just one description of the sine wave function is produced. A high quality and fully interactive piece of learning material could be produced for, perhaps, $1,000. If 1,000 institutions share this one item, the cost is $1 per institution. But if each of a thousand institutions produces a similar item, then each institution must pay $1,000, with a resulting total expenditure of $1,000,000. For one lesson. In one course.

The economics are relentless. It makes no financial sense to spend millions of dollars producing multiple versions of similar learning objects when single versions of the same objects could be shared at a much lower cost per institution. There will be sharing, because no institution producing its own materials on its own could compete with institutions sharing learning materials.

Courses? No, Not Courses

If we accept the premise that institutions will share learning materials, then we need to ask, what will they share? The answer that intuitively offers itself is – courses.

Many sources of online learning materials, for example Telecampus [http://courses.telecampus.edu/], or the Web of Asynchronous Learning Networks [http://www.aln.org/], list only courses. These are good listings; they are divided into subject areas, with each subject page containing a list of similar courses offered by different institutions. These lists of courses are directed at potential consumers of learning material, that is, students. Students are typically motivated by an interest in a topic (Downes, 2000) and select courses from the list of offerings in that topic. Moreover, students are typically *offered* learning materials in course-sized units, and attempt to complete degree of diploma programs defined as sets of related courses.

Why, then, would institutions not share these courses? To a certain degree, they already do so. Most colleges and universities define course articulation policies, whereby a course completed at one institution is accepted for credit at another institution. A good example is the Baccalaureate Core Course Equivalency defined by Oregon State University for courses at thirteen regional community colleges [http://www.orst.edu/Dept/admindb/arttable/scr1140_arttab.htm].

Course articulations are the result of complex negotiations between teams of

academics. Consider, for example, the information contained in the Illinois Mathematics and Computer Science Articulation Guide [http://www.imacc.org/articulation/]. To count as equivalent credit for, say, a trigonometry course, a candidate course must require certain prerequisites and contain material covering a certain set of topics.

Course sharing between institutions is difficult to define and maintain. One reason is the regional nature of course articulations – it is notable that Oregon State University has made no attempt to articulate courses offered by, say, community colleges in Florida. The detailed topic-by-topic definition or articulation agreements are also a challenge. It is unlikely that *any* course could be shared by any significant number of institutions in different states or different nations.

This disparity is also reflected in online course listings. Returning to the Telecampus guide we find twenty separate history courses listed [http://courses.telecampus.edu/subjects.cfm?category=12]. No two courses share the same name. And though a number of courses focus on the same region and time period, no two courses share the same contents. This is true to a degree more or less across all subjects and across all institutions. Although courses may share elements in common, it is rare to find two courses from two institutions that share the same, and only the same, set of elements.

Thus, courses themselves are *not* suitable candidates for sharing. Yet the dominant form of online educational today is the course. So it should come as no surprise that there is very little sharing of educational resources, even online resources, despite the tremendous cost savings. Despite what the world needs, what the world is getting is 1,000 different versions of "Introductory Trigonometry". It makes no sense, and the current system will have to change.

Sharing the Old Way

Whether at the K-12 or college level, today's classroom is already an example of extensive resource sharing. While it is likely that neither the producers nor the consumers of these resources would describe the transactions as "sharing", if we define *sharing* as one centrally produced resource used by many, then these classes are sharing resources. The clearest example of resource sharing the old way in today's classrooms occurs through the use of textbooks. These resources bear all the hallmarks of sharing: they are centrally produced and obtained as needed by classroom instructors around the world. In many cases, the information in textbooks is so commonly used the work becomes standard.

However, textbooks are just one type of item among many that are shared by classes around the world. No K-12 school is complete without a set of wall maps in geography classes, periodic tables of the elements in science classes, and sets of large block letters for the early years. A rich and useful set of

classroom displays is distributed by organizations as varied as astronomical societies [http://www.aspsky.org/catalog/class.html], museums, and publishing companies. In the area of multimedia, teachers employ a wide variety of centrally produced materials including filmstrips and videos, CD-ROMs and other software, presentation graphics and even complete learning resources, such as are produced by Plato [http://www.plato.com].

It is important to review the old ways of sharing resources not only to show that resource sharing is an established fact in today's classrooms, but also to point to some of the elements of resource sharing already in place. For it is reasonable to expect that many of the elements of resource sharing the old way will be replicated in an online environment. For one thing, as mentioned, various publishers and content producers produce resources centrally and distribute them to classes around the world. And while many of these resources are distributed free of charge, the majority of shared resources in classrooms are purchased from their respective producers or intermediaries. Textbook publishing and sales, especially, is a lucrative industry. The National Association of College Stores estimates U.S./Canadian college store sales to be $8.959 billion for the 1998-99 academic year (NACS, 2000).

Second, for the most part, the resources distributed in this manner are not entire classes, but rather, components of classes. This is most clearly the case for classroom aids such as wall maps and posters. But even more comprehensive materials such as textbooks are used only in part, as part of a class. The vast majority of course syllabi require that students obtain more than one textbook. Courses frequently use only parts of textbooks; entire chapters are omitted as being beyond the scope and purpose of the course. Moreover, students frequently use parts of books (or parts of journals) in their research and reading. That's why most university libraries come equipped with photocopiers.

Contemporary Sharing

Though most educational institutions offer only complete courses online, many other agencies have started offering smaller, more portable learning materials. These materials fall short of what we will later define as *learning objects*, but they offer some insight as to the direction and potential of online resources. Immediately we see a division of the territory into, first, the learning resources themselves, and second, lists (or portals) of learning resources. In some few cases (usually where the institution has a wealth of content) these services are combined.

In Canada, the leading learning resources portal is probably Canada's SchoolNet [http://www.schoolnet.ca/]. Follow the link from the home page into "Learning Resources" and select a topic area. A list of resources will be displayed, each

with a short description and a link to an external website. SchoolNet also provides metadata information for each site and provides an "advanced search" using metadata [http://www.schoolnet.ca/home/e/search/advanced_e.asp]. Each resource in the "curriculum" area is approved by a professional "pagemaster" [http://www.schoolnet.ca/home/e/pagemasters/]. For the most part, however, SchoolNet lists and links to institutional home pages, and not to learning resources per se. Teachers using the SchoolNet service must still search through these sites in order to locate suitable materials.

There is a U.S.-based site [http://www.eoe.org/] maintained by the Educational Object Economy Foundation and providing direct links to learning resources themselves. In addition, Merlot [http://www.merlot.org/] currently lists more than 2,000 learning applications that can be accessed via the WWW. These applications are specific materials on specific topics. For example, Merlot lists items such as "Chaucer" [http://icg.fas.Harvard.edu/\char126\relaxchaucer/index.html], "The Great 1906 Earthquake and Fire" [http://www.sfmuseum.org/1906/06.html], and RSPT Expansion: Perturbation Theory [http://www.bgu.ac.il/\char126\relaxsergeev/perturb.htm]. These are examples of materials that have been sorted into category and subcategory, and contributed by educators from around the world.

Educators attempting to use Merlot's resources, though, will still experience frustration. While the topic hierarchy is more detailed than SchoolNet's, and although much more focused resources are listed, educators must still spend quite a bit of time browsing for materials. Moreover, there appears to be no resource metadata and the search mechanism provided on the Merlot site is no better than standard web search engines.

As we can see from this discussion of articulation, there is a need for a mechanism to connect online learning resources with detailed course objectives. This much more advanced form of resource listing forms the basis for the selection and categorization of resources in MCI WorldCom's MarcoPolo project [http://www.wcom.com/marcopolo] (see also Downes, 2000, July/August). MarcoPolo is a compilation of teaching resources from six educational institutions that provide free Internet content for K-12 education. What the six partners have in common, and what makes this an important and interesting development in online learning, is an adherence to national curriculum and evaluation standards in the subject areas. Material is categorized by grade level, and individual items are matched to individual learning topics. Despite its strengths, however, MarcoPolo is a closed project; only the six member institutions contribute content. There is no centralized search facility and no metadata listings for the resources. The only curricula supported are United States school curricula, so the resource is not useful in a global marketplace.

Other resources are available, but these three sites typify the contemporary art of shared learning resources. There is much to be done to make these resources widely useful. Much better systems of categorization and searching, and more

robust mechanisms for updating and submissions are required. Learning resources need to be tied more closely to learning objectives, but in such a way as not to be tied to a specific curriculum. An even greater weakness appears when we look at the collective set of learning resources (or applications, as Merlot calls them) offered by these three sites. It is nearly impossible to identify consistency in format, scope, methodology, educational level or presentations. Some resources include lesson plans, but many others do not. Some are authored in Java, others in HTML, and others in a hybrid mixture known only to the author. Some involve 10 minutes of student time, others would occupy an entire day. And there is no structured means for an instructor to know which is which.

