CORE
Metadata, citation and similar papers at core.ac.uk
Provided by Qatar University Institutional Repository

Developing Engineering Learning Objects Online Portal with LabVIEW and an Open Source Web Content Management System

Abstract

Learning objects (LOs) are independent chunks of knowledge normally used for instructional or learning purposes. LOs are normally reusable in the sense that they can be adopted and adapted for various learning and instructional scenarios. They are also tagged with metadata which includes descriptive information allowing them to be used and searched easily. LOs are sometimes metaphored as being a LEGO. Examples of LOs could contain multimedia content, instructional content, learning objectives, instructional software and software tools, and computer simulations. Many LOs are designed to be mediated online.
In engineering education, computer simulations based learning objects could be the most beneficial for conveying hard engineering concepts for the engineering science learner. Computer simulations have been reported to facilitate conceptual understanding and leaving positive impact on students learning in numerous number of engineering education research articles. In the last two decades, many software packages have been developed for enhancing the engineering design and analysis process, examples are Matlab/Simulink, PSpice, LabVIEW, etc. These has been used consequently by academics for enhancing their students learning.
LabVIEW is one of the most versatile computer software packages. It is used comprehensively in the industry as well as in academia. LabVIEW started as computer software interface of PC based data question equipments, however, it has grown much beyond that offering comprehensive toolkits and already implemented functions. Also it has great connectivity facilities with Matlab/Simulink, C++, and Visual Basic allowing communicating already developed codes in the latter with its core engine. The other important specification of LabVIEW is its embedded internet tools enabling publishing its programmed GUIs on the world wide web in easy and handy way.
Web content management systems is the third generation of web publishing applications after HTML and web authoring software packages such as FrontPage and Dreamweaver. It is used to manage and control a large, dynamic collection of Web material (HTML documents and their associated images). A WCMS facilitates content creation, content control, editing, and many essential Web maintenance functions. In contrast with the web development tools such as HTML, FrontPage, Dreamweaver, etc., a CMS enables faster development, cost effectiveness, and online flexibility. The basic idea of any web content management system is that a non-technical person often needs to be able to keep their own website up-to-date without having to call on a web developer to make changes every time. Of course there are some things that can only be done by a web developer, but for simpler tasks such as changing the wording of a paragraph, it is an unnecessary burden and expense for both parties if you have to get a developer to make the changes.

This paper provides an A to Z prescription of implementing a standardized Learning Objects online portal. This describing in detail a LabVIEW based Learning Object architecture, using a proper IEEE LOM metadata generation tool, and finally how on the top of that a Joomla web content management system can be used for developing the online portal.

Introduction

A learning object (LO) is a resource, usually digital and web-based, that can be used and re-used to support learning. The IEEE Learning Standards Committee defines learning objects as any entity, digital or non-digital, that can be reused or referenced during technology supported learning (IEEE, 2002). In Ip et al.(2002), learning objects are defined as "a computer mediated or delivered module or unit, that stands by itself, that provides a meaningful learning experience in a planned learning context", hence emphasizing the digital nature of learning objects. LOs are characterized with many attributes, mainly they are featured of (MERLOT, 2009):
Micro elements: LOs are micro learning elements, small chunk of information that may take couple of minutes instead the couple of hours approach of learning.
Encapsulation: LOs are self-contained and can be taken independently; Reusable: LOs are reusable, and they can be mutated to another courses or learning activities easily.
Aggregation: an LO can be grouped with other LO's or into larger learning content, course, etc..
Metadata Description: every learning object has descriptive information allowing it to be easily found by search engines The advantage of this micro design of learning materials in terms of small chunks is the usability and transferability to another related courses or learning activities with minimal modification effort.

LOs will typically have a number of different components, which range from descriptive data to multimedia and information about copy rights and educational level. At their core, however, there will be an instructional content, and probably assessment tools. The idea of learning objects could have been borrowed from the notion of object oriented programming in software engineering where many objects are utilized, linked, and operated together to perform larger macro operation targeted by the final software product. Sometimes, learning objects are described similar to the LEGO (Wiley, 2001). Learning object deployment in the teaching and learning process has been found to bring added positive value to the learning process in many pedagogical studies (Krauss and Ally, 2005; Cook et al., 2007; Jones and Boyle, 2007).
In the recent years, research and funding for LOs have considerably increased. Many services and databases have arisen, such as the specialized LOs journal "Interdisciplinary Journal of E-Learning and Learning Objects", the web data base of LOs MERLOT, and the Reusable Learning Objects Centre of Excellence in Teaching and Learning in the UK (Rlo-CETL, 2009), which has received £2.5 million initiative fund, spread over five years for the further development and research the LOs. A learning object is not just a piece of text or a graphic or a video clip, these elements can be used in the process of developing a multimedia LO. Also, the LO is not an entire course on a particular topic. Since most LOs are designed to be delivered online, they have particular benefits of rapid and rich prototyping of distance learning courses as well as blended learning courses. A key issue of LOs development is the use of metadata which provides a standardized description of learning objects enabling finding the needed content when a macro learning component is meant to be built through a set of learning objects. The latter are normally stored in online repositories, metadata schemas enable exchanging learning objects metadata among repositories that admit similar schemas for representing their metadata (Najjar, 2008).

Metadata is a crucial part in the digital resources lifecycle (Polfreman and Rajbhandari, 2008). Metadata can be described as being data about data, for instance an indexing information of a book in a library is a metadata about the book data. For learning objects, this metadata is usually encoded in an Extensible Mark-up Language (XML) file. Having the metadata structure, one can build a course on process control systems for instance by combining related learning objects from connected repositories. There are many metadata standards that have been made available for indexing learning objects such as the Dublin Core Metadata Element Set (DCMES) (Dublin Core, 2003); the IEEE LTSC Learning Object Metadata (LOM) (IEEE, 2002 & 2005); and the Alliance of Remote Instructional Authoring and Distribution Network for Europe (ADRIANE) (Duval et al., 2001). The IEEE LOM is a widely spread standard (Najjar, 2008). In the next section, further details about the LOM standard are introduced.

