# Creating Learning Objects

***Carlos Francisco Lerma***
***General Directorate of Technological Innovation***
***Universidad Autónoma de Tamaulipas,***
***Ciudad Victoria, Tamaulipas, México***

cflerma@uat.edu.mx

## Abstract

The changes that computers have introduced to society include the way and speed in which humans acquire knowledge and process information. The use of computers has been driven by academic research, and it is within academic settings where the use of computers has been felt stronger by society.

This paper identifies the different types of learning objects and determines the factors that should be taken into consideration in the creation of learning objects, along with the different tools needed to create them.

Next, we will analyze the factors that should be taken into consideration in the creation of these modules, such as learning types, social and psychological factors of learning that guide the creation of learning objects; tools and recommendations proposed to facilitate the creation of learning objects and economic costs of developing these materials.

**Keywords:** learning, objects, creation, types, metadata, cost

## Introduction

Ever since its conception, computer software has been seen as a potential tool used as an effective aid in the learning process of many areas of knowledge. Fighter pilots, bus drivers, and even car sales personnel have benefited from the extensive use of electronic software tools that provide not only a situational, experiential and dynamic learning environment, but also, in many cases, a set of learning guidelines that take the students through a training/learning process that eases the acquisition of knowledge and reduces the required time to learn new skills or concepts.

Learning Objects represent an important element when using electronic media to deliver educational contents to an audience in a learning environment. Learning Objects provide both of the aforementioned concepts: the basic building blocks used to construct the software tools and a dual-purpose method that not only points out the way to use the software tools but also serves as a guideline to deliver content in an organized and effective way. Experienced educators have benefited from these tools by adding powerful and dynamic tools that enhance and enrich the overall learning experience. Novice educators with no pedagogical experience adopt them as a tool that enables an effective transfer of their experiential learning to students while providing them with an

ordered method to structure a course's lessons.

Learning objects have proven to be effective as aids to accelerate the process of learning. But their construction does not involve a simple method, as their nature is directly linked to the intricacies of human cognitive and learning processes. Learning objects differ from ordinary software development projects because their assembly is not only guided by simple client requirements but by the learning particularities of the students that will use them. A learning object used to train fighter pilots will differ significantly from one that will be used by car salespersons. Taking this fact as starting point, not only is the target audience the main factor to take into account but also the sociological, psychological, and cognitive particularities of that specific target audience, which will be assessed later in this chapter.

## Types of Learning Objects

Software tools revolutionized many fields of science due to the practical and almost immediate benefits that information processing brought to humans in the 1960's when the use of computers to automate the processing of large amounts of information became an everyday reality. Being evolution a natural process in almost every aspect of human life, software was not the exception. Learning objects derived directly from the use of software tools with educational purposes, becoming self-contained elements whose main application was to aid in the teaching of specific concepts. As special types of software became specifically learning-oriented, learning objects appeared as pieces of electronic media with specific application in the educational field.

The term "Learning Object" derives initially from Object - Oriented Programming, a modern trend in software development focused towards the use of visual elements integrated into interface design, and its application into development efforts whose main purpose is the creation of software with educational purposes.

Wiley (2000) provides a classification of types of learning objects, making it easier to understand the nature of the elements contained inside of them:

- *Fundamental* – The basic, most simple form of learning aid. A simple image depicting a stage of a surgical procedure
- *Combined-closed* – Still a simple element but one that integrates more complex mechanisms in order to provide an explanation. An animation or video clip depicting a surgical procedure, including audio
- *Combined-open* – Several simple objects encased inside integration elements. An integration element (i.e. a website) that includes the image and the video clip of the surgical procedure, along with the use of plain text that explains the procedure
- *Generative-presentation* – Combination of objects providing advanced visual and auditive capabilities with limited interactive features. A dynamic Flash animation capable of generating and recreating a visual picture depicting a surgical procedure and its inherent operational conditions
- *Generative-instructional* – Combination of objects providing advanced visual and auditive capabilities with advanced interactive features, allowing a high level of hands-on experience. A dynamic Flash animation, linked to an image and text database, capable of generating a graphic environment depicting a surgical procedure and its operational environment, where users can manipulate surgical instruments and monitor patient vital signs in order to provide hands-on instruction