Creating Content and The Cost of Online Learning

It may be argued that university courses are fundamentally different from K-12 courses. While there is a great deal of commonality between grade 1 English from school to school, university courses are individual entities in their own right. Each time a course is offered by a university professor, it is created anew, adding a new interpretation or a new reading of familiar material. While this argument may be true enough, it is important to look at the cost of creating courses, and especially online courses, in this manner. Creating an online course from scratch is a long, labour intensive process. Costs can vary from $4,000 (all figures in Canadian dollars) to $100,000.

To cite a typical example, Bates (2000) estimates that a course consumes 30 days of a subject expert's time, plus an additional seven days for an Internet specialist, plus additional expenses for copyright review, academic approval, and administration. A budget for course development, adapted from Bates' Distance Education and Technology (DET) unit (p. 138), is presented in Table 1.

Bates' estimate is conservative. He assumes an experienced course author and HTML specialist. He does not include any instructional design costs. Course design is straightforward and does not involve the development of any interactive media or course-specific Java programming. All of these would add significantly to the $24,000 total cost.

Delivery costs on Bates' model amount to an additional $13,161, as depicted in Table 2 (adapted from Bates, p. 138).

Almost all online course developers use the design model Bates describes. It involves a course being developed from scratch, using nothing more than a traditional university course or a good textbook as a guide. The course author typically authors all the content, including examples and demonstrations,

Table 1: Sample Course Development Budget

Subject Experts	30 days @ $400 / day	12,000
Internet Specialist	7 days @ $300 / day	2,100
Graphics and Interface Design	4 days @ $300 / day	1,200
Copyright Clearance		700
Total Direct DET Costs		16,000
DET overheads	25% of 16,000	4,000
Faculty of Education Approval		4,000
TOTAL		24,000

Table 2: Sample Course Delivery Budget

Library		1,000
Server costs		300
Tutoring	40 students @ $220	8,800
Registration	$14 x 29	406
Administration	$28.86 x 40	1,155
Printed materials and postage		1,500
TOTAL		13,161

quizzes and tests. Because of the cost of development, there is little use of course specific software or multimedia. The course is then offered to a small number of students over a limited time, resulting in course fees that are comparable, if not greater than, traditional university course fees.

We can do so much better than this. We need to design online courses – even university courses – in such a way as to reduce these costs without diminishing the value of a university education. We need to do this by extracting what these courses have *in common* and by making these common elements available online.

Let me begin by presenting some examples. Consider the Teacher's Guide to the Holocaust [http://fcit.coedu.usf.edu/holocaust/]. This site consists

of dozens of resources on the Holocaust that may be used and reused by any teacher approaching the subject. Each of the class activities (see [http://fcit.coedu.usf.edu/holocaust/activity/activity.htm] for links to lesson plans and activities) could be treated as an individual learning object. The Holocaust is a very large subject – much larger than sine waves – and is appropriately divided into many components. But it is far easier, and results in far greater quality, to assemble a lesson or series of lessons from these materials, than to create something from scratch.

Or consider "Hamlet." There is not, of course, one single description of Hamlet. However, there is only one text of the play "Hamlet" and it is not a stretch to envision a definitive online multimedia edition. Such an edition would not only contain the text, it would also contain video clips, audio clips, commentary from selected sources, pop-up glossaries, and more. I have actually seen a CD-ROM version of "Hamlet" presented this way; all that is needed is online distribution. It is not a stretch to imagine a multimedia company spending $1,000,000 on such a production. Assume that "Hamlet" is taught in 10,000 schools, colleges or universities around the world (hardly a liberal estimate). Assume 20 students per class (an underestimate, to be sure!). At $5 per student, the company would make the $1,000,000 investment back in 1 year! The economics are very good, and this excellent resource would be cheaper than even the book alone. A course specializing in "Hamlet" would employ the digital "Hamlet" as a central resource, and incorporate as well essays, discussions and articles from scholars around the world. There is no reason why an academic journal cannot contribute a learning object, in the form of an article, or even a set of articles.

A description of the sine wave, or an account of the Holocaust, or a reading of "Hamlet" – these become "a piece of learning material" when they are able to meet a learning objective. Of course by *description of a sine wave* we refer to more than merely a page or two of text plus an illustration. That is not what happens in the classroom; students are given a variety of examples, asked to calculate their own examples, are tested on their understanding, and so forth. A better phrasing, perhaps, is . *lesson on sine wave functions*.

Learning Objects from a Theoretical Perspective

Course Construction and Rapid Application Design (RAD)

Courses developed along the Bates model are expensive because of two major (and related) design features. First, all course material is created from scratch, and second, this material is applied only to the limited number of students taking this particular course. In order to lower costs, therefore, a course development program must enable educators to avoid creating everything from scratch, and to allow created course content to be applied to a much larger number of students.

From a certain perspective, an online course is nothing more than just another application, and software engineers have long since learned that it is inefficient to design applications from scratch. Educators need to apply design techniques learned long ago by the software industry, and in particular, they need to learn a concept called Rapid Application Design (RAD).

Rapid Application Design is a process which allows software engineers to develop products more quickly and of higher quality. RAD involves several components, including a greater emphasis on client consulting, prototyping, and more informal communications (see [http://sysdev.ucdavis.edu/WEBADM/document/rad_toc.htm] for more information). But of interest here is the engineers' re-use of software components within the context of a CASE (computer-aided software engineering) environment. The application of RAD for software development allows a designer to select and apply a set of predefined subroutines from a menu or selection within a programming environment. A good example of this sort of environment is Microsoft's Visual Basic [http://msdn.microsoft.com/vbasic/], a programming environment that lets an engineer design a page or flow of logic by dragging program elements from a toolbox.

Similar methodologies exist for a wide variety of creative or constructive tasks. A professional chef, for example, will carefully design a kitchen environment so that when he is called upon to create crepes suzette, the essential ingredients – including pre-mixed recipe ingredients – are readily available. Auto mechanics also work in a dedicated environment and have at hand every tool and component they may need to fix anything from a Lada to a Lamborghini.

Online course developers, pressed for time and unable to sustain $24,000 development costs, will begin to employ similar methodologies. An online course, viewed as a piece of software, may be seen as a collection of reusable subroutines and applications. An online course, viewed as a collection of learning objectives, may be seen as a collection of reusable learning materials. The heart – and essence – of a learning object economy is the merging of these two concepts, of viewing reusable learning materials as reusable subroutines and applications. Educators in the corporate and software communities have known about this concept for some time. As Wayne Wieseler, an author working with Cisco Systems, writes, "reusable content in the form of objects stored in a database has become the Holy Grail in the e-learning and knowledge management communities" (Wieseler, 1999, p. 4).

Object-Oriented Design

To delve more deeply into the construction and organization of learning objects, it is necessary to introduce another concept from computer programming, object-oriented design (e.g., Montlick, 1999). The idea behind object-oriented design is that prototypical entities, once defined, are then cloned and used by a

piece of software as needed. Suppose, for example, as a programmer you needed to store information about "students." You would first design a prototypical student and define for it properties common to all students. Many aspects of the prototypical student would be undefined, however, such as the student's name, age, or telephone number. These unknowns would be given placeholder values (or 'defaults') until they are defined.

When a program needs to work with a student, it refers to the prototype and *clones* a copy of the prototype in the computer's memory (it is actually called cloning in computer science – in perl the prototype is cloned and *blessed* to reserve its place in memory). The newly cloned prototype is given a name, and then values or attributes are assigned to it. For example:

```
Clone_object: type=student id=New_student

New_student -> name = ‘‘Fred Smith’’

New_student -> age = ‘‘32’’

New_student -> phone = ‘‘555-1212’’
```

Where object-oriented design gets interesting – and useful – is in the methodology used to *construct* object prototypes. For clearly, an entity like Fred is a complex entity. Fred is an animal, so he has animal properties, such as age, height or weight. Fred is human, so he has human properties, such as a birthday, eye colour, and hair colour. Fred is a Canadian, so he has properties common to all Canadians, such as a social insurance number and a postal code (were Fred American, he would have a social security number and a zip code). Fred is a student, so he has student-specific properties, such as a student number or a list of classes. Were Fred an instructor, he would have instructor-specific properties as well, such as a parking spot.

When we define a student prototype for the first time, it makes no sense to define for this prototype alone each and all of these properties. This would mean that we must define similar sets of properties for all the people involved in an educational setting: students, instructors, cafeteria workers and groundskeepers. Rather, what happens in object-oriented design is that the most basic prototype is constructed first – in this case, a generic *animal* prototype. Then, the next more detailed prototype, a *human* is defined. The human prototype "inherits" the animal prototype; that is, we say that all the properties an animal can have, a human can have as well. Thus, when we create the human prototype, we need only create those properties and behaviours that are unique to humans.