Building Metadata Profile

Building a metadata profile that conforms to the IEEE LOM is exhaustive and a time consuming process (Najjar, 2008; Polfreman and Rajbhandari, 2008). This has been reported to be a restriction factor of metadata generation by the learning objects authors, limiting the LO searchability and outreach. Hence a couple of automatic metadata profile generation have been developed. Automatic generation tools was reported to be the cure of the metadata generation bottleneck (Polfreman and Rajbhandari, 2008). One of these tools that developed for the IEEE LOM standard is the LomPad (LomPad, 2009). LomPad is an open source java based tool, it is bilingual (English/French) and enables the user an easy way of deploying LOM based metadata

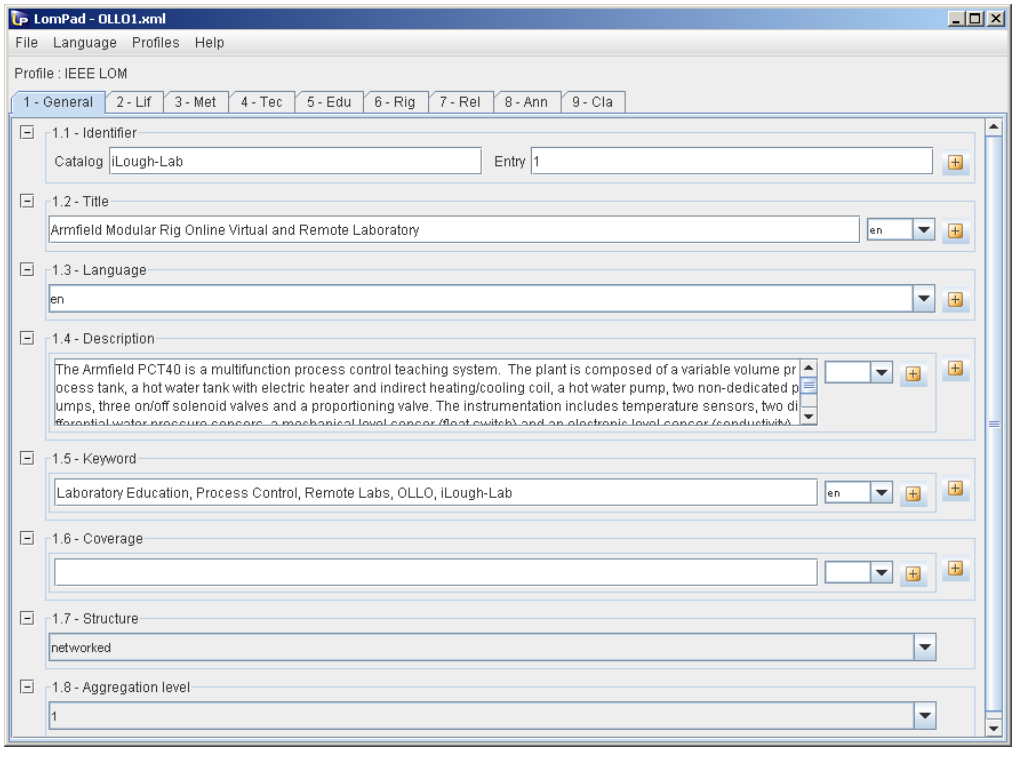

Figure 1. The LomPad automatic IEEE LOM metadata generation tool

in their learning objects. The tool interface is shown in Figure 1. There are nine main pages, each of them is dedicated to one of the main LOM categories. After filling the data, the user can view the metadata file either in HTML or in XML format. After saving the metadata in XML, it can be deployed on the web in the specific webpage related to the learning object. Figure 2 shows a sample code of a metadata file generated with the LomPad tool. The file describes a learning object of virtual and remote laboratory developed at the chemical engineering department of Loughborough University (Abdulwahed and Nagy, 2009; iLough-Lab, 2009).

```
<?xml version="1.0" encoding="UTF-8" ?>
<lom xmlns="http://ltsc.ieee.org/xsd/LOM"
xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"
xsi:schemaLocation="http://ltsc.ieee.org/xsd/LOM
http://ltsc.ieee.org/xsd/lomv1.0/lom.xsd">
 <general>
- <identifier>
<catalog>iLough-Lab</catalog>
<entry>1</entry>
</identifier>
 <title>
<string language="en">Armfield Modular Rig Online Virtual and
Remote Laboratory</string>
</title>
<language>en</language>
```

Figure 2. A sample code of the Armfield rig OLLO metadata file conforming with the IEEE LOM standard.

LabVIEW as a Platform for Building Learning Objects

Many software platforms have been used for developing LOs such as Flash, Photoshop, and web design tools, however, building LOs that contains simulations of science and engineering concepts can be tough, since mathematical and programming algorithms should be built from scratch. LabVIEW provides a flexible and rich environment for developing the STEM LOs due to its diverse features. LabVIEW, a short of Laboratory Virtual Instrument Engineering Workbench, is defined as " a graphical programming language that has been widely adopted through industry, academia, and research labs as a standard for data acquisition and instrument control software" (Travis and King, 2007). LabVIEW has been extensively used in academia, it can be considered an ideal platform for implementing STEM LOs due to the following factors:

- LabVIEW is inherently a Virtual Instrumentation (VI) development environment. Its programmes codes and files are called VIs. The virtual instrumentation concept facilitates building effective and compact interfaces for illustrating STEM concepts (both functionally and from user interface perspective).
- LabVIEW offers a comprehensive library for measurements, filtering, and data analysis.

- LabVIEW offers the possibility of implementing simulations for a wide spectrum of engineering and science problems. LabVIEW toolkits offer a very comprehensive set of already implemented functions.
- LabVIEW has embedded internet connectivity tools that enables rapid web deployment of LabVIEW Virtual Instruments (VIs).
- LabVIEW has many connectivity tools that enables importing another engineering files, i.e. Matlab/Simulink codes, or Solidswork 3D models. This is particularly an enabler of using legacy code.
- LabVIEW has also connectivity tools with other applications such MS Office, or SQL database,
- LabVIEW codes are called "G-code" since LabVIEW is a graphical based programming. The G-code is more appealing for engineers than text based programming. With G-, programming is done through drag and drop graphical functions instead of line text.
- LabVIEW offers a software application development kit that enables creating stand alone executable applications from LabVIEW VIs, similar to other programming languages such as Visual C++ or Visual Basic. Hence, stand-alone executable LOs can be implemented.
- In LabVIEW, interacting with data is done through a professional Graphical User Interface (GUI) that would include controls, graphs, and 3D visualization tools, which are selected from a comprehensive library set. These can be easily customized.
- LabVIEW offers many embedded solutions for internet publishing.
- LabVIEW offers multimedia connectivity where audio/video files can be embedded in the developed application.
- LabVIEW has an active collaborative community and lots or reusable legacy codes.

These features and others makes LabVIEW a favored platform for implementing STEM LOs.