Allert et al. (2004) provide a purpose-oriented definition of learning objects. This definition helps understand the scope of a learning object in terms of its purpose, be it direct instruction of concepts or providing guidelines and/or methods to use learning objects in learning environments:

- First – Order Learning Objects (FOLOs): These are "resources which are created or redesigned towards a specific learning objective". These learning objects depend on the final objective of the learning process, making them specific to the concepts they aim to explain. Logically, the learning objectives or final concepts will define the form of the learning object and its content. In concrete terms, FOLOs represent the prime matter used to initially construct learning objects. Texts, images, animations or videos fall into this category since they are used basically to give form to learning objects.
- Second – Order Learning Objects (SOLOs): Defined as "resources which provide and reflect a strategy", SOLOs define guidelines and strategies to plan a learning environment. They also provide aid and serve as a strategic plan to promote the creation of knowledge.

Both classification criteria illustrate an important fact about learning objects, they are small elements that combine themselves to form self-contained systems or sub-systems whose nature can be objective (concept specific, tangible learning materials) or subjective (learning methods and systems focused towards the transfer or acquisition of knowledge). Still, it was previously noted that learning objects are dynamic in terms of their nature. Technology is not static and its inherent evolving characteristics suggest several aspects regarding the properties of learning objects that need to be taken into account. Table 1 shows those properties in detail.

**Table 1. Goals of Learning Objects**

**Source: WBITC Trends. http://www.wbtic.com/trends_objects.aspx**

| Goals of Learning Object Design | |
|---|---|
| **Goal** | **Description** |
| **Reusability** | Learning content modularized into small units of instruction suitable for assembly and reassembly into a variety of courses |
| **Interoperability** | Instructional units that interoperate with each other regardless of developer or learning management system |
| **Durability** | Units of instruction that withstand ever evolving delivery and presentation technologies without becoming unusable |
| **Accessibility** | Learning content that is available anywhere, any time—learning content that can be discovered and reused across networks |

## Static Learning Objects

Additionally, as we discussed the major classifications of learning objects based in their inherent nature, Table 1 does not cover a classification of learning objects based on their dynamic and evolving nature. Learning objects can be produced as software materials intended to be fairly static and constant. Their contents abide to a specific form and style that does not vary throughout the overall learning experience. This approach is useful in a controlled learn-

ing setting where participants have almost equal levels of experience and learning capabilities. Such learning objects can be found in professional training materials used for professional certifications offered by IT/Telecomm companies like Microsoft or Cisco Systems where simulators, animations and explanations follow a predefined path that does not vary in terms of difficulty levels or overall lesson detail.

Static Learning Objects are helpful in instructional settings with simple and/or very straightforward requirements, generally similar to those in which educators need to have full control of the learning method and process. This may become a drawback when educators need to adapt to the ever changing way of knowledge acquisition of students. In order to fulfill this need, Dynamic Learning Objects are conceived and designed to be used within learning environments where an educator encounters variable levels of skill, experience, retention and reasoning amongst a group of learners. Dynamic Learning Objects represent a great challenge when adapting educational software to the true nature of humans' learning methods.

## Dynamic Learning Objects

The nature of the cognitive/learning process in humans has been identified by extensive research as a dynamic process that takes many forms and changes its path from one individual to the other. Learning/cognitive processes take place inside the learning individual, regardless of the delivery method used by the teaching entity. The method relies on the most successful learning experiences or methodologies previously used by the individual in order to understand any given topic required for him to learn. On most occasions, a student learning any given topic will focus his/her senses towards the teacher in a lesson with the sole purpose of acquiring knowledge in the first place but it is the dynamic nature of the assimilation, synthesis and retention processes that follow what becomes the main focus when educators and developers try to design learning objects that adapt to those variable forms of learning. Harnessing and assimilating the ever-changing nature of the aforementioned processes will result in effective methodologies and procedures that contribute to elevate the effectiveness of learning objects by educators or developers.

Different fields of science might require different learning methods for different students. A simple analysis of learning techniques show us that Mathematics and applied sciences rely on clear explanations followed by repetitive exercises in order to grasp a concept (i.e. How to solve an equation or how to determine a finite amount based in different factors that integrate a formula). Social sciences and arts rely more on reading comprehension, image association and pure concept retention in order to understand a series of events (i.e. a synopsis of a group of events that conform a historical era or event like the French Revolution or World War II). Languages and linguistics make use of associations of visual elements of characters and pieces of text and auditive resources in order to explain how to group characters or words to construct phrases or sentences in order to speak or write a given language, the latter concept exemplified in Lam et. al. (2004) through research conducted in China with the purpose of addressing those needs while learning to handle the complexities of Chinese characters.