And so this defining continues on up the hierarchy. When we create a student prototype, we define a student as inheriting all the properties of a human, or

all those properties of a Canadian, and define only those properties that are unique to students. Thus programmers can quickly and efficiently create a new type of entity – a special class of students, for example, or a new nationality – by inheriting the necessary properties from more generic entities.

Object prototypes also define prototypical actions or behaviours for their clones. For example, a behaviour we might expect from a student is to register for a course. The student prototype has this behaviour predefined as a function; when a clone is created, it comes complete with this behaviour. Hence, we can make our clone do things by referring to these predefined functions (or, in computer terminology, *methods*). To have Fred Smith register in a course, for example, we would execute a command that looks something like this:

```
New_student -> register_in_a_course(course_id = ‘‘3212’’)
```

The course into which Fred is registering is itself another object. In our management system, a course prototype has been defined, and at some point, a specific course has been created using that prototype. When the function "register_in_a_course" is executed in Fred, the Fred-object communicates with the course-object and executes a related function in the course object, "add_student_to_course."

Objects may interact – or more generally, be *related* to each other, in many ways. The most useful and common form of interaction is the *containing* interaction. Just as Fred may contain various other objects (such as a heart or a liver, most obviously, but also $4.95 in change, a 6 inch ruler and a pager), one object may in general contain one or more other objects. A course may contain students, for example. Or a course may contain units or modules. A unit may contain a test. Each of these items is an object, defined from a prototype, which may interact with other objects in predefined ways. In a course which contains both a unit test and a grade book, for example, the unit test could interact with the grade book. What would happen is that Fred (the 'student' object) would interact with the test (the 'test' object'), which in turn would interact with the grade book (a 'grade book' object).

Open Standards

A third major concept drawn from the world of computing science – and especially from the recent emergence of Internet technologies – is the use of *open standards* in course construction. An open standard is similar to a language understood and used by everyone. Just as, for example, the meanings of such terms as *Paris*, *the capital of France*, and *European* are understood by almost all speakers of English, so also in an open standard are the meanings of terms and definitions widely understood and shared. The open standard with which most online educators are familiar is Hypertext Markup Language, or HTML.

This language is a shared vocabulary for all people wishing to read or write Internet documents. The term ¡h1¿ is commonly understood as a header tag; the term ¡I¿ denotes italics.

Open standards may be contrasted with *proprietary*, or *closed* standards. Consider a document written in an older version of MS Word, for example. This word processing program used a special set of notation to define italics, bold face, and a wide variety of other features. Because other software manufacturers did not know these standards, only people using MS Word could read a document written in MS Word.[1]

The purpose of open standards is to allow engineers from various software or hardware companies develop devices and programs that operate in harmony. A document saved in an open standard could be read, printed or transmitted by any number of programs and devices.

The IMS protocols and SCORM

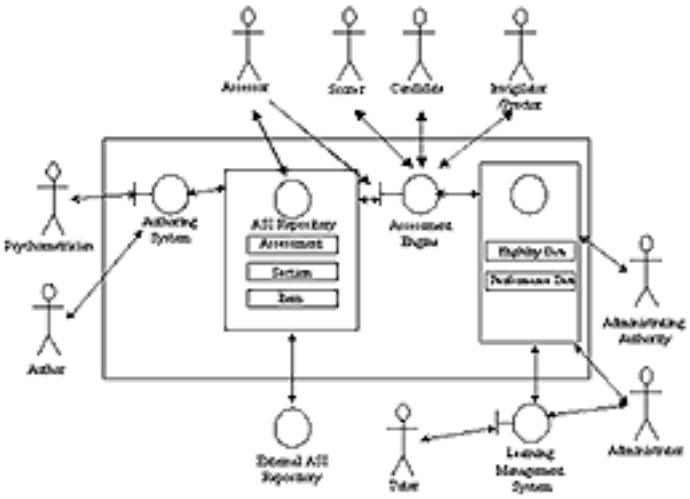
The IMS (Instructional Management Systems) Project is a consortium of educational institutions, software companies and publishers. The Project's objective is to "promote the widespread adoption of specifications that will allow distributed learning environments and content from multiple authors to work together (in technical parlance, "interoperate")" (IMS Global Learning Consortium, 2000). By *distributed learning environments and content*, the authors mean different sets of learning materials, authored in different programming languages using different programs and located on different computers around the world.

This is an elusive goal. It amounts to enabling content produced using Blackboard [http://www.blackboard.com/] and stored on a computer in Istanbul (an interactive atlas, perhaps) to be used in a course authored in WebCT [http://www.webct.com/] and located in Long Island, New York. The term *used* means, in this context, that the two elements – the atlas and the course – could interact with each other. The atlas, for example, might report to the course how long a give student spent studying cloud formations, and the course might instruct the atlas to display the appropriate university logo and links to discussion boards.

In order for this to work, the atlas in Turkey and the course in the United States must define similar objects in a similar manner. For example, both programs must understand what is meant by 'course', or 'institution', or even 'logo'. Thus there is a need to obtain a common definition of the objects and properties used by the two separate systems. Thus, the core of the IMS specification involves the definition of prototype objects, or more accurately, descriptions of prototype objects, since they would be defined differently using different computer

languages. The IMS Enterprise Information Model (IMS Global Learning Consortium, 1999) for example, defines a 'Person Data Object', a 'Group Data Object', and a 'Membership Data Object'. In a similar manner, objects must interact with each other in predefined ways. If one program is expecting a grade as a digit and calls it 'grade', and the other sends it as a word and calls it 'score', then the two programs are unable to interact. A document like the Question & Test Interoperability Information Model Specification (IMS Global Learning Consortium, 2000c) defines the manner in which various components of a testing system interact with other elements of a wider instructional management system. Figure 1, from IMS Global Learning Consortium (2000c), is illustrative of the interactions being considered.

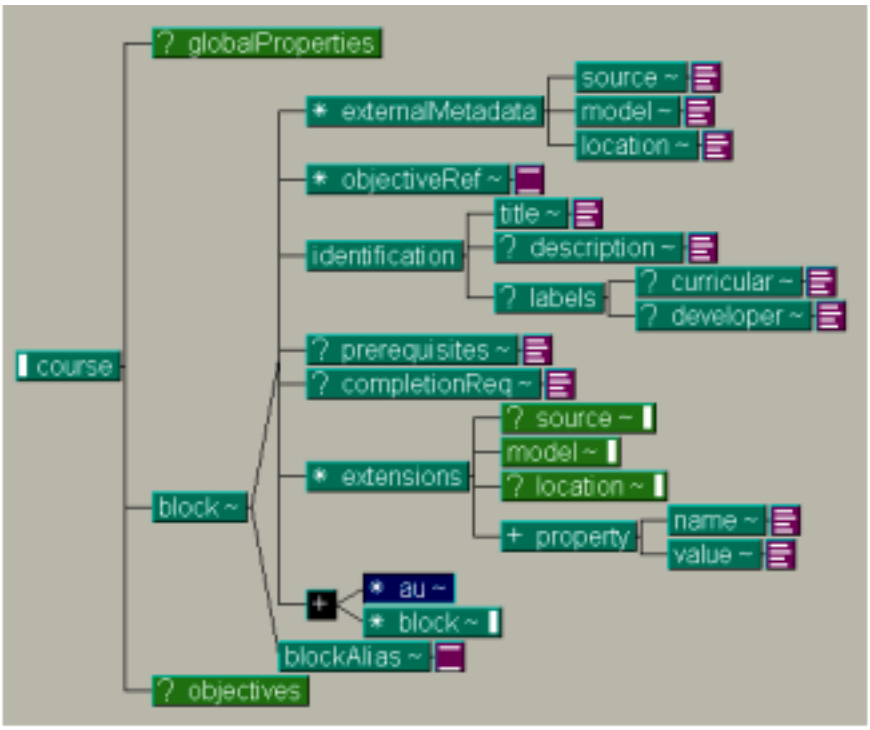
Figure 1: Assessment System Components



This diagram depicts the types of objects which interact. The little stick figures are person-objects, and it is worth noting that no fewer than nine separate types of person-objects are defined. The circles represent *key components*, each of which is an independent piece of software (e.g., the authoring system, or the assessment engine). More detailed implementations of this basic structure are defined by more specific projects. One major project of this type is Advanced Distribute Learning's Sharable Courseware Object Reference Model, SCORM (Advanced Distributed Learning Initiative, 2001). This document describes in detail the object hierarchy in a course and how objects' methods (which are, recall, predefined functions) are defined.