LabVIEW Programming Environment

LabVIEW developing environment consists of three main parts, the front panel, the block Diagram, and the icon/connector. The front panel is the interactive GUI of a VI, named so as it is similar to a physical instrument front panel. On the front panels, controls (user inputs variables) are placed as well as indicators (the LabVIEW programme outputs). The controls generally looks similar to physical buttons, switches, and knobs that can be found on a conventional instrument. Controls supplies the block diagram with a needed data for processing. The indicators get back the generated data by the LabVIEW programme and display them for the user.
The block diagram is the place where the actual VI's source code (written in G-) is contained, hence the block diagram is similar to the text lines found in conventional programming languages such as Java or C++. The components of the block diagram (the G-code) are lower level VIs, built-in functions, constants, and programme execution control structures such as "For" and "While" loops. When the programmer places a control or indicator on the front panel, LabVIEW will automatically create a relevant "terminal" on the block diagram. . .
Figure 1 shows a LabVIEW VI' Front Panel (left) and Block Diagram (Right). This VI is designed to simulate Lorenz system, which is a nonlinear system described by the following three equations:

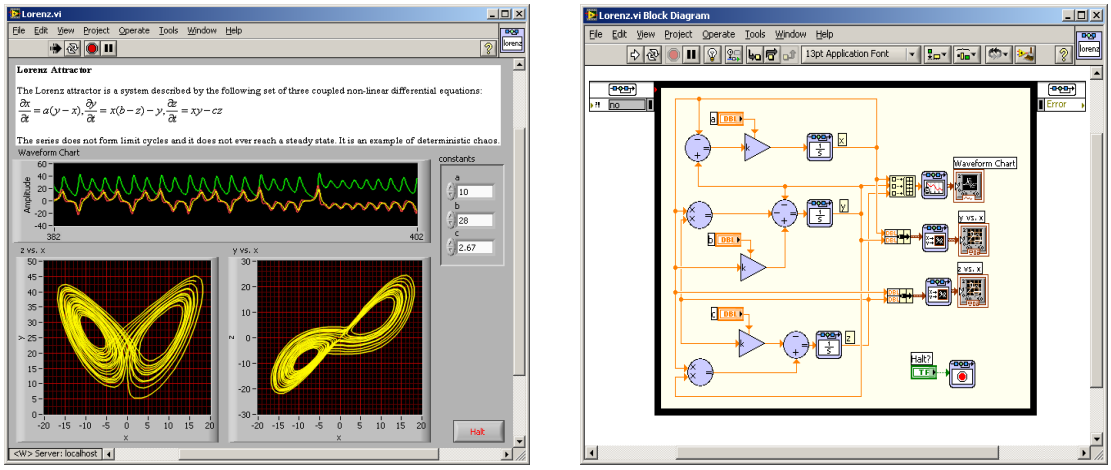

Figure 1. Simulation VI of Lorenz system, left is the front panel, while right is the block diagram.

$$\frac{dx}{dt} = a(x-y), \frac{dy}{dt} = x(b-z)-y, \frac{dz}{dt} = xy-cz$$

Lorenz system is normally used in nonlinear control systems text books to explain limit cycles. As shown in Figure 1, the front panel passes the user inputs (a, b, c) through the specified controls to the block diagram, and returns the programme outcomes from the block diagram to the front panel presenting them in a graphical simulation format via the relevant indicators.

A subroutine in LabVIEW is constructed through creating a Sub-VI, which is a VI with icon and connectors specifying its inputs and outputs. Using low level sub VIs and calling them at a higher level is very useful when programming complex LabVIEW applications.

As most programming languages, LabVIEW use For and While loops to control repetitive operations in a VI. The LabVIEW For and While loops are resizable boxes in which the sub routine that the programmer needs to be repeated is placed in. The For loop executes for a specific number of times. The While loop includes a conditional terminal that can terminate the loop execution. Transferring a variable value from one iteration to the next one of a For or While loop in LabVIEW is achieved through what is called a shift register.

The conventional programming command “if-then-else” is achieved in LabVIEW through a case structure. Case structures have multiple sub diagrams, but only one is executed at once. The selection of this sub diagram depends on the condition status. LabVIEW also contains dialog functions which bring up a dialog box containing a message to the user. The While loop and the Case structure are found almost in every LabVIEW programme.

Procedural programming languages like C or Java have inherent control flow because statements execute in the order they are written in the programme, but LabVIEW G-Code executes at once. LabVIEW can determine the sequence of programme execution (control flow) by using Sequence Structure, which is an ordered set of frames that execute sequentially.

LabVIEW contains a couple of timing functions such as Wait(ms), Tick Count(ms), Wait Until Next ms Multiple, Time Delay, and Elapsed Time to help to measure time, synchronize tasks, and allow enough idle processor time so that loops in a VI don’t need time to execute that is out of the processor maximum capacity.

LabVIEW like other programming languages can process scalar variables and arrays of variables. A LabVIEW array is a collection of elements that are all of the same nature. There are many built-in functions in LabVIEW for manipulating arrays such as Initialize Array, Array Size, Build Array, Array Subset, Delete From Array, etc. For grouping elements of different nature, LabVIEW uses the data type cluster which is analogue to a struct in C or the data member of a class in C# or Java. A LabVIEW cluster can be thought as a bundled different wires into one cable. Clusters are particularly useful for reducing the number of wires or terminals associated with a VI. Many of LabVIEW functions are polymorphic, i.e. can adjust to inputs of different-sized data.
Visual representation of data in LabVIEW can be achieved by using Chart and Graph. A chart appends new data to old data and plotting one point (or set of points) at a time (added to the previous points) hence showing the data change in dynamical way. While a graph, shows the full set of data after it has been generated. A waveform is a LabVIEW useful function for plotting analogue signals against time. LabVIEW also offer the possibility of 3D plots, but only available on windows machines.
LabVIEW includes several functions for manipulating strings such as obtaining the string length, merging two strings together, converting a string to a number, etc. With the LabVIEW functions, Write to Spreadsheet File and Read From Spreadsheet File, a programmer is able to write and read data from a disk file. For dealing with dynamic data, the functions Write to Measurement File and Read From Measurement File are useful.
Similar to C , Java, or Pascal, LabVIEW holds Local, Global, and Shared Variables. The local variable enables accessing a front panel object from different places in the block diagram of a VI reducing the wiring and the graphical complexity of the block diagram. A global variable enables accessing values of any data type among many VIs that are running at the same time while local variables are limited to the same VI only. A shared variable is similar to a global variable but works across multiple local and networked applications. Every indicator and control in LabVIEW has a set of base properties such as color, visibility, etc. These properties can be programmatically manipulated with the LabVIEW function, Property Node.

LabVIEW Internet Connectivity

The LabVIEW offers many ways of establishing internet connectivity. There are three main methods, a) through the internet connectivity toolkit, b) through the built-in LabVIEW web server, web publishing, and remote panels tools, and c) through the LabVIEW web services feature that were included from LabVIEW 8.6 and forward.
The internet connectivity tool is the oldest support of web connectivity provided by LabVIEW during the late 1990s. Then the built-in web server, web publishing, and remote panels features were introduced later on to make the internet publishing easier and more handy, yet the newer versions of LabVIEW continued to include the internet connectivity toolkit for use if needed with legacy code. However, developing web applications has been always recommended with new features instead of the internet toolkit (NI, 2009). With the LabVIEW built in web server, the user can publish a VI through the LabVIEW web publishing tool in three ways:

- Snapshot: which is a static image of the VI's front panel.
- Monitor: This is an image of a VI's front panel that is refreshed every N seconds. The user can configure the N parameter.