These variations represent a starting point towards understanding of the use and selection of teaching/learning methods that give form to learning objects development. Adapting those combinations of learning methods to the modus operandi of learning objects is crucial to ensure that the latter can be effectively used by heterogeneous groups of students where learning capabilities and overall skill levels vary greatly. Conquering this obstacle also preconceives an important expectation: the fact that learning objects can change its form, level of difficulty, delivery method and overall information load by means of information collection resources such as surveys and questionnaires whose results serve as guidelines to dynamically generate learning objects and logical paths to use them.

Additionally, there is an important distinction to be made in relation to dynamic learning objects design (which can also be applied to static learning objects): the type of design orientation. This characteristic clearly identifies the type of approach used in the design process of the learning object, which can be teacher-oriented or student-oriented. Teacher-oriented learning objects are developed with teachers' methods and preferences as the main design element and are preconceived to follow teachers' methods and techniques in relation to the learning path that control the flow of activities during the lesson. This design orientation is useful in learning object design projects intended for use in groups displaying rather stable learning conditions and groups of students with homogenous learning skills and expertise.

Student-oriented learning objects are developed around the students' requirements and preferences. This learning object orientation tries to adapt lesson flow, contents, difficulty level and lesson pace to the students' learning/cognitive methods and capabilities by gathering information before the start of the lesson. This enables the learning object to determine what pieces of information must be chosen as lesson elements, what aids must be invoked to provide additional support to the lesson, what type of elements (visual, auditive or both) the user prefers to manipulate and if the lesson pace will be set to fast, slow or an intermediate spot.

The aforementioned classification criteria for learning objects enumerates the main elements that need to be integrated and the orientation needed by the final product in order for it to be useful in real life. We can see that centering the development method on users' requirements/preferences is a good choice when trying to narrow the gap between man-made learning aides and human cognitive processes. But, even when this is a highly desirable result, it is far from being the solution to all the issues regarding learning object design. There might be specific situations in specific fields of knowledge in which instructors need to have control of what is being taught, enabling them to have a more effective control regarding the learning process of students. Equally, tight control and pre-defined learning materials designed without taking into account the variations in a learning environment have an equally negative impact as the opposite orientation scheme we previously mentioned.

An example of an entry-level dynamic learning object is proposed by Lam and Ki (2004) to aid students learning Chinese. The learning object proposes the use of an operational framework coupled with visual and auditive aids whose properties can be dynamically manipulated by the user in order to understand inherent variable elements in the Chinese language, specially those visual (written) and auditive (sounds/pronunciation).

This dynamic learning object is comprised of a tool whose main function is to display Chinese characters and their variations. Users select characters commonly used in Chinese and perform different activities, like changing the form of the characters by altering the strokes of which they're made of or comparing symbols that bear close resemblance to one another not just graphically but also in regards of their meaning. More advanced activities include constructing a character based on certain strokes with the help of aides like a flashing pointer whose blinking increases as users come close to correctly assemble a character and auditive components that reproduce spoken versions of assembled characters as they are combined after assembly to form sentences or paragraphs.

The application and adaptation of dynamic learning objects to the intricacies of human learning processes is particularly interesting in this case, due to the fact that these learning objects are quite simple but their design enables them to adapt to different learning levels and capabilities in extremely heterogeneous groups. This model was field-tested in Hong Kong using groups of students in different learning settings taking into consideration differences between students like their variable skill levels and academic progress rate. Positive results were ob-

tained from tests, having the tool been accepted by teachers and students and considered as a helpful aid.

## Technologies Used in Learning Object Development

In order to begin the construction of a Learning Object, one of the elements that developers and educators alike must choose is the technology they will use to build their product. This part is essential in the construction process because it resembles the use of a tool to perform a task by a construction worker: it relies heavily on how comfortable, experienced and knowledgeable a worker is when it comes to using that specific tool (like a sledgehammer, a chisel or a nail gun). Expertise in terms of the use of a development tool when it comes to produce a quality learning object is crucial in order to deliver the desired knowledge or instruct a subject in terms of a specific topic because the developer knows which features and special characteristics he/she must apply when using the desired development tool according to the project in progress. The objective is to use those development tools and their features to the fullest in order to produce a learning object that can accurately deliver the knowledge to the end user, reduce the time it takes to effectively acquire and assimilate that knowledge and adapt to the multiple ways in which students synthesize information internally.