Figure 2 depicts a sample hierarchy from SCORM, displaying only the 'Global Properties' node.

Figure 2: SCORM's Global Properties node



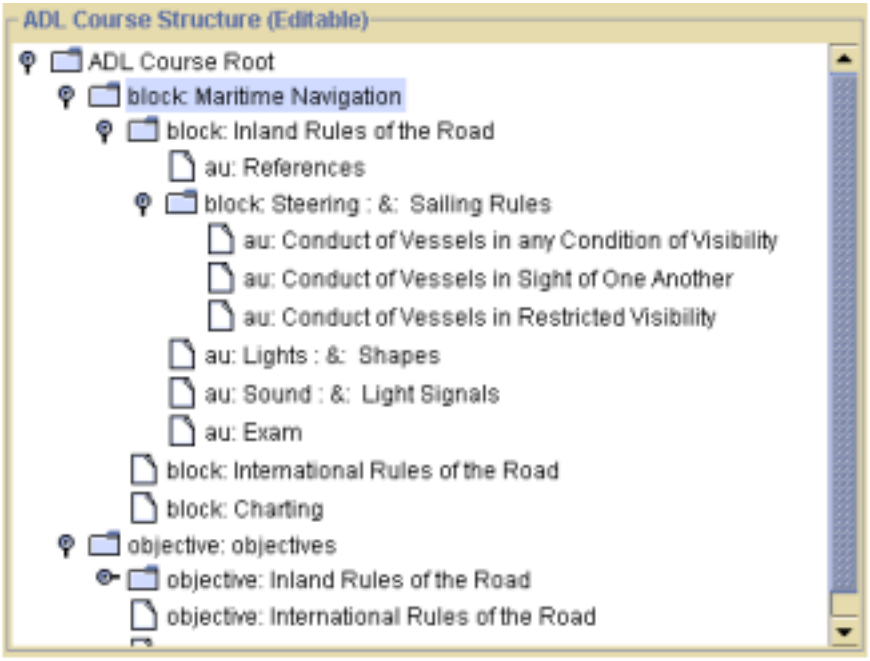
from Advanced Distributed Learning Initiative (2000, p. 34)

The *globalProperties* node contains or references information about the course as a whole, such as prerequisites and course identification. It also provides information describing the general approach used during the design of the course. Drilling more deeply into the course itself, SCORM defines the course components. In an example course structure from SCORM (Advanced Distributed Learning Initiative, 2000) Figure 3 depicts course components and relationships.

(from Advanced Distributed Learning Initiative, 2000, p. 33)

This diagram defines some of the major components of a Maritime Navigation course. Note how the typical course components, such as references, exams, and lesson objectives are each included as distinct components. Each of these elements is an object in its own right; the course as a whole – also an object – contains these discrete objects. As we can see, the IMS Protocols, and specific implementations, such as SCORM, define in detail the potential structure of an online course. In IMS and SCORM, a course – and the elements surrounding a course, such as students, grade books, and prerequisites – are depicted as interacting and inter-related objects.

Figure 3: Sample Course Structure (SCORM)



A Common Language

Thus far we have considered only what may be called the *semantics* of a learning object economy. We have looked at what things (or objects) there are, what they do, what we call them and how we define the meanings of our words. As yet, we have not considered how such a system might be distributed and interoperable. We have not shown just how a computer system in Turkey might send information to a program running in the United States. In order for these two systems to communicate, it is important not only that they be talking about the same things but also that they have a common *language*.

The common language adopted by IMS and SCORM – and being adopted by database programmers, librarians and designers around the world – is the eXtensible Markup Language, or XML, developed by the World Wide Web Consortium (1997a). XML is a means of representing documents according to their internal structure. Thus, for example, a book containing chapters and verses might be represented in XML as follows:

```
<tome name=‘‘Bible’’>
<book name=‘‘Genesis’’>
<chapter name=‘‘1’’>
```

```
<verse name=‘‘1’’>
In the beginning God created the heaven and the earth.
</verse>
<verse name=‘‘2’’>
And the earth was without form, and void; and darkness was upon the face of the deep.
 And the Spirit of God moved upon the face of the waters.
</verse>
...
</chapter>
...
</book>
...
</tome>
(The Bible, n.d.)
In a similar manner, a course containing units, modules and exercises may also be represente
...
<course>
<block id=‘‘B1’’>
 <identification>
<title>Maritime Navigation</title>
 <labels>
<curricular>UNIT</curricular>
 </labels>
</identification>
 <block id=‘‘B2’’>
<identification>
 <title>Inland Rules of the Road</title>
<labels>
 <curricular>MODULE</curricular>
</labels>
 </identification>
<au id=‘‘A1’’>
 <identification>
<title>References</title>
 </identification>
<launch>
 <location>/Courses/Course01/Lesson01/au01.html</location>
</launch>
 </au>
<block id=‘‘B3’’>
 <identification>
<title>Steering \&\#38; Sailing Rules</title>
 <labels>
<curricular>MODULE</curricular>
 </labels>
</identification>
```

. . .

(Advanced Distributed Learning Initiative, 2000, p. 103-104).

XML has two useful features in the context of the current discussion. First, it is structured. An object hierarchy may be defined such that one object may contain other objects, and such that any given object may be assigned any number of properties. Thus, XML is capable of representing an object hierarchy as defined above. All the components of our online course object economy may be described in XML, even non-digital objects such as students, classrooms and books. Second, XML is machine-readable (and machine-writeable, which amounts to the same thing). This means that a computer program can produce a properly formatted XML document using information stored in, for example, a database. It also means that another, different, computer program can read that file and assign the proper values to the proper variables in its own internal representation system.

XML is to structured information what HTML is to structured documents. Each provides a means of distributing content to other systems no matter where they are located and no matter what program they are running. Thus, a piece of learning material, no matter where it is located, may be seamlessly integrated into an online course, provided that the XML tags are employed consistently, that is, provided the semantics are the same.

In an XML document, a schema establishes the semantics of a system of tags.[2] For example, the Dublin Core [http://dublincore.org/] establishes a schema for referring to printed documents. Any XML document which describes a book (and which uses Dublin Core) would use XML tags (and hence, assign corresponding properties to a corresponding book object) defined by the Dublin Core (1999) Metadata Element Set.

Authoring Learning Objects

Authoring Learning Objects – Data

While most guides and references currently discuss online course authoring, the proper reference point is the authoring of learning objects, where a learning object is an element of a course as described above. As we have seen, a learning object may be one of any number of items: a map, a web page, an interactive application, an online video – any element that might be contained inside a course. There are two major facets to authoring learning objects. The first is the *content* of the learning object itself; the second is the *metadata* describing the learning object. We might think of authoring learning objects as akin to

authoring pieces of a puzzle, in which case the content is the image or picture on the surface of the piece, while the metadata is the shape of the piece itself, which allows it to fit snugly with the other pieces.

Today, the most common medium for content is hypertext markup language (HTML). Course authors are able to employ a variety of authoring tools such as Microsoft's FrontPage [http://www.Microsoft.com/frontpage/] or Macromedia's Dreamweaver [http://www.macromedia.com/software/dreamweaver/]. These tools make it possible to create quite sophisticated pages, especially FrontPage, which through a series of extensions allows authors to embed interactive applications into the page. The problem with these HTML pages is that they're not portable, especially not FrontPage generated files, which must interact with a Microsoft server. A web page designed for one course at one university will contain course and university specific information such as the name of the course, the name of the university, and even a colour scheme. To be used or adapted by another course, the pages need to be redesigned. Moreover, HTML pages, especially pages designed using FrontPage, do not display well in multiple formats. A separate version must be created if, perhaps, the page needs to be delivered over wireless access protocol (WAP; see the WAP portal [http://www.wap.com/] for more information), or if it is input as data for analysis by a Javascript or CGI process. HTML, as it is currently implemented by these products, combines content and presentation information, thus narrowly limiting its portability.

In order to be portable, a document's content must be, first, structured, and second, separated from presentation information. This goal is accomplished by XML, which uses tags to structure information and which refers presentation information to a separate document entirely – an XSL file (World Wide Web Consortium, 1997b).

A significant step in the right direction is to create course materials *not* in HTML, but rather, in a structured markup language such as XML. A good example of this is the approach taken by British Columbia's Open Learning Agency, which creates its courses in SGML (Standard Generalized Markup Language), a tagged language very similar to XML (Paille, G., Norman, S., Klassen, P. & Maxwell, J., 1999). By organizing content in this way, print versions, web versions or even wireless versions may all be produced from the same base document in a matter of seconds. Structural elements such as tables of contents and page numbers are generated on the fly, while course or institution specific information is defined in the template. In addition, specialized documents, such as course outlines, may be generated from the same source (e.g., see [http://www.openschool.nc.ca/outlines/] for the Open Learning Agency: Open School Courses and Resources outlines).