- Embedded: In this option, a controllable version of the VI front panel is published as embedded plug-in at the client browser. Hence, the VI is displayed in real time at the client side and the client have control over it.

The web services features were developed to give the LabVIEW internet application programmer more flexibility in communicating between LabVIEW VIs and other web technologies such as Java or Flash, e.g. for developing Java based applications that can communicate with LabVIEW VIs running on a LabVIEW web server. The remote labs developed in this work has be achieved through the LabVIEW built-in web server and web publishing tools.

Architecture of a LabVIEW Based Learning Object

The basic structure of a LabVIEW based LO will consist of many components as shown in Figure 2. These could be:

- Model: The mathematical model of a process, physical phenmona, or engineering structure to be simulated for further analysis and understanding.
- Learning Content: This is the content to be learnt in association with the LO, this may include Microsoft docs, images, spreadsheets, etc.
- Legacy Code: This is external code written in Matlab for instance that is useful to be incorporated in implementing a desired LO.
- Multimedia Files: This may include video and audio files.
- Software Engine: The coordinating G-Code of all the previous constituting components of the LO. LabVIEW engine.
- LabVIEW Web Server: The LabVIEW web server publishes the developed LO online.
- Metadata: The metadata file describing the LO to be included in the HTML webpage that embeds the LO after publishing through the LabVIEW webserver.

An efficient way of implementing equations intensive mathematical models is to use one of the many mathematical script nodes such as the “MathScrip Node”, the “Matlab Script”, or the “Formula Node”. The LabVIEW MathScript syntax is similar to the Matlab language syntax. The “Matlab Script” node on the other hand can be used for utilizing an already implemented Matlab code within the LabVIEW environment. The node calls Matlab to execute the scripts, hence, a copy of the Matlab software version should be installed on the computer because the script nodes invoke the MATLAB software script server to execute scripts written in the Matlab language syntax. For simpler mathematical models, the block diagram programming (G-Code) can be used instead of a mathematical node. Communication between LabVIEW and an external software application such Microsoft Office or Windows Media Player can be done through the LabVIEW ActiveX Server, hence the possibility of integrating learning contents or multimedia files into the LabVIEW based LO is facilitated.
The software engine is the LabVIEW code that analyzes the data, manipulates it, and co-ordinates the data communication between the user and the LO components. The engine will be responsible for directing the data to the appropriate visualizing components on the front panel and for processing the user input from the latter. Simulation results can be elegantly visualized in the LabVIEW front panel by using the extensive LabVIEW virtual instruments libraries.

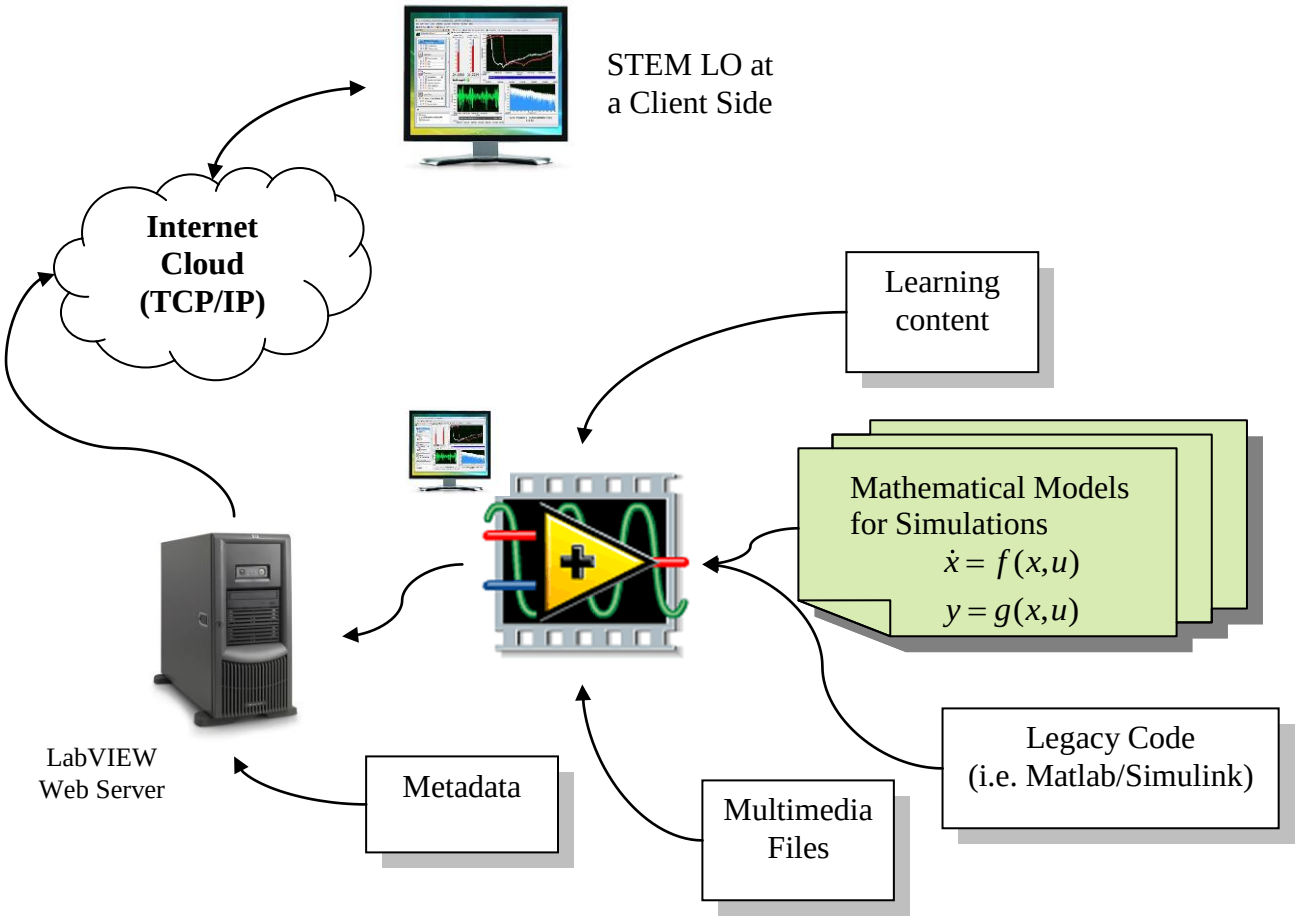

Figure 2. Architecture of a LabVIEW based Learning Object

Once the LabVIEW based LO code is finalized, it can be converted into a stand alone executable LO by using the LabVIEW application builder toolkit. This executable software can then be installed and run on any PC. The stand alone software installer will install the LabVIEW runtime engine, which will enable running the executable software without the need of the LabVIEW development platform to be installed on the client PC. Remotely accessing the LO through the internet can be achieved via the LabVIEW web server. The LO will be embedded into an HTML file, a metadata describing the LO can be added to that file. To top up such HTML files of LO, it is important to envelop them with online portal. In the next section, detailed description of developing online portals with a content management system.