There is not a specific technology or tool that is considered the "Magic Bullet" when it comes to effectively build a learning object. The choice depends on many factors that affect any software development project:

- Initial Requirements: First step towards the start of most projects, analyzing project requirements and objectives helps to eliminate possible candidates in terms of tools to be used to develop a learning object.
- Functionality: Specific tools might not be able to perform or design a certain function or movement specifically needed in order to deliver a concept or explanation.
- Developer's Expertise: Familiarity with a certain development tool or method is an important factor when a developer chooses which software he will use to build a learning object. It is important to contemplate this variable in order to get the most out of our developer's efforts and expertise.
- Visual Impact: Certain learning object development projects require a high degree of visual dynamism. Certain tools provide better ways to handle graphics than others.
- Element Interaction: The degree in which a development tool can handle the fusion of different types of media like text, audio, video and animation; in a single object.
- Compatibility/Integration: The ability of a tool to be able to interact with file types from other tools. This feature makes it easier to compensate flaws or pitfalls between two or more tools in a development project. Compatibility to run on different platforms and adapt to different hardware configurations is also a very important issue when selecting learning object tools.
- Ease of Use: Related to the second entry in this list, ease of use focuses more on novice educators or developers and how fast they can learn to use software tools to develop learning objects.
- Cost: Choosing authoring tools and technologies can also be directly linked to project budgets constraints. Well-funded projects with no budget limitations can

afford to use advanced tools, while projects with poor funding and budget constraints can benefit from the use of open-source or freeware applications, programming languages and platforms.

It is clear that there is not a technology focused towards the production of specific types of learning objects. Developers and educators alike can make use of any technology available at hand as long as the final product clearly fulfills the educational roles that it is supposed to play. In relation to the functions that development tools fulfill there is the need to establish a significant difference between two groups in order to better understand their roles inside of an educational setting. One group of technologies is intended to produce learning objects *per se*, while a second group provides the platform from which those learning objects are stored, managed and delivered to end users according to their needs and requests. Last, an information component or property embedded to learning objects is needed in order to ease on the identification, classification and storage of learning objects contained in repositories.

## *Development Tools*

Development tools are intended to produce learning objects in the first place. They provide means to manipulate text, images, video and other basic media. The applications or learning objects developed by using these types of tools represent what can be called a "front end": The elements that are closer to the user and that will be directly manipulated by it. Their secondary function consists in integrating or packaging those basic elements into one single entity which will become a learning object. Additionally, advanced development tools provide means to integrate educational methodologies that guide the developer/educator into organizing their content in order to maximize the effectiveness. García and García (2005) propose an advanced tool that not only integrates basic element editing and integration capabilities into a single software product, but also provides guidelines to organize and structure the content of a learning object according to learning ontologies, achieving a high level of quality in terms of the final product.

Development tools include, but are not limited to:

### Office suites

Office suites like Microsoft Office, Corel WordPerfect Office and StarOffice (by Sun Microsystems) provide tools to perform basic actions like text editing and spreadsheet generation. Presentations generated by components of office suites like Microsoft PowerPoint have been used in classrooms extensively and provide a good tool to organize the contents of a course. They also have good capabilities to integrate different types of media into a single learning object.

### Hypertext editors

These tools allow users to create Hypertext documents (webpages or websites) by using a simple interface. Because of the fact that building a webpage involves using Hyper Text Markup Language (HTML), users who lack the programming skills can still create hypertext documents by taking advantage of a graphic environment in the same way that skilled programmers would by manipulating commands and HTML tags. Hypertext editors also allow integration other types of media by linking them into a single document.

### Vector graphics editors

These tools allow users to produce high-quality animations. These visual aids are very popular amongst graphic designers due to the fact that file length is typically short and issues like movement, image transition, morphing and the inclusion of sound and text is almost seamless. Once compiled into a single file, objects produced by vector graphics editors are also able to run on any

type of computer without the need of a special software or plug-in, while still retaining their original features and quality. Some of these editors provide developers additional tools to manage the flow of presentations and handling the relations between the elements included within the lesson, making it easier for educators to control the pace of the lesson and to visualize the entire contents of the course (i.e. by generating maps of elements). Examples of vector graphics editors include Macromedia Flash, Director and Authorware, the latter being an example of a product that includes the aforementioned educator-oriented aids to control lesson flow and overall visualization capabilities.