SGML documents may be generated and edited using any common SGML editor. One example is Auto-Graphics' Smart Editorial System (SES) [http://www.auto-graphics.com/publishing/SGML.html]. But the implementation – at least as used by OLA – is not portable. The course documents are undif-

ferentiated wholes, so they would have to be adapted by other institutions *as a unit*. In any case, it is not reasonable to employ one language for all parts of an online course. What we are more likely to see, and are beginning to see already, is a set of different languages for different parts. IMS is slowly drafting these specifications and now has four sets: meta-data, enterprise, content packaging, and question-and-test (IMS Global Learning Consortium (2001). Related sets of specifications are being defined by the World Wide Web Consortium, such as Math Mark-Up Language (MML) [http://www.w3.org/Math/] and the Synchronized Multimedia Integration Language (SMIL) [http://www.w3.org/AudioVideo/].

Rather than use a single tool, such as an XML or SGML editor, course authors will begin to use tools designed for specific purposes. Already, we have seen some of these developed, one of the most popular being Half-Baked Software's Hot Potatoes, a tool for designing online quizzes [http://web.uvic.ca/hrd/halfbaked/]. It is not hard to image a suite of standards-compliant applications emerging into the marketplace: one for drafting course outlines, one for creating individual lessons (LessonBuilder [http://www.innovamultimedia.com/lbuilder.htm] for example), another for authoring slide shows, another for creating case studies, and so on. For example, the University of Bristol's TML (Tutorial Markup Language) (Brickley, n.d.a), described a common authoring language for online tutorials and quizzes. The purpose of TML is to "designed to separate the semantic content of a question from its screen layout or formatting" (Brickley) and in so doing, provides a structural framework for tutorial content (the boxes are not part of the document, and are placed there for clarity).

```
-------------------------------------------------------------------
|<!DOCTYPE TML PUBLIC "-//ETS//DTD TML 4.0//EN//" [ ] >           |
|<TML>                                                            |
| <-- Arbitary normal HTML -->                                    |
| |-------------------------------------------------------------| |
| |<TUTORIAL>                                                   | |
| |<QUESTION ATTEMPTS=3 NAME=Capitals TYPE=Multiple-Choice>     | |
| | |-----------------------------------------------------------| |
| | |<p>The text of the question.It consists of HTML text</p>| | |
| | |<CHOICES>                                                | | |
| | | |-----------------------------------------  |            | | |
| | | |<CHOICE CORRECT>This is a correct choice |            | | |
| | | |<CHOICE>This is an incorrect choice      |            | | |
| | | |    .                                    |            | | |
| | | |    .                                    |            | | |
| | | |-----------------------------------------  |            | | |
| | |</CHOICES>                                               | | |
| | |<SCORE>                                                  | | |
| | | |-----------------------------------------  |            | | |
```

```
| | | |<GAIN CORRECT ATTEMPT=1 VALUE=3>          |            | | |
| | | |<GAIN CORRECT ATTEMPT=2 VALUE=1>          |            | | |
| | | |<LOSE HINT VALUE=1>                       |            | | |
| | | |    .                                     |            | | |
| | | |    .                                     |            | | |
| | | |------------------------------------------|            | | |
| | |</SCORE>                                                | | |
| | |<HINTS>                                                 | | |
| | | |------------------------------------------|            | | |
| | | |<HINT>This is a hint                      |            | | |
| | | |<HINT>This is another hint                |            | | |
| | | |    .                                     |            | | |
| | | |    .                                     |            | | |
| | | |------------------------------------------|            | | |
| | |</HINTS>                                                | | |
| | |<RESPONSES>                                             | | |
| | | |------------------------------------------|            | | |
| | | |<WHEN CORRECT><B>That's right!</B>        |            | | |
| | | |<WHEN OPTION=d>You were close that time   |            | | |
| | | |<WHEN INCORRECT>Sorry, that was wrong     |            | | |
| | | |    .                                     |            | | |
| | | |    .                                     |            | | |
| | | |------------------------------------------|            | | |
| | |</RESPONSES>                                            | | |
| | |--------------------------------------------------------- | |
| |</QUESTION>                                               | |
| |<QUESTION ATTEMPTS=3 NAME=Protocols>                      | |
| |    .                                                     | |
| |    .                                                     | |
| |                                                          | |
| |</QUESTION>                                               | |
| |----------------------------------------------------------| |
|</TUTORIAL>                                                   |
|</TML>                                                        |
|--------------------------------------------------------------|
```

(Brickley, n.d.b)

What is interesting about the TML project is that software have been developed both for authoring and for displaying TML documents [http://www.ilrt.bris.ac.uk/netquest/about/soft/]. Demonstrations available online, such as Crisp and May's Chemistry tutorial,[3] show how a TML file would be rendered as a series of HTML pages viewed by the student.

Authoring Learning Objects – Multimedia

The model for most of the learning materials described above – authored by a subject matter expert, presented in text (even with supporting graphics and animation) – is the book, or at the very least, the course manual or course guide. More and more non-textual resources are appearing every day, however. Video clips, small applets, interactive animations, simulations – these are authored using a wide variety of programs ranging from video editing software to Java editors to Macromedia's Director. Many of these are available online, such as the animated slide show, "Deepest Impacts: A Species Demise E.L.E." [http://www.to-scorpio.com/link3e.htm]. They are developed and distributed because, as J. Bradford DeLong, a developer of several economics animations, writes, "I think that there is a reasonable chance that [they] are – or could become – a vast improvement over the textbook presentation" (DeLong, 1998, para. 3).

Many more resources are not available online. Schools face continual pressure to purchase a wide variety of educational CD-ROMs and teaching software.[4] Thus even online courses present challenges for students and instructors as various software applications need to be purchased, delivered and installed into students' computers. With the emergence of Applications Service Providers (ASP – not to be confused with Microsoft's Active Server Pages) the distribution of software via CD-ROM and floppy disk will slowly evaporate. Application Service Providers are online services that automatically deliver and install software on an as needed basis to client computers (Seymour, 1999, June 28).

Designing Learning Objects – Data or Multimedia

In the previous sections, I have described the mechanics of creating learning objects. Before continuing with the technical description of course components it is important to look at the *content* of learning objects. While there will be, no doubt, much debate regarding the instructional design of learning objects, in practice designers have opted for a performance-based or competency-based theory of design. For example, Cisco System's RIO (Reusable Information Objects) project is explicitly performance based. Drawing on work by Ruth Clark (see [http://www.clarktraining.com/seminars.html] for outlines) RIO "views all training as a means to enable a worker to successfully complete a task." (Wieseler, 1999, p. 9). The process follows three steps:

1. Identify the job task
2. Identify the skills and knowledge necessary to complete the task
3. Develop training in modular chunks that are organized to support the task

Learning, with this model, is outcome-based rather than content-based. It focuses on what people want (or need) to do, rather than on what there is to know. Suppose, for example, Cisco introduced a new product. A traditional approach to training would be to list the product's features, to develop the course based on this features list, and to test students on their recall of the features. A performance-based approach, by contrast, would begin by assessing customer requirements. These requirements would then be matched with product capabilities. Students would be tested on their ability to recommend the product in appropriate situations (Wieseler, 1999).

Most educational institutions would find a definition of learning objects based on specific tasks to be somewhat limiting. However much work has been done regarding the definition of learning *outcomes* in general, and a wider definition of learning objects would be tied to these outcomes. Specifically, the content of a learning object would be derived from a discussion of a course's (or a lesson's) learning objectives, where the achievement of these outcomes can be measured in terms of students' performance. In sum, the overall content of a learning object would be similar in scope and nature to the content of a typical lesson. Many lesson-planning aids exist; the template depicted in Figure 4, from Ohio Schoolnet (n.d.b), is typical.

Figure 4: Lesson Planning Template (from Ohio Schoolnet)

Ohio Schoolnet's template is notable because an instructor may click on any given component to view a detailed description. A learning object authoring environment would employ a very similar interface, while clicking on the component area would enable an editing screen for that component. Thus, for example, if the author clicked on "Learning Objectives", she would be greeted with a list of learning objects appropriate for that course, from which she would select one or more. Or if she clicked on "Tools and Resources" a list of suitable online resources would be displayed.

Authoring Learning Objects – Metadata

For any object, text-based or multimedia, an associated set of metadata needs to be created. The type of object determines the content of the metadata. For example, an image might have a property labeled *photographer*, and a piece of text might have properties labeled *editor* or *publisher*. Whatever the properties, the authoring of metadata itself will be straightforward for most course designers. Because metadata files are machine-writable, authors will simply access a form into which they enter the appropriate metadata information. The form, generated either by a web page or by a specific piece of application software, will send field information to a metadata page editor.