The Joomla Web Content Management System

A content management system is defined as " a process of collecting, organizing, categorizing, and structuring informational resources of any type and format so that they can be saved, retrieved, published, updated, and repurposed or reused in anyway desirable" (Yu, 2005). A web content management system (WCMS or Web CMS) is a content management system (CMS) software, usually implemented as a Web application, for creating and managing HTML content.

It is used to manage and control a large, dynamic collection of Web material (HTML documents and their associated images). A WCMS facilitates content creation, content control, editing, and many essential Web maintenance functions (Wikipedia, 2009).
In contrast with the web development tools such as HTML, FrontPage, Dreamweaver, etc., a CMS enables faster development, cost effectiveness, and online flexibility (McKeever, 2003; Yu, 2005; Souer et al., 2008). Souer points out that WCM software acts both as a controlling mechanism and as an enabler: it controls the processes of managing the Web content with workflow, scheduling, authorization, reuse of content and archiving. WCM software enables organizations to operate their own content-delivery strategy with specific user interaction, personalization, and multi-channel delivery. The life cycle of websites development with a WCMS includes two phase, the content collection, and then the content delivery of publishing (McKeever, 2003). A content such as text, graphics, video, audio, etc., is collected and stored in a repository, then the delivery phase will extract the content and publishes it for the world wide web. In non-automated environment, such as when developing with HTML or FrontPage, this is very difficult and time consuming process (McKeever, 2003).
The basic idea of any web content management system is that a non-technical person often needs to be able to keep their own website up-to-date without having to call on a web developer to make changes every time. Using WCMSs helps organizations to manage complex emerging websites with high quality (Boiko, 2001). Certain aspects can only be developed by a web developer, but for simpler tasks such as changing the wording of a paragraph, it is an unnecessary burden and expense for both parties if you have to call a developer to make the

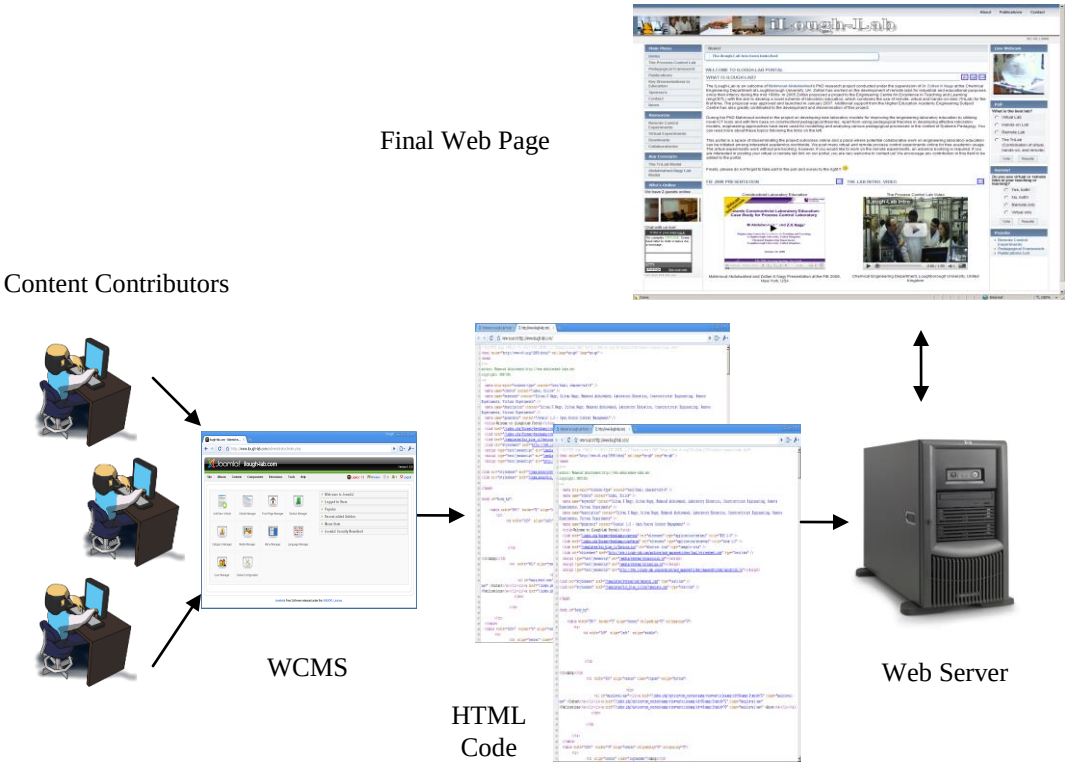

Figure 3. General model of a WCMS functionality

changes.
With a WCMS, content files can be created or collected from available sources, these then would be managed through storing them into a repository (hosting storage space for instance) which will include the CMS database that contains information about locations of the content files, the users profiles, security login data, etc. The publishing process is done through extracting the repository components and constructing a publishable online pages. General model of a WCMS functionality is show in Figure 3.
Some of the key features of WCMSs are templating, editable content, document management, and the administrator and user control. Most CMSs come with automatic templating feature. This enables creating a standard output templates in HTML that can be automatically applied to new and existing content, allowing the appearance of all that content to be changed from one central place without any change in the content itself. With a CMS, a contributor can edit a website content and then enables its publishing online. Managing the life cycle of a document from initial creation time, publication, and archive and document destruction is core part of most CMSs. The administrator and users control includes the identification of all key users and their content management roles, as well as the ability to assign roles and responsibilities to different content categories or types. Powell and Gill (2003) list a number of generic features of WCMSs, these are: streamline and automate administration, implement web-forms-based content administration, distributed structure of content management and control, separate content from layout and design, create reusable content repositories, implement central graphic design management, automate work/flow management, build sophisticated content access and security, make content administration database-driven, include structures to collect and use metadata, allow customization and integration with legacy systems, allow achieving and version control. It is argued that WCMS provides a solution for overcoming problems associated with the usage of nowadays data-intensive web applications such as consistency, navigation, content audit and control, tracking of content, and data duplication (Vidgen et al., 2001).
Currently, there are many available systems for web content management, both commercial and open source. For this project, it was aimed to adopt the open source choice to avoid any further costs, provided the fact that many of the open source CMS such as Joomla (Joomla, 2009) and Drupal (Drupal, 2009) have equivalent features with commercial systems. Both Joomla and Drupal were classified as 1st and/or 2nd winners of the “Open Source CMS Award” for the years