### Advanced programming languages

These tools provide developers and educators with the building blocks to create advanced applications that can be adapted to become learning objects. Programming languages and techniques also allow for accurate planning and development of applications according to very specific requirements. By manipulating commands and instructions defined into a programming language's structure, a developer can create application that can run in a stand-alone manner or by using alternate pieces of software. They also provide capabilities to extend different software tools like the ones listed above, by providing the means to modify them and connect them to operating systems and databases. Programming languages provide the maximum level of application customization and the resources to create brand new software products. Additionally, techniques like object-oriented programming (currently supported by most programming languages and commercial software development products) allow for the creation of pieces of software that integrate a graphic interface, allowing for the manipulation of elements like buttons, forms and text boxes that end users can easily understand and work with. However, in order to benefit from these features and capabilities, developers and educators require having knowledge, training and expertise in the use of advanced programming languages. This condition makes them more difficult to use than any of the previously mentioned tools. Examples of these tools include most programming languages used today, like Visual Basic, PERL, C++, Java and Development suites like Microsoft Visual Studio .NET and Sun Java Studio.

## *Platform Tools*

While development tools provide the means to build learning objects, platform tools provide the infrastructure needed to store, run and distribute learning objects to end users who require them. Platforms are robust operating systems installed on high-performance pieces of computer hardware whose main purpose is to control and manage resources stored in them while handling the requests for access to files from multiple clients at once. Platform tools also organize content by cataloguing it and keeping it ordered according to designers' needs. Platform tools also house systems whose sole purpose is to provide management of learning objects according to a specific deployment.

Platform tools include, but are not limited to:

### Operating systems

They represent the most important software installed in a computer, since it allows it to run in a proper manner. Operating systems define how many and which people can access resources stored in a computer, restricts storage quotas for users, allows or denies access to files and resources and makes them available to a large amount of users, amongst many other activities. Operating systems are designed to be installed either in smaller personal computers (desktop operating systems) or large servers (network operating systems). Learning objects can be stored and distributed from both platforms, but the nature of learning objects is more focused towards multiple access of files by multiple users, making network operating systems the *de facto* choice for

learning object storage and deployment. Additional platform tools like databases are supported by operating systems, since the latter becomes the underlying support structure on which the former are built on. Operating systems include the Microsoft Windows family of operating systems (Windows 2000, XP, 2003), Solaris (by Sun Microsystems), UNIX, Linux and Macintosh's Mac OS as some of the most popular names in the industry nowadays.

### Databases

Tools designed to maintain an ordered structure of elements housed inside a computer. Databases apply methods of organization to large groups of information that needs to be displayed or transmitted in an ordered manner. Databases work by ordering data according to search criteria that looks for records inside of cataloguing units called tables. Subsequently, tables contain organizational units named fields, in which special information is stored. A group of fields subsequently forms a record. Query engines look for specific information contained inside of fields, generating matches once information has been found. Databases are mounted on top of operating systems in order to consolidate a database server. Examples of databases used widely today are Microsoft SQL Server, Oracle, Informix, DB2 (by IBM) and MySQL.

### Learning management systems (LMSs)

These tools provide a management system with organization and classification capabilities to deliver online courses and lessons to groups of users. In general, a learning management system can be defined as a software tool that manages educational content and resources stored in a repository (server) in order to deliver them to users in a controlled learning environment. These can either be bought from manufacturers or built from scratch by a developer or educator according to specific needs. They are also mounted on top of an operating system in order to work, but they limit their management capabilities to the educational content stored in the server rather than the overall resources of it. Learning management systems also perform important tasks regarding the delivery of learning objects and courseware by tracking user habits, how many times are resources accessed, can send and receive e-mail, establish collaboration tools like discussion forums, reception of documents and delivery and analysis of tests. Depending on specific needs, learning management systems can be tweaked to include additional custom features. Popular commercial solutions include WebCT, Blackboard and Microsoft SharePoint Portal Server.

