Authoring Environment for E-learning Production Based on Independent XML Formats

Alberto González Téllez
Dept. de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia
46022 Valencia, Spain

Abstract— **E-learning content production has faced the problem of making content independent from authoring applications and delivery platforms. Both aspects are very relevant to the need of keeping learning institutions as much independent as possible from software vendors. Content independence and interoperability is also required to easily share and move e-learning repositories. We propose a user friendly authoring environment based on the independent, open and XML compliant product Docbook; its simple, flexible and well structured markup make it easy to interoperate with other XML languages, particularly those related to e-learning. We are particularly interested in IMS Content Packaging and IMS QTI because they allow importing learning material into our Sakai based learning management system named PoliformaT. Docbook has also a powerful and fully customizable set of XSLT styles sheets that allow high quality web and paper delivery. The user front end is built on the commercial XML editor from XMLmind XXE. The major part of the environment is portable to other XML editors because it relies exclusively on Docbook components. At the moment the proposal presented is much more flexible, usable and productive than the available alternatives.**

E-learning; interoperable formats; XML; IMS; Docbook

I. INTRODUCTION

After being established as a mature technology e-learning has faced the problem of content management in such a way that content be platform and vendor independent. XML originated on the web as a set of standards that provide open and independent data representation and processing, has become the more convenient platform to define interoperable content representation. In spite that these efforts have been more oriented to content packaging (i.e. SCORM and IMS Content Packaging), nowadays there are XML compliant languages that cover the whole range of content types (text, graphics, multimedia, etc). In this way interoperability and standard data processing can be applied to the whole content hierarchy, from individual content items to a whole subject, course or set of courses. In this work we present a learning content authoring environment based on the XML compliant and open product Docbook [1]. The main interesting features of Docbook are its well structured and format independent definition, its total independence from particular software products and vendors, and its complete set of mature and well tested web and paper format generation tools. The simple and well structured Docbook markup simplifies its interoperability with other XML compliant standards, particularly those oriented to import and export e-learning content in learning management systems (LMS) [2].

Docbook has been the framework of choice in other e-learning related works like [3][4][5][6][7]. In [7] we developed our first Docbook customization (TeachdB) aimed to provide content reuse and one-source-multiple-format delivery. In this work we present an extension of this customization oriented to interoperate with IMS Global Consortium Standards [2], particularly IMS Content Packaging (CP) [8] and IMP Question and Test Interoperability (QTI) [9]. The whole publishing process is schematically depicted in figure 1.

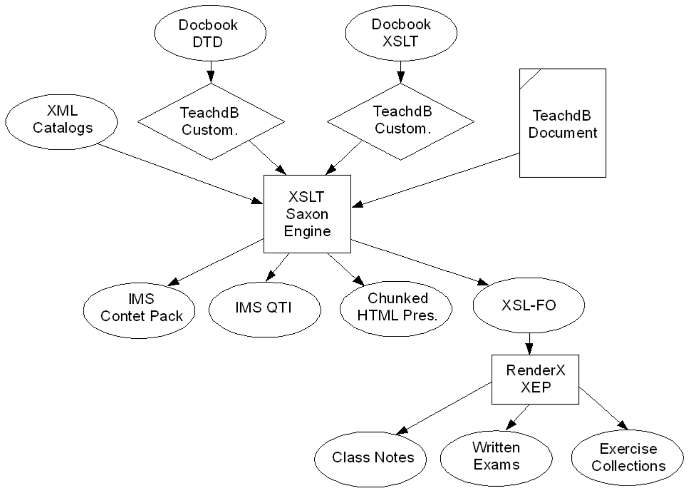


Figure 1. E-learning publishing process based on Docbook.

The ability to easily produce IMS CP and QTI is a key feature for us because our university has based its e-learning support on a Sakai [10] customization named PoliformaT [11] and Sakai uses these two IMS standards to import and export content modules and assessments, respectively. Then our main contribution is the development of a concrete friendly e-learning production tool based on open, independent and XML compliant standards and fitted to Sakai as target LMS. The approach can be applied to other environments like SCORM formats and Moodle LMS [12].

The content of the paper is organized as described next. In section 2 we briefly describe IMS CP and QTI standards. Section 3 is dedicated to summarize our Docbook customization [10] intended to fulfill our requirements. Section 4 contains the main issues involved in the implementation of IMS CP and QTI generation from our Docbook customization. Section 5 describes how authors have to use the environment. It also includes a description of real production examples and some figures about the impact that it had in our teaching experience. Section 6 is devoted to compare our proposal to other available alternatives. Finally some conclusions and future work directions are considered.

II. IMS CP AND QTI STANDARDS

IMS Global Learning Consortium began at 1997 and since then it has published about 20 standards dealing with interoperability for learning systems and learning content and the enterprise integration of these capabilities [2]. In the next two sections we briefly describe the two IMS standards that are relevant for us: IMS Content Packaging and IMS Question and Test Interoperability; both are XML compliant.

A. IMS Content Packaging

As stated in the standard "the IMS Content Packaging provides the functionality to describe and package learning materials, such as an individual course or a collection of courses, into interoperable, distributable packages" [8]; its latest version is 1.1.4.