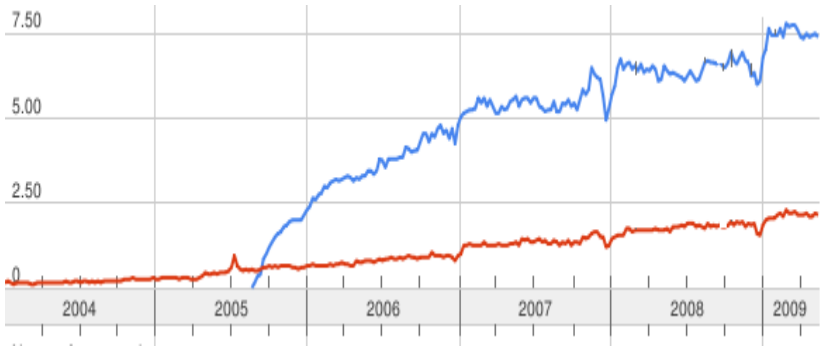

Figure 4. Joomla (Blue, 3.5) vs. Drupal (Red, 1.0) search rates from Google Trends

2006, 2007, and 2008 (Packt Publishing Awards, 2009).
The main superior feature of open source WCMS is the community of developers. A popular web content management systems has a community of many hundreds, or even thousands, developers who work on enhancing the code, releasing new versions, solving bugs, and building new features for extending the WCMS functionality.
Drupal is in particular useful for developing online websites that contain social networking features of web 2.0 applications. On the other hand, Joomla is more suitable for the portal type websites, yet Joomla can be equipped with social networking capabilities and it has been reported to be the tool of building social networking websites (Yue et al., 2009). Additionally, the learning curve of Joomla is quicker than Drupal, which might explain the popularity of Joomla over Drupal as shown in Figure 4. The presented data is taken through Google Trends and it reflects the internet search ratio of Joomla vs. Drupal (Google Trends, 2009). The data shows that the users propensity to search for Joomla is 3.5 higher than searching for Drupal. The iLough-Lab is meant to be an online portal, hence, the Joomla web content management system (WCMS) has been adopted for its development. Probably, the iLough-Lab is the first remote laboratory portal to be implemented with a content management system.

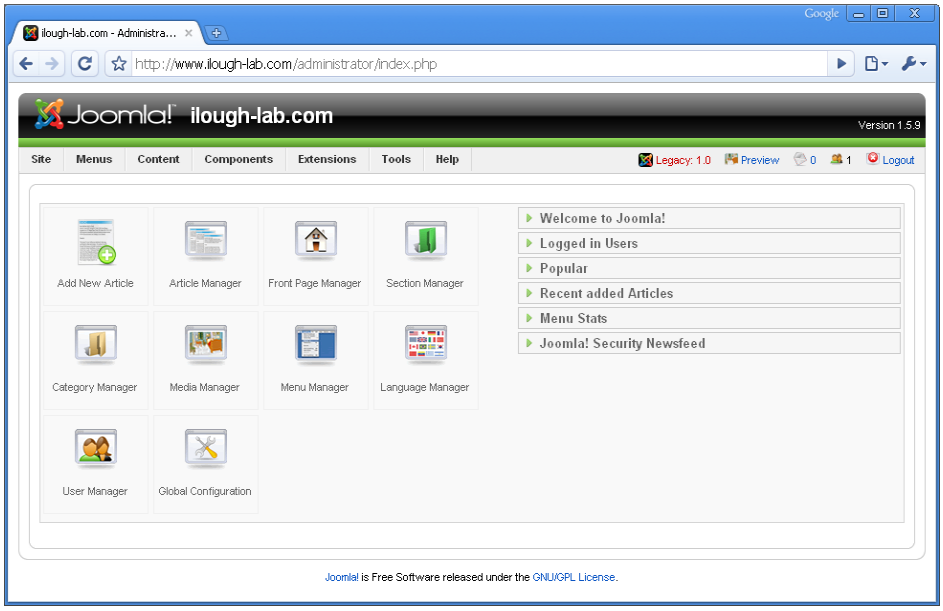

Figure 5. Joomla administrator control panel

Joomla was released first in 2005 as an open source WCMS. Joomla defines its system as “an award-winning content management system (CMS), which enables users to build Web sites and powerful online applications. Many aspects, including its ease-of-use and extensibility, have made Joomla the most popular Web site software available. Best of all, Joomla is an open source solution that is freely available to everyone” (Joomla, 2009). The name Joomla comes from the Arabic/Swahili word “Jumla” means “All together” or “As a whole”, since installing a new extension into Joomla will be integrated fully as a whole (LeBlanc, 2008).
As other WCMSs, Joomla is targeted for non-technical’s to flexibly manage complex websites. It also provides additional tools which help to add more functionality to a website such as a

guestbook, forum, events calendar, etc. Joomla has become very popular since its first version Joomla 1.0 release in 2005, having the largest share market of open source WCMSs. Joomla is not a commercial business and it relies on the open source model, there are no paid employee, the core developing team is composed of volunteers who have passion towards the system working on developing it in their free time. Joomla system comes with user friendly and easy to navigate administrator control panel for managing the content such as shown in Figure 5.

Joomla Features

Some of the main features of the Joomla WCMS are user management, content management, polls, menu manager, and web services (Joomla, 2009). The user management is done through a registration system of the websites users. These users are allocated to one of the user groups. These groups holds different permissions on access, editing, publishing, and administration. The Joomla content management is a three levels hierarchy. The root of a content is an article which is located under a category which is located under a section. The website users can view published articles (under permissions restrictions, if any) and e-mail them to a friend or convert them into a PDF files. These articles can be archived and unpublished. The Joomla provides an embedded WYSIWYG editor (acronym from What You See Is What You Get) for creating articles., With this editor, the final output to the article is the similar to what is seen during the editing process. The articles can be shown on the front page, or through links from menus and other modules. Polls are embedded in Joomla to enable a website designer a quick tool for surveying the website users opinions. With the menu manager, the website administrator can create menus, enabling a nested hierarchy if needed. The menus can be designed with different styles, e.g. the common dropdown format. The Joomla embedded web services, enable the designer to use Remote Procedure Calls (via HTTP and XML).

Joomla Architecture

Joomla docs refers to Joomla as a "Three Tiered" system with three layers such as shown in Figure 6. At the bottom is the "Framework Layer" which contains a set of libraries that is required by the framework itself or other third party extensions for communicating the systems

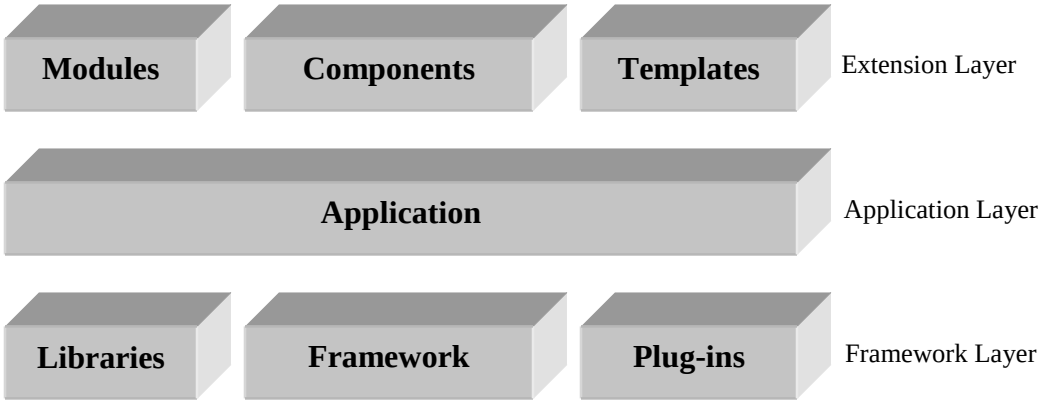

Figure 6. Joomla three tier layers.

inputs and outputs with the database, the framework layer also contains plug-ins that extends the framework functionality.