Xuan et al. (2004) propose what is called a "Learning Objects Management System" (LOMS), similar to a Learning Management System but exclusively focused in managing learning objects stored in a server, rather than the functions of a full-scale learning management system. This system integrates a new type of management strategy for learning objects because many LMSs limit themselves to the management of courseware, but not of learning objects *per se*, providing a tool whose importance lies in extending the cataloguing and organization of learning objects inside of a plain LMS. By extending these capabilities inside of a LMS, the provisioning of learning objects to the user can be more accurate according to the type and special characteristics of learning objects requested by users.

## Metadata

The last element of technology in order to create learning objects, metadata is not a technology in itself, but an information property of files and objects that aids in their correct classification and cataloguing. Metadata can be defined as "data used to describe other data. It can be used to describe information such as file type, format, author, user rights etc. and is usually attached to files but invisible to the user" (Europe4DRM, 2005). These data attributes make it easier for Learning Management Systems to look for the elements that it needs in a repository that may contain large

amounts of learning objects due to the fact that the search engines on most LMSs (and LMOSs) make use of metadata once a query is submitted or the system is looking for a file.

After understanding the types of learning objects that can be produced according to the needs of educators and institutions and understanding the different technologies available for developers and educators in order to produce an effective learning management system to deliver those learning objects, the economic aspects of the production of such objects and systems is an important aspect of project planning. The cost modeling of learning objects is important to assess the overall amount of economic resources that will be allocated in order to produce an effective product that, at the same time, fits into the budget of the responsible individual or institution.

Currently, the Working Group 12: Learning Object Metadata of the Institute of Electrical and Electronics Engineers (IEEE, 2002) has produced the most comprehensive and accurate standards to define the minimum set of metadata that learning objects must contain within themselves. Initially, this standard "specifies the syntax and semantics of Learning Object Metadata, defined as the attributes required to fully/adequately describe a Learning Object", setting the foundation for the elements to be considered by the creators of a specific learning object in order to accurately describe its contents. According to this standard, some of the metadata fields that must be contained in a learning object are: Type of object, author, owner, terms of distribution and format.

If the case applies to a specific learning object, metadata can also contain the following "pedagogical attributes: Teaching or interaction style, grade level, mastery level and prerequisites. The standard also defines that these attributes of metadata must accommodate the possibility of them being extended or reduced as needed and their nature can be required to be mandatory or optional. The standard also provides security and privacy information among other optional fields.

## Learning Objects Software Cost Modeling and Analysis

A software cost model is a tool used to assess the amount of resources needed to build a software system. These models work by integrating a series of estimations represented by rough numerical values that equal or near equal to real-life true values. These tools are used by developers to calculate many important factors, like work schedule, implementation deadlines and predictions towards final product value and support schemes. Depending on the estimation, the information given and processed will aid towards the determination of many other factors, like project feasibility and overall planning. Cost models also aid in the allocation of resources throughout the span of the project.

Most cost models rely on information from past projects. Whether it is a model of general use, like the Constructive Cost model (COCOMO) or the Function Points model, historical data is used to make precise comparisons based on effort, resources and total project cost in order to provide a developer with a template that is applied to a list of requirements. Once the template is integrated with the data from the client's requirements, the result provides a numerical value that expresses the desired projections in terms of the final value of a project.

Currently, there are several preconceived software development cost models that serve as a good stepping stone to produce an initial assessment towards an effective determination of the final price of a software development project. These techniques are based on mathematical models that represent the cost elements and their interactions between themselves by tying them to rather complex formulas. The outcome of those calculations represents many aspects that need to be taken into account to determine the final value of the development project. Size is a primary cost factor in most models. There are two common ways to measure software size: lines of code and function points. Values like total effort (expressed in dollars per hour), function points and thousand of lines of code provide quantifiable amounts that accurately represent the elements needed to take into account in a software development project.

Peters (1999) lists four basic steps in software project estimation:

1. Estimation of size of the software system expressed in Lines of Code (LOCs) or Function Points (FCs).
2. Effort estimation expressed in hours per person or months per person.
3. Project's schedule estimation expressed in months
4. Project's cost estimation expressed in dollars (or local currency)

## Size Determination

Estimating the size of the software system constitutes the first step towards the construction of a model. This process begins by reviewing the project's basic requirements, since a customer must always provide an initial list of requirements in order to understand what is needed to do. This process takes up a moderate amount of time, since the developer must have as much information regarding what needs to be done and if special requirements (functional or technological) need to be fulfilled in order to consider the model useful. Sometimes, a developer might encounter difficulties when trying to assess the initial requirements or estimates, but this phase can be fulfilled by a simple face-to-face encounter with the client, where the latter could only give a broad idea of what the product is supposed to do and what kind of requirements it must fulfill. Subsequent information from the client is useful throughout this part, since that enables the developer to perform adjustments and re-estimate the size of the software system.