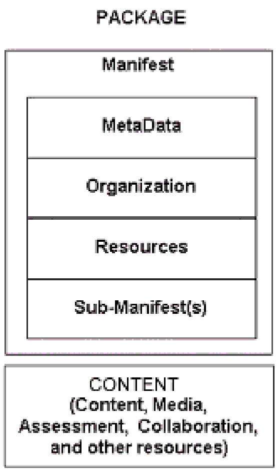


Figure 2. Content Package structure.

As shown in figure 2 a Content Package has two parts: a manifest and a content pool. The manifest corresponds to an XML file, commonly named *imsmanifest.xml*, with a root element <manifest> with the following child elements:

- <metada> is an optional element that describes the package according to IEEE 1484.12.1-2002 standard for Learning Objects Metadata. Other metadata can be included by their corresponding namespace.
- <organizations> is a required element with zero or more <organization> children that describe the structure of content resources.
- <organization> includes a hierarchical composition of <item> elements pointing to content resources by means of unique ids.
- <resources> is a required element that includes the references to all the content resources included in the package by means of a unique id and a sequence of one or more <file> children.
- <manifest> is an optional element that specifies zero or more sub-manifests.

The manifest, schemas and resource files are packed into a compressed file, typically using ZIP format. The manifest and the schemas are located at the root and resources are organized into subfolders.

B. IMS Question and Test Interoperability

IMS Question and Test Interoperability is defined by the standard as "(it) describes a basic structure for the representation of question (item) and test (assessment) data and their corresponding results reports" [9]. The latest version of IMS QTI is 2.1 but at present PoliformaT only supports version 1.2, then this will be our target version.

There are two aspects covered by the standards ASI and Results Reporting, both are totally independent. ASI stands for "Assessment, Section, Item" and it is the component relevant to us because it deals with assessment authoring.

An IMS QTI ASI compliant document is an XML document validated against the schema *ims_qtiasiv1p2.xsd*. Our LMS uses IMS QTI ASI as the format to import and export exams and, among all the compliant structures, it only accepts the following one:

- A root element <questestinterop> with one <assessment> child. In fact only one <assessment> element can be included in an IMS QTI compliant document.
- The <assessment> element includes one <section> child.
- The <section> element includes a sequence of <item> elements. Every item implements an assessment question.
- An <item> element is compounded by three elements:
  - o <itemmetadata> describes question features like the type of question.
  - o <presentation> includes presentation content like enunciate and response options.
  - o <resprocessing> includes the description about the response alternatives (correctness, score, etc).

III. DOCBOOK AUTHORING ENVIRONMENT

The basic Docbook 4 tools are the DTD files and the XSLT stylesheets that produce web and print formats. Both of them are open source and can be obtained from the Docbook site [1].

These basic tools have to be complemented with an authoring frontend and some customizations in order to be productive and fulfill particular application requirements. Either DTD and stylesheets are fully customizable at several levels of complexity [13][14], usually simple customizations will suffice.

In the next two sections we briefly describe our Docbook markup customization and the tools that we have selected to produce and manage documents.

A. TeachdB markup

We have minimized markup customization in order to maintain the maximum compatibility level with Docbook. We have named our customization TeachdB [7].

Our learning/teaching documents are subject themes, lab guides and exams. Subject themes include theory and solved and unsolved exercises. Lab guides include theory briefing, lab activities and exercises. Exams are built reusing exercises from subject themes and lab guides. Exam questions can be open or test type, at the moment we only support multiple choice one answer test questions.

The delivery formats were initially chunked HTML briefings, to be used as class presentations, and PDF to deliver class notes and exercise collections to students in paper format. Lab guides and exams are delivered to students as PDF. The preference of paper format was because we began our Docbook working before PoliformaT started. Even after PoliformaT paper format is still widely used.

From the technical point of view subject themes and lab guide documents use <article> as root element. To distinguish one type of document to another we have made the next Docbook customization:

- *role* attribute of element <article> is restricted to "class" and "lab" values.

An important issue in our customization is exercise management and then the main markup customization part is related to exercises. We implement exercises using <qandaentry> elements. Docbook organizes <qandaentry> elements inside <qandadiv> elements and <qandadiv> elements are grouped into <qandaset> elements. We classify exercises into delivered in class notes, stored in repository and candidates for exam. This is implemented by means of the following customization:

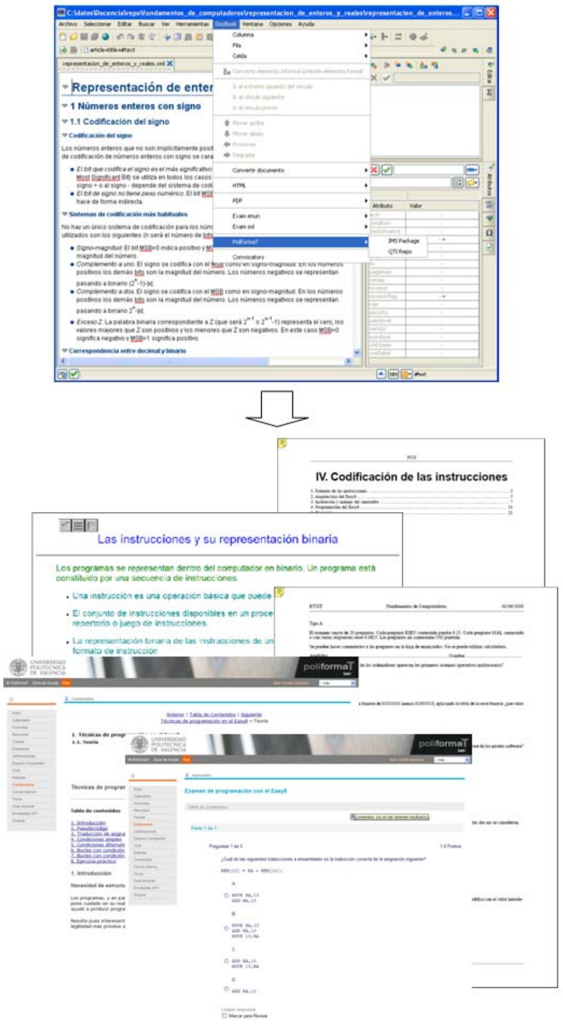
- *role* attribute of element <qandadiv> is restricted to "class", "repo" and "reserved".

In Docbook a <qandaentry> element has one <question> child and cero, one or more <answer> children. We have customized <qandaentry> elements in order to be able to:

- Mark an exercise as solved (show the solution) or unsolved (hide the solution).
- Know their last use type and date.
- Implement multiple choice one answer test questions.
- Select a solution pattern or a complete solution depending on the type of document delivered (i.e. exam or exam solution).

The customization consists of:

- *role* attribute of <qandaentry> element is restricted to "example" and "exercise" values.
- Two new attributes *lastusetype* and *lastusedate* are added to <qandaentry>.
- A new element <options> is added to <question> element in order to implement multiple choice questions.
- A new attribute *type* is added to <answer> element, with available values 'response' and 'solution'.



Representación de enteros
1 Números enteros con signo
1.1 Codificación del signo
IV. Codificación de las instrucciones
Las instrucciones y su representación binaria
poliformaT

Figure 3. One content source multiple delivery formats.

Written exams are produced using XInclude in order to avoid content duplication and allow content reuse. In this way exams are built by pointing exercises located in subject themes or lab guides. This mechanism has improved noticeably the exams quality in such a way that exams errors are much less probable. Exam document have their own new markup tree described in [7].

The extension to deliver IMS CP and QTI has not imposed any additional markup customization if we limit content to text and still images. However these new formats are able to include interactive elements, like Java applets and multimedia clips, into both module sections and QTI items. So we are considering a customization of Docbook <audioobject> and <videoobject> elements, and the addition of the element <appletobject> at the same level as the former ones. At the moment we simple use standard Docbook <ulink> elements to include these types of materials.

We also plan to extend our QTI item support to fill-in the blanks that will require extending our <qandentry> customization.

Another candidate for customization is metadata, at the moment we do not include metadata inside IMS CP or QTI apart from the general metadata included in PoliformaT templates. Hierarchical elements in Docbook include info children like <articleinfo>, <sectioninfo> and <blockinfo> that can be customized as metadata containers [3].

B. Authoring tools

In order to be productive a user friendly editor has to be selected. XML editors tend to be difficult to use by non experts and this has been an important handicap to XML. Nowadays the situation has changed and there are at least two products that offer a comfortable interface to authors:

- XMLMind XML Editor (XXE) [16].
- oXygen XML Author [17].

XXE was probably the first XML editor oriented from the beginning to authors. We have followed its development and its evolution has been impressive and it certainly offers an easy to use and productive interface. At present XXE is our choice. oXygen has been mainly oriented to XML programmers and recently the Author version has been released. It is an alternative to keep in mind. Both XXE and oXygen XML Author are commercial products but very affordable licenses are available for academic use.

Our Docbook customization can be easily ported to both editors because it mainly deals with Docbook itself. Our first attempt has been to embed it inside XXE.

IV. IMS CP and QTI Production Tool

After becoming aware of PoliformaT use of IMS CP and QTI formats for importing and exporting courses and assessments we began to extend our production environment [7] to automatically produce these formats from our pool of TeachdB documents. Basically it requires the implementation of some XSLT stylesheets that reuse Docbook XSLT for HTML generation and some simple console scripts. The scripts are platform/tool dependent but XSLT code is portable.

A. IMS CP Generation

Every subject inside PoliformaT has a Content site where content is organized as modules that contain one or more sections. Every section can be a single or chunked HTML page. The content inside Content site can be exported or imported as a whole or module by module using IMS CP.

We use the Content section of our subjects in PoliformaT as a complementary way to publish theory and problem collections. This material is also published as PDF in another subject site named Resources. The added value of HTML is that it allows enriching our learning documents with interactive elements like Java applets, multimedia clips and web links.

We import the subject content in PoliformaT by modules, in such a way that every module corresponds to a subject theme.