The process of converting form data to XML data is very simple. Here is the code, in perl (assuming forms data has been saved in a standard hash file %FORM):

```
while (($fk,$fv) = each %FORM) {
$output .= ‘‘<$fk>$fv</$fk>$backslash$n’’;
}
print $output;
```

More complex metadata editors will include mechanisms for parsing and displaying existing metadata documents. They will also include forms for a wide variety of resources; the list of fields in these forms are defined by schemas, as discussed above. Sophisticated metadata editors will not define the fields for different types of forms internally. Rather, they will access schemas from various sources around the Internet. A list of available schemas for online learning is provided on the IMS website [http://www.imsproject.org/metadata/mdbest01.html]. The editor will retrieve the titles of these schemas from a central index, and once the author selects a title, will read the specifications and create the form accordingly.[5]

What is significant is that all of this occurs behind the scenes. All the author needs to know is what *type* of metadata is being created, and that type is defined by the type of object being described. As a side note, it is worth noting that

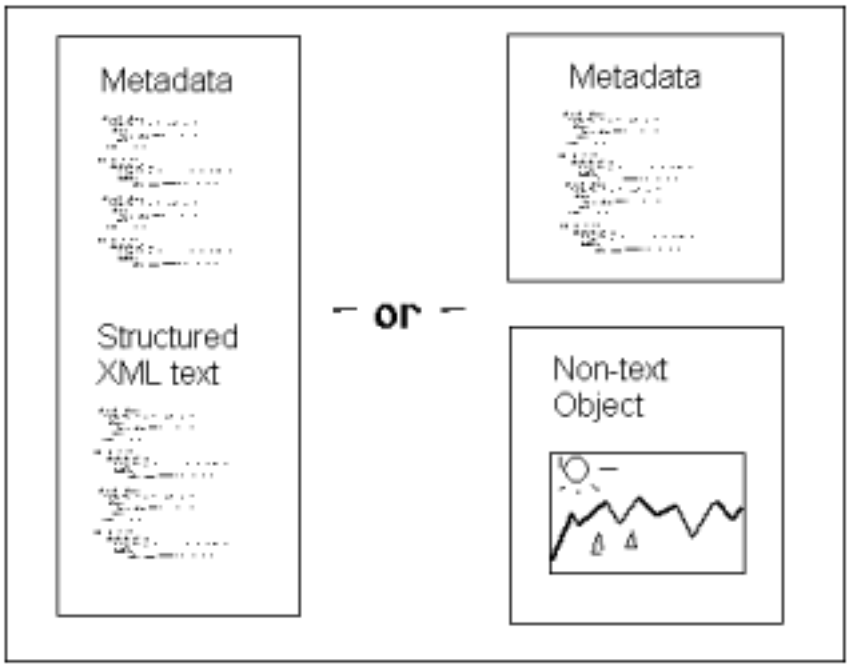
schemas for a wide variety of entities, and not just course components, are being defined in this way (see [http://www.schema.net/] for more information).

In the case of multimedia objects, many editors will have metadata generators built in. This is already the case with some Microsoft products, such as MS-Word, which saves MS Word files in a (Microsoft specific) XML format. Such products will save users the time and trouble of typing the same information over and over (such as their name, institution, and the date).

Authoring Complex Objects

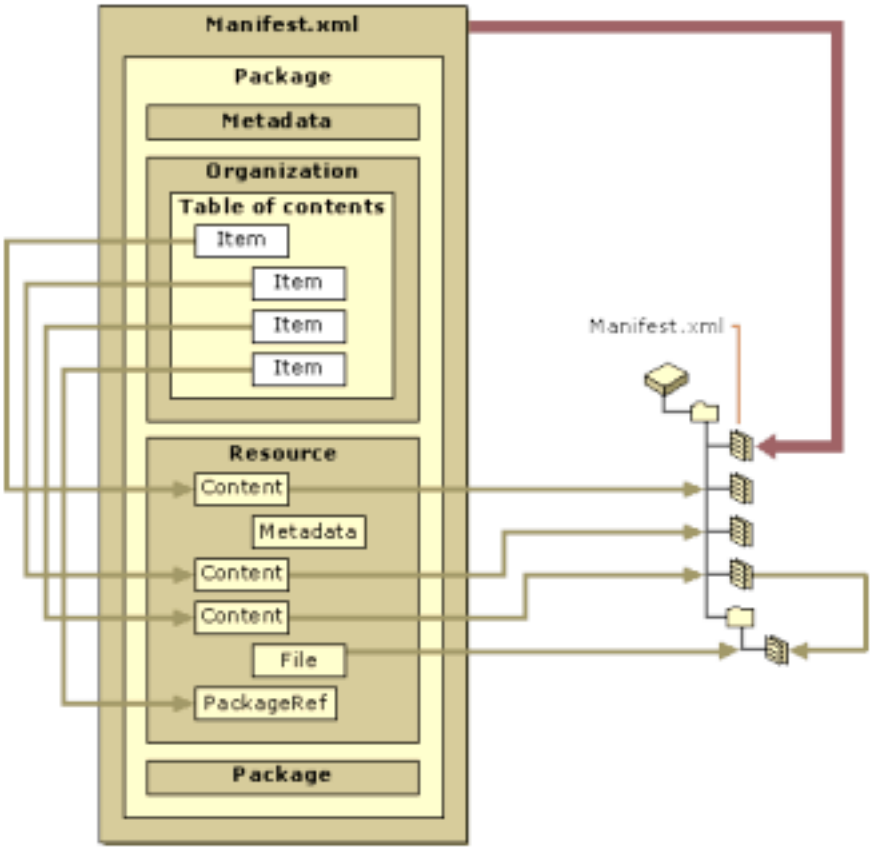
By now you should have the following picture of a learning object (Figure 5) in your mind:

Figure 5: Learning Object depicted



Each of these objects is created and stored in a database. The contents of this database are available to course authors. Some databases may be available over the Internet, while other databases will be available only internally. In order to create a more complex entity, like a lesson, a number of these entities are collected together in what is called a package (IMS Global Learning Consortium, 2000b). A package is a structured representation of a set of independent objects. Microsoft's LMS concept (Figure 6) provides us with a good illustration.

Figure 6: LMS Conceptual Model



(from Microsoft LRN Toolkit, see [http://www.microsoft.com/eLearn/] for more information)

The *manifest* is similar to the shipping label for the package, detailing the contents of the package. The table of contents is an ordered representation of the titles of each item. The metadata for items themselves may be actually contained in the package, or pointed to by a line in the page. Similarly, resources themselves may be contained in the package, or pointed to by a line in the package (obviously, non-textual resources, such as images, must be pointed to).

Again, the package is described in XML. However, it is unlikely that course authors will write this file by hand, or even that they will ever view the file directly. Course authors will generate this file by interacting with it through a program such as Microsoft's LMS (hopefully *not* LMS, because it forces users into a Microsoft-only environment, thus defeating the interoperability require-

ment described above).

How would this work? At this point, much of what follows is speculation, since the required systems have yet to be constructed. Using an authoring tool, an author will select (from a drop-down list) a packaged-sized entity, for example, "Lesson." The authoring tool will retrieve the schema for "Lessons" either from a local database or – better – from a central schema resource online. The schema defines the fields that must be filled out (filling some automatically, especially if the lesson is part of a large project). Additionally, since the object in question is a *package*, the program knows that it will be composed of other objects: an interactive display, for example, a movie, or some other resource. These options are presented to the author: the author selects "insert" and then selects the type of object to be inserted.

At this point, in traditional course authoring, the author would start to write content for the new component. And this will still be an option – if the author selects "new" the appropriate authoring tool will be opened and the author can create a new resource, as described above. But many authors will select from a list of available resources. If the author is authoring a lesson, the course authoring system already has some significant information. It knows, for example, what the topic of the course is, what the grade level is, what the geographic region is, and more. These would all have been defined when the course was created, and these values are inherited by any object that forms a part of the course.

If, then, the author wishes to add a resource, the authoring system has the information it needs to conduct a highly selective search of resources. The system may search a local database, but more likely, it will search an online learning objects repository. Such a repository won't actually *contain* these resources – they will be distributed on websites around the world – but it will contain information *about* those resources. Specifically, it will contain those objects' *metadata*.