In order to assess the size of a software project, a developer can produce a precise estimation of this value by comparing the requirements list to historical data or by applying a template of algorithms to the requirements list. If the developer uses the first option, he/she must downsize or escalate the size of the current project in direct relation to a previously produced software product. Once this process has been repeated several times by comparing it to previous projects, the result is a final value that resembles a similar project. The difference lies in the fact that the new estimation has already integrated specific values that are particular to that specific solution and that were not estimated in the past.

By choosing the second option, a developer must assess the number of features or functions that the software system will need to have and then, an algorithmic approach will be applied to it in order to understand the weight of every one of those required features or characteristics. This option is very useful when it is used by a more experienced developer, due to the fact that those types of developers are very specific when it comes to functionality pricing and know how much a component or function really cost. It is also very useful when the software system tends to adopt a "tree form", in which several subsystems or sub-functions are needed to be integrated to a main system, similar to modular systems like SAP.

## Effort Determination

Effort estimation uses the data from size estimation in order to come up with very important elements that will shape the software development process. Effort will be measured by man-hours/dollars spread out throughout the course of a schedule set by the project leader. This part of the process is marked by the determination of schedules and the use of the Software Development Lifecycle (SDLC). The SDLC acts as a signaling mechanism that announces the developer when a phase must start and end. It also serves as a tool to assess the amount of time that will be needed for each and every type of development activity: design, coding, and initial testing. This part highlights a very important fact about software development: not everything is coding. In order for it to function properly, a software system must undergo several phases and coding is only one of many elements.

The effort estimation phase uses a similar approach from the size estimation phase. It goes back in time to do a comparison with information from past projects in order to come up with a realistic approach represented by a finite amount, which can be either an up scaled or downsized version of a past project. The basic premise to calculate effort consists in determining the value for each man-hour of the project, whether it is a design, coding or testing effort. Each category must be weighed separately, as it is a logical that the same effort will not be put into an hour of those three categories. Once obtained, the effort value will reflect the monetary value that each man-hour represents, in relation to the type of effort done.

Once again, a developer can determine values by using two different approaches towards the effort. The first approach consists of going back to historical data in order to find a similar project to the one being analyzed. Once found, the historical data will be compared to the current project in terms of the complexity of each effort function (design, coding and testing) in order to come up with a value that represents the reality in terms of the current software project. This approach also assumes that the tools, schedule and development lifecycles are similar in both the past and current software projects.

The second option is more suitable when working towards a very specific project that doesn't allow for comparison with historic data due to either lack of the latter or because of the high level of complexity of the project to be developed. This approach involves the developer to estimate the total size of the software system (using either method for estimation of size) and then applying an algorithmic method that enables a direct conversion of size to effort. As a reference, the original COCOMO model, the updated COCOMO II model and the Putnam Methodology are three cost models whose accuracy can be relied upon, since they have been modeled using data collected from thousands of software development projects and its effectiveness has been proven in the field with a high level of success.

## *Schedule Determination*

The determination of the schedule in relation to the software project depends entirely on the determination of size and effort. Once these two elements have been determined, the next logical step is to distribute the workload across the allotted period of time that the development operation will take in order to determine work distribution between team members.

For an easier understanding, the schedule estimation phase takes the form of a technique used by developers named the Work Breakdown Structure. According to Olson (2001, pp. 119-121), "the Work Breakdown Structure is a top-down hierarchical chart of tasks required to complete the project. The WBS is hierarchical in that different levels of detail can be described. The overall project consists of a set of major activities or major project sub-elements". The WBS is also a good way to control the flow of tasks across the lifespan of a project, since many of those tasks can be set up in a sequential matter, chaining them together. This condition can be worked out by effectively mapping out the correlation between tasks and assigning team members to carry them out in an orderly fashion.