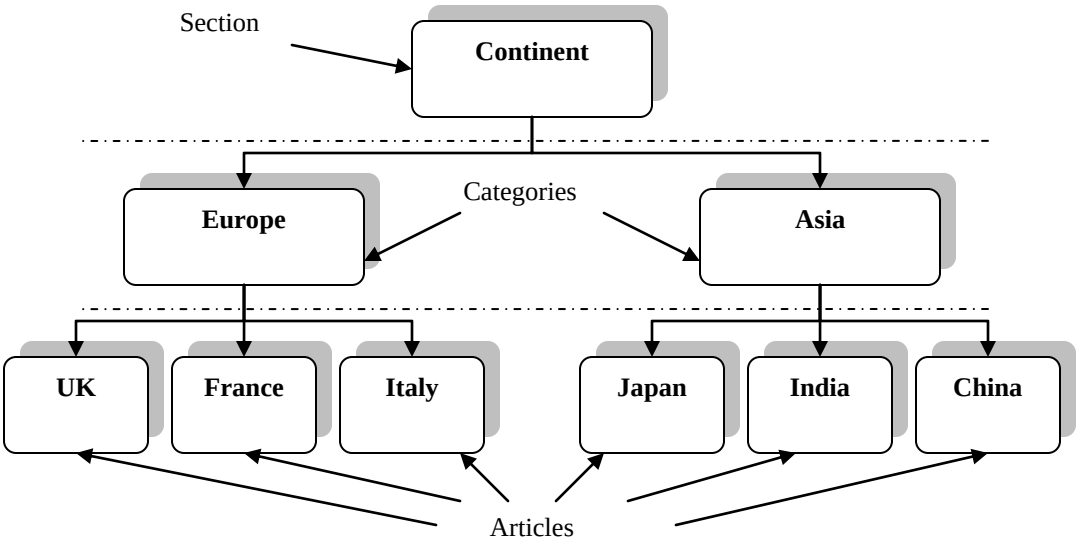

Figure 7. Section – Category – Article hierarchy example.

The middle layer is the "Application Layer" which consists of four main applications, the front-end view of the content management system (the website that the user see when requesting the website address in a web browser), the back-end administration (the administration control panel that was shown in Figure 5, the installation application that installs Joomla on a web server, and an application for supporting remote administration of Joomla websites. The upper layer is the "Extension Layer", which allows user-defined extensions to be added to the Joomla system on the form of templates, components, and modules which counts for 3175 extensions as of August, 2009 (Joomla, 2009).

The Joomla main architecture is composed of the following: components, content, content items, category, section, module, plug-in, component, and a template containing all the previous. Content is basically the text, pictures, and media of the website.

The content is normally organized in content items called articles. The articles are the lowest layer of the three level hierarchy. Any article can be divided into two sections, the introductory, and the main section. A category is the middle level of the hierarchy holding one or more content items. The top layer is a section, which holds one or more category. Figure 7 shows an example of a website having geographical information. The section continents contains two categories, Asia, and Europe, and each category contains three content items. In brief, Joomla uses sections, categories, and articles (or Content Items) as a mechanism for organising the content, similar to directory folders and files in a standard file system. The data that makes up the content is stored in a MySQL database. The Joomla Administrator application is basically a GUI for that database. Modules are the Joomla windows of the website's front end. They are placed around the website edges surrounding the main content in the centre. The website menus for instance are shown in modules. Modules in principle are the main user interface with a Joomla based website. Plug-ins