The authoring system consults the repository and runs a search. The results of the search are provided in a menu for the course author. Some of these results are approved by standards bodies, and some are not. Some are defined by grade level and even learning objective, as defined by, say, the Western Canada Protocols, and some are not. Some are available for free, while others will require that royalties be paid. The author can instruct the authoring tool to accept only resources approved by a certain standards body or meeting a certain learning objective, or falling within a certain price range. The author at this point may preview the material, or she may decide to insert it into the course. At this point, the metadata – not the object itself – is inserted into the course package. The author moves on to the next item in the lesson, and in a very short time – hours, not days – completes the lesson, and eventually, the course.

Displaying Learning Objects

Learning Object Repositories

Consider the impact of a resource like Martindale's Health Science Guide, a resource center listing 60,000 teaching files and 129,000 medical cases [http://www-sci.lib.uci.edu/HSG/Medical.html]. Such a resource, if made available to medical schools around the world, would greatly facilitate the creation of courses in medicine and could provide a sustaining source of revenue for the Martindale Centre.

The core of a learning object repository is the central database containing the tens or hundreds of thousands of individual objects. Such databases will be multi-functional; online courses constitute only one of the end uses to which these objects will be put (other uses might include online journals and magazines, personal websites, knowledge management applications, and more). Often, these databases will be working databases for separate enterprises entirely. For example, a government may place all legislation, regulations, procedures manuals and tables into a database. This information would be accessed by an array of applications and end users, including lawyers, real estate agents and the press. The very same set of resources would also be made available to online courses. Attached to each object in the database will be a metadata file, as described above. This file will include subject-specific information, but also, information applicable to online learning (such as grade level, subject area, and more). The cost structure for materials retrieval will also be included in the metadata.

A system of learning objects repositories around the world (it is very unlikely that there will be one) will access this metadata to form its own, compiled, set of resources. The online learning repository will retrieve only that metadata relevant to online learning. It is this, filtered, metadata that will be accessed by online learning systems. Such a system may seem far-fetched but is already being implemented in online journalism. Content producers, such as Reuters, provide their material to content syndicators, such as I-syndicate [http://www.isyndicate.com/], Individual.com [http://www.individual.com/], or Netscape Netcenter [http://my.netscape.com/]. These sites retrieve the content – in the form of Rich Site Summary (RSS) or other XML-type files, process it for display, and relay it to individual users. Individual users, playing the role of newspaper editors, can create customized daily newsfeeds that appear on a web page or in their email every day.

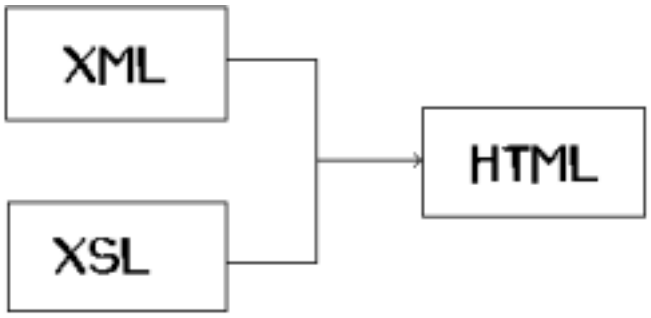
Existing learning portals, such as Learn2.com, HungryMinds, Learn.com, Fatbrain, and SmartPlanet, are beginning to move toward this model of content delivery (Barron, 2000; Karrer, 2000). Topic specific business-to-business learning portals are providing customized learning from within the context of learning

management systems. TrainingTek.Com, for example, allows course designers to select learning objects from a menu of options within the context of their learning management system [http://trainingtek.com/]. A similar resource was recently launched by Internet and publishing giant, America OnLine (2001).

Displaying Learning Objects

Because learning objects are distributed as XML files, they may be displayed using a wide variety of hardware and software combinations. The simplest and most straightforward implementation of this is through the conjoining of XML files with related style sheets defined in XSL (extensible Stylesheet Language), as mentioned above. For example, an XML file and an XSL file merge to create an HTML file, as depicted in Figure 7.

Figure 7: XML and XSL merge to create HTML



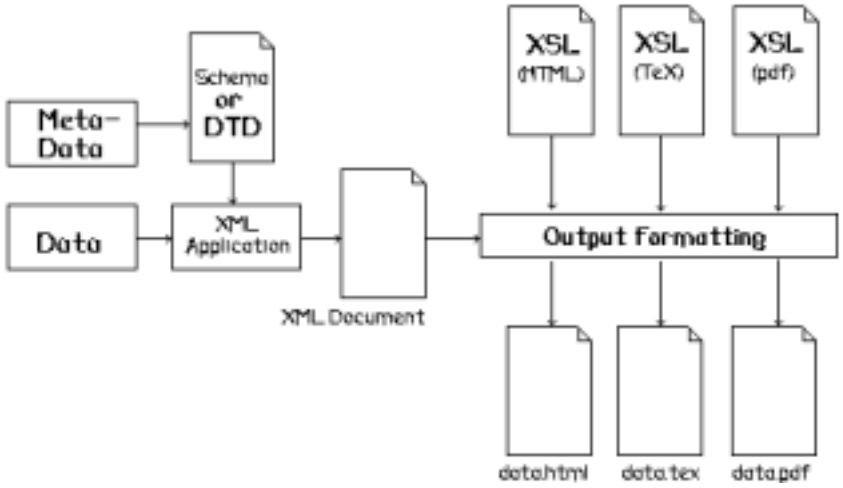
In this simple example, each element in an XML file is associated with an output format (defining such things as font styles and sizes or background colours) in the XSL file. The XSL file describes these elements in standard CSS (Cascading Style Sheet) (World Wide Web Consortium, 2001). Thus combining the XML and XSL definitions yields an HTML output understood by web browsers. Because style and substance are separated, the same XML data may be output in a variety of formats. Figure 8 depicts this process.

(adapted from Bourda & Hélier, 2000)

The learning object itself is composed of data and metadata (left), stored in a database. An XML application interprets the data through a schema, which defines the properties that will be displayed in the resulting XML document. This document is sent to output formatting, which will use one XSL file to produce an HTML page, another XSL file to produce a LATEX page, and yet a third XSL file to produce a printable PDF page.

This model provides tremendous flexibility. For a given set of data, one set of

Figure 8: Preparation of XML data and examples of output formatting



metadata may produce an XML file useful for online students while another may produce an XML document useful for real estate agents, depending on which schema is used. In addition, any given XML document may be output in a variety of formats, depending which XSL file is used. One XSL file may define the format used by the University of Chicago while another may define the document template used at the University of Toronto. One style sheet may be used for online viewing, while another may define the print version of the document. Indeed, an XML file, merged with other XML and XSL sources, may be displayed in a highly customized or personalized format. Proposed, for example, is an agent-based learning system that recognizes individual users and formats pages accordingly. This agent would alter not only display preferences, but would also amend content according to previously established user preference (Suzuki & Yamamoto, 2000).

In traditional education, learning objects would be distributed in the context of a course; that is to say, an online course would consist of an ordered collection of related learning objects unified by some set of common learning objectives or course-wide assessment techniques, such as a final essay. Learning objects would also be used in a wide variety of non-traditional educational scenarios.

No matter how learning objects are distributed, their method of access will be similar in every event. By clicking on a link (or making a similar selection), a student will from his or her desktop browser send a request to the object, which will be delivered as an XML file. The request will also refer the distributor to another XML file, which will contain user specifics, such as their name, institution, and course. On the distributor end, the learning object will be

packaged and send to the client. In many cases, the package will be a simple XML file, though often, the XML file will be compiled on the fly in response to the user data sent with the request. The distributor will also note the request in its logs, update its billing or license data, and if requested, send notification to the student's employer or educational institution.

The object, when retrieved, is then fed to a viewer. The viewer may be included in standard web browsers (as, for example, graphics are today) or may require a separate viewer. Viewers may be defined (and even available) through the institution's web site. Most likely, specialized viewers will be downloaded and installed on an as-needed basis as java applets are today. Different viewers will be used for different types of output device. This, just as graphics today may be viewed on a screen, send via fax, printed on pager or sent as an email attachment, so also different viewers for each type of learning object will be available for different distribution mechanisms.

The Learning Object Economy: A Polemic

We can be visionary. We can imagine a proliferation of cottage industries involved in the production of learning objects. Standards bodies, reviewers and other filter mechanisms will become important. Because a payment scheme is built-in, the model becomes sustainable. But because each individual object view is so inexpensive, online learning becomes affordable.

Yet what about traditional university education, where professors see their courses as unique creations which re-make the field of enquiry each time they are taught?[6]

This approach is the core of traditional liberal arts education. It is this very aspect of online learning which pits computer-assisted learning, such as is envisioned in a learning object economy, against traditional face-to-face professorial learning. Let me grant that this sort of reexamination of the material is necessary and desirable. But let me question whether this process at the same time serves as an effective teaching methodology.