The addition of elements like a responsibility matrix and an organization chart make it easier to identify boundaries like task responsibility and areas of action.

## *Estimating Cost*

Once the size, effort and schedule estimation phases have been determined, the last element that needs to be determined is the total cost. Cost is represented by a numerical monetary value that expresses the total amount of resources put into the software development project. In accordance to traditional cost accounting practices, the three elements of cost are:

- Labor: Generally known as salaries, this group includes all the resources allocated into paying for professional services by laborers. In this particular case, every person directly involved in the development process. This can be considered the core of the cost groups, since most of the cost comes from labor. The determination of the cost group is made by multiplying the wages paid for each hour of work by the total number of hours worked.
- Materials: All matter used into manufacturing a product.
- Indirect Production Costs: Every other element that, because of its nature, cannot be fitted into the previous two cost groups.

# Conclusions

The creation of Learning Objects is an activity that involves many intricacies in regards of the effectiveness they must possess in order to serve as a usable tool to aid and/or accelerate the process of learning in students. Initially, the first barrier encountered in the process of introducing learning objects in the classroom consists on learning objects being dynamic enough to adapt to the internal learning processes of every student in a class. This is a problem that will be tough to solve, mainly because of the extremely difficult way to determine how the learning process takes place in humans. Even though this is an important issue, it does not lessen the effect that learning objects can have in improving and/or accelerating the learning process once they are applied in an educational setting.

Learning objects by themselves are effective but, in order to obtain the best results, learning objects must be used in conjunction with alternate technologies that allow for the use of multiple learning objects, to reuse them, to observe user behavior in order to select the most suitable objects for every student, to establish a way to store, arrange, classify and extract learning objects and to display them in a proper manner so the user can manipulate them. This is achieved by the use of repositories and Learning Management Systems. As stated by Atif et al. (2003), the maximum level of adaptation to a user's behavior and learning needs is achieved by the interaction between learning objects, a learning management system and a repository combined with tracking a student's behavior in order to select the objects that are more suitable for his/her level of advancement in the learning process.

Cost determination is important in order for developers and educators to make initial budget considerations that help them shape the cost of a learning objects development project. Depending on the type of project, it is crucial to consider the amount of economic, human and intellectual resources in order to fine tune its development phase and to identify those parts of the budget where resource allocation is scarce and had to increase. This will ultimately lead to better resource allocation and to achieve the objectives of the project.

# References

Allert, H., Richter, C., & Nejdl, W (2004). Lifelong learning and second-order learning objects. *British Journal of Educational Technology*, *35*(6), 701-715.

Europe4DRM. (n.d.). Retrieved 7 – 15 - 2005 from: http://www.europe4drm.com/l_menue/glossary/glossary.htm

Garcia, F. J. & Garcia, J. (2005). Educational hypermedia resources facilitator. *Computers & Education 44*(3), 301-325.

IEEE-LTSC (2002). WG12, Working Group Information. Announcements and News, Position Statement on 484.12.1-2002 Learning Object Metadata (LOM) Standard Maintenance/Revision. 10 December.

Lam, H. C.; Ki, W. W.; Chung, A. L. S.; Ko, P. Y.; Lai, A. C. Y.; Lai, S. M. S.; Chou, P. W. Y.; & Lau, E. C. C. (2004). Designing learning objects that afford learners the experience of important variations in Chinese characters. *Journal of Computer Assisted Learning, 20*(2), 114-123.

Olson, D.L. (2001). *Introduction to information systems project management*. McGraw-Hill.

Peters, K. (1000). Software project estimations. Course Notes, Software Productivity Center. Simon Fraser University.

Wiley, D. A. (2000). Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. In D. A. Wiley (Ed.), *The instructional use of learning objects*. Retrieved 2 - 17 - 2005 from: http://reusability.org/read/chapters/wiley.doc

Xuan, W., Li Z., & Fang, Y. (2004). An implementation of learning objects management system. *Advances in Web-Based Learning – Icwl, 3143*, 393-399

## Biography

**Carlos Francisco Lerma** is an operating systems service engineer for the General Directorate of Technological Innovation at the Universidad Autónoma de Tamaulipas in Ciudad Victoria, México. He holds a Bachelor's Degree in Public Accounting from Universidad Autónoma de Tamaulipas and a Masters' Degree in Telecommunications and Network Management from Syracuse University in Syracuse, New York, USA.