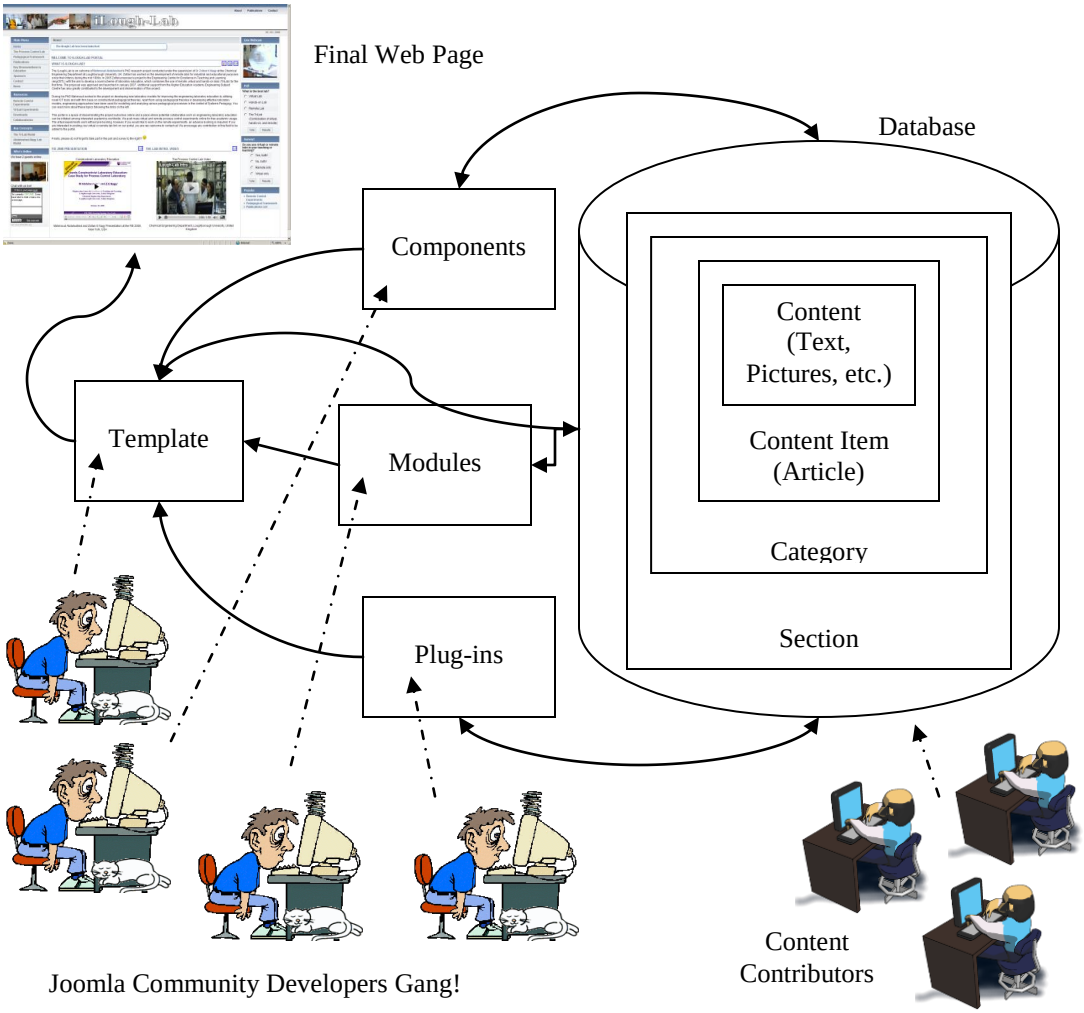

Figure 8. General architecture of the Joomla WCMS

are small programs to be run before any article is shown on the website. Plug-ins are normally provided with configuration parameters. Components are Joomla web applications that can be installed as add-ons to the Joomla features that come with the basic version. These add-ons can be then accessed and configure through the 'Components' menu in the Joomla administrator panel. Finally, the template is the vessel of displaying a Joomla website. The template consists of an index.php file that defines the layout of the site's pages, and a cascading style sheet (CSS) file to define the formatting to use. The template determines the module positions on the screen, fonts, colours, etc.

The Joomla featues and architecture provide a flexible and easy way for online publishing. However, the main power of Joomla is in its extensibility. Since Joomla is an open source system, interested developers world wide can contribute there extensions to the system. These third party extensions can be components, modules, plug-ins, or templates. Currently there are thousands of Joomla extensions that have been developed by hobbyists, dedicated professionals, and low profit commercial agents. Most of the extensions are free to use or can be obtained for a modest price. These extensions spread over wide spectrum of categories such as: access and

security, contacts and feedback, photos and images, administration, content and news, search and indexing, authoring, site management, bridges to other web applications, sports and games, calendars and events, directory and documentations, structure and navigation, e-commerce, clients, programming and edition, communication, hosting and servers, style and design, community and groupware, multimedia, etc. These extensions can be installed easily into Joomla through its installation system. Figure 8 shows a general architecture of Joomla and the relationships between its different elements, content contributors, and system programmers.
We have developed an online portal with regard to laboratory education (www.ilough-lab.com) successfully using Joomla. The site contains also links to learning objects of virtual labs that have developed with LabVIEW. The combination of both techniques have proved efficiency and rapid development with rich features.

Conclusions

This paper is intended to introduce STEM academics to an A-Z recipe of implementing rich and effective learning objects online portal using LabVIEW and Joomla. The main features of both technologies have been described. The architecture of implementing a LO with LabVIEW has been detailed. It is hoped that this paper will encourage STEM academics on developing and publishing learning objects online.

References

Abdulwahed M, Nagy Z K, 2009. Applying Kolb's Experiential Learning on Laboratory Education, Case Study. Journal of Engineering Education, Vol. 98, no 3, pp 283-294

Boiko B, 2001. Understanding Content Management. American Society for Information Science, Vol. 28, October-November, pp. 8-13.

Cook J, Wharrad HR, Morales R, Windle D, Leeder T, Boyle R, and Alton, 2007. Implementations, Change Management and Evaluation: A Case Study of the Centre for Excellence in Teaching and Learning in Reusable Learning Objects. Journal of Organizational Transformation and Social Change, 4, 1: 57-73.

Drupal, 2009. www.drupal.org

Dublin Core, 2003. Dublin Core metadata element set v1.1, 2003, Retrieved 06.08. 2009, from http://dublincore.org/documents/2003/02/04/dces/.

Duval E, 2004. Learning Technology Standardization: Making Sense of it All. Computer Science and Information Systems, Vol. 1, No. 1, February 2004, [Duval, et al., 2001a] E. Duval, et al., The ARIADNE knowledge pool system, Communications of the ACM, vol. 44, no. 5, pp. 73-78, 2001.

IEEE, 2002, IEEE LTSC Draft Standard for Learning Object Metadata.

IEEE, 2005. IEEE Standard for Learning Technology—Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata.

iLough-Lab, 2009. www.ilough-lab.com

Ip A, Young, and Morrison I, 2002. Learning objects - Whose are they?. In Proc. of 15 th Conf. of the National Advisory Committee on Computing Qualifications, pages 315–320.

Jones R and Boyle T, 2007. Learning object patterns for programming. Interdisciplinary Journal of Knowledge and Learning Objects, Volume 3, pp 19-28.

Joomla, 2009. www.joomla.org

Krauss F and Ally M, 2005. A Study of the Design and Evaluation of a Learning Object and Implications for Content Development. Interdisciplinary Journal of Knowledge and Learning Objects, Volume 1.

LeBlanc, 2008. Learning Joomla! 1.5 Extension Development. Packt Publishing, Birmingham.

LomPad, 2009. http://helios.licef.ca:8080/LomPad/en/index.htm, retrieved on 06.08.2009

McKeever, S. (2003), "Understanding web content management systems: evolution, lifecycle and market", Industrial Management and Data Systems, Vol. 103 No.9, pp.686-92

MERLOT, 2009, http://www.merlot.org/merlot/index.htm , retrieved on 06.08.2009.

Najjar J, 2008. Learning Object Metadata: An Empirical Investigation and Lessons Learned. PhD thesis, Katholieke Universiteit Leuven (K.U.Leuven), Belgium.

NI, 2009. www.ni.com
Packt Publishing Awards, 2009. http://www.packtpub.com/award

Polfreman M and Rajbhandari S, 2008. MetaTools - Investigating Metadata Generation Tools. JISC Reports- ie-repository.jisc.ac.uk, retrieved on 06.08.2009.

Rlo-CETL 2008, http://www.rlo-cetl.ac.uk/, retrieved on 03..05.2008.

J. Souer, P. Honders, J. Versendaal, and S. Brinkkemper. A framework for web content management system operation and maintenance.Journal of Digital Information Management (JDIM), pages 324--331, 2008.

Travis J and Kring J, 2007. LabVIEW for Everyone: Graphical Programming Made Easy and Fun. Prentice Hall: Upper Saddle River, NJ.

Vidgen R, Goodwin S, Barnes S. 2001. Web Content Management. In Proceedings of the 14th International Electronic Commerce Conference, Bled, Slovenia, 465–480.

Wikipedia, 2009. www,Wikipedia.org

Wiley DA, 2001. Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. The Instructional Use of Learning Objects.

H. Yu, Content and WorkflowManagement for Library Websites: Case Studies. Hershey: Information Science Publishing, 2005.

KB Yue, D De Silva, D Kim, M Aktepe, S Nagle, 2009. building real world domain specific social network websites as a capstone project. Journal of Information Systems Education, Vol.20, 1, pp. 67-76.