To put the question in as sharp a light as possible: do first-year engineering students need a *brand-new* Shakespeare course, or will the interpretation developed last year (or 2 years ago, or in Saskatchewan) do the job? And moreover: is it fair to require that students, whose primary goal is at best a surface understanding of "Hamlet" to *pay* for the development of a *brand-new* interpretation, when last year's, or Saskatchewan's, would have done just fine?

I agree that hand-rolled bread, carefully prepared by a master chef, is superior in quality to a standard loaf purchased at a supermarket. But to a person who is merely hungry – rather than a connoisseur – the obligation to purchase *only* hand-rolled bread is more than just an imposition, it amounts to a denial of

basic sustenance for many.

The question is: *could* we teach first-year English using "Hamlet" modules? *Could* we reduce the cost of such learning by an order of magnitude? Are the endless creations of professors *necessary* for the eventual goal of cultural literacy? Is it reasonable to *deny* such an education to many (especially in less developed nations) in order to generate each course anew each year in each university classroom?

I apologize for this liberal use of italics, but I am trying to emphasize how it looks from the other side of the equation. And I'm trying to express the sort of thinking when such object-based courses are inevitably accredited. How will the hand-craft institutions justify their art? Granted, we need reinterpretations of Shakespeare, but do we need a thousand such reinterpretations a year?

There is very much a tension, between those who create the knowledge, and who jealously guard their monopoly over its propagation and distribution, and those who must consume that knowledge to get a job, to build a life, to partake fully in society. My personal belief is that arts and humanities professors – even those who teach senior courses – will have to redefine their approach or be priced out of existence. Probably history, not argument, will show whether this belief is well founded.

References

Advanced Distributed Learning Initiative. (2000). Sharable Courseware Object Reference Model (SCORM) (version 1.0). Retrieved May 30, 2001: [http://www.adlnet.org/Scorm/docs/SCORM_2.pdf]

Advanced Distributed Learning Initiative. (2001). Sharable Courseware Object Reference Model (SCORM) (version 1.1). Retrieved May 30, 2001: [http://www.adlnet.org/Scorm/downloads.cfm]

America Online. (2001). AOL-at-School. Retrieved May 30, 2001: [http://school.aol.com/teachers/index.adp]

Bates, A. W. (2000). Managing technological change. San Fransisco: Jossey-Bass.

Barron, T. (May, 2000). A portrait of learning portals. ASTD Learning Circuits. Retrieved May 30, 2001: [http://www.learningcircuits.com/may2000/barron.html]

Bourda, Y. & Hélier, M. (2000). What metadata and XML can do for learning objects. Webnet Journal, 2(1), p. 29.

Brickley, D. (n.d.a). TML language specification. Institute for Learning and Research Technology: University of Bristol. Retrieved May 30, 2001: [http://www.ilrt.bris.ac.uk/netquest/about/lang/]

Brickley, D. (n.d.b). Tutorial markup language (TML). Institute for Learning and Research Technology: University of Bristol. Retrieved May 30, 2001: [http://www.ilrt.bris.ac.uk/netquest/liveserver/TML_INSTALL/doc/tml_user.html]

Council for Higher Education Accreditation. (1999). Distance learning in higher education, CHEA update number 2. Retrieved May 1, 2001: [http://www.chea.org/Commentary/distance-learning-2.cfm]

DeLong, J. B. (1998). Multimedia. Retrieved May 30, 2001: [http://econ161.berkeley.edu/multimedia/Multimedia.html]

Downes, S. (2000). Hungry minds: A commentary on educational portals. Online Journal of Distance Learning Administration, 3(1). Retrieved May 1, 2001: [http://www.westga.edu/\char126\relaxdistance/downes31.html]

Downes, S. (2000, July/August). MarcoPolo. The Technology Source. Retrieved May 1, 2001: [http://horizon.unc.edu/TS/default.asp?show=article\&id=792]

Dublin Core Metadata Initiative. (1999). Dublin Core metadata element set (version 1.1: reference description). Retrieved May 30, 2001: [http://dublincore.org/documents/1999/07/02/dces/]

IMS Global Learning Consortium. (1999). IMS enterprise information model (version 1.01). Retrieved May 30, 2001: [http://www.imsproject.org/enterprise/eninfo03.html]

IMS Global Learning Consortium. (2000a). About IMS. Retrieved May 30, 2001: [http://www.imsproject.org/aboutims.html]

IMS Global Learning Consortium. (2000b). IMS content packaging: Packaging information model (version .92 – public draft specification). Retrieved May 30, 2001: [http://www.imsproject.org/content/cpinfo02.html]

IMS Global Learning Consortium. (2000c). IMS Question & Test Interoperability Information Model Specification (version 1.0 – final specification). Retrieved May 30, 2001: [http://www.imsproject.org/question/qtinfo01.html]

IMS Global Learning Consortium. (2001). Specifications. Retrieved May 30, 2001: [http://www.imsproject.org/specifications.html]

Karrer, A. (May, 2000). Building a learning portal. ASTD Learning Circuits, May, 2000. Retrieved May 30, 2001: [http://www.learningcircuits.com/may2000/may2000_ttools.html]

Montlick, T. (1999). What is object-oriented software? Software Design Consultants. Retrieved May 30, 2001: [http://www.catalog.com/softinfo/objects.html]

National Association of College Stores. (2000). College Store industry financial report 2000 edition. NACS: Oberlin, OH.

Ohio Schoolnet, (n.d.). Lesson planning template. Retrieved May 30, 2001: [http://tlcf.osn.state.oh.us/blueprint/index.html]

Paille, G., Norman, S., Klassen, P. and Maxwell, J. (1999). The effect of using structured documents (SGML) in instructional design. In NAWeb Conference Proceedings.

Seymour, J. (1999, June 28). How Application Service Providers will change your life. The Street.Com. Retrieved May 30. 2001: [http://www.thestreet.com/comment/techsavvy/759956.html]

Suzuki, J. & Yamamoto, Y. (2000). Building a next-generation infrastructure for agent-based distance learning. Manuscript submitted for publication. Retrieved May 30, 2001: [http://www.yy.cs.keio.ac.jp/\char126\relaxsuzuki/project/pub/ijceell.pdf.zip]

The Bible (n.d.). Genesis. Retrieved May 30, 2001: [http://www.genesis.net.au/\char126\relaxbible/kjv/genesis/]

Wieseler, W. (1999). RIO: A standards-based approach for reusable information objects. Cisco Systems White Paper. Retrieved May 30, 2001: [http://www.coursenet.com/media/pdf/RIO_Whitepaper_techlearn.do.pdf]

World Wide Web Consortium (1997a). Extensible Markup Language (XML). Retrieved May 30, 2001: [http://www.w3.org/XML/]

World Wide Web Consortium, (1997b). Extensible Stylesheet Language (XSL). Retrieved May 30, 2001: [http://www.w3.org/Style/XSL/]

World Wide Web Consortium, (2001). Cascading style sheets. Retrieved May 30, 2001: [http://www.w3.org/Style/CSS/]

Endnotes

1. Of course, this is not strictly true; with Microsoft's cooperation, vendors could create translation engines which would 'import' MS Word documents – but always with a loss of formatting.

2. World Wide Web Consortium. XML Schema Part 0: Primer. W3C Working Draft, 07 April, 2000. Retrieved May 30, 2001: [http://www.w3.org/TR/xmlschema-0/]. Schemas are more recent implementations of the (perhaps) more familiar DTD (Document Type Definitions).

3. Crisp, Joel, and May, Paul. This is an example tutorial for Chemistry. Web site. Undated. Retrieved May 30, 2001: [http://www.ilrt.bris.ac.uk/netquest/liveserver/cgi-bin/tml.pl/netquest/liveserver/qbanks/demos/paul/chemtute/chemtute.tmlo?Intro=/netquest/about/demos/]

4. As a designer for a Grade 12 mathematics course in Manitoba, Canada I had to deal with the fact that some specific calculator software was actually embedded into the western protocols mathematics curriculum. See Western Canadian Protocol, The Common Curriculum Framework for 10-12 Mathematics. June, 1996, p. 99. Retrieved May 30, 2001: [http://www.wcp.ca/math/10-12/cluster/applied.pdf]

5. See, for example, Liljegren, Jonas. RDF Schema editor. Retrieved May 30, 2001: [http://paranormal.se/perl/proj/rdf/schema_editor/]. Liljegren's schema editor retrieves schemas and, on the basis of the retrieved schema, generates a form for inputting new objects.

6. My thanks to Terry Butler for this phrasing and for motivating the polemic which follows.

Citation Format

Downes, Stephen (2001) Learning Objects: Resources For Distance Education Worldwide. *International Review of Research in Open and Distance Learning*: 2, 1.
http://www.icaap.org/iuicode?149.2.1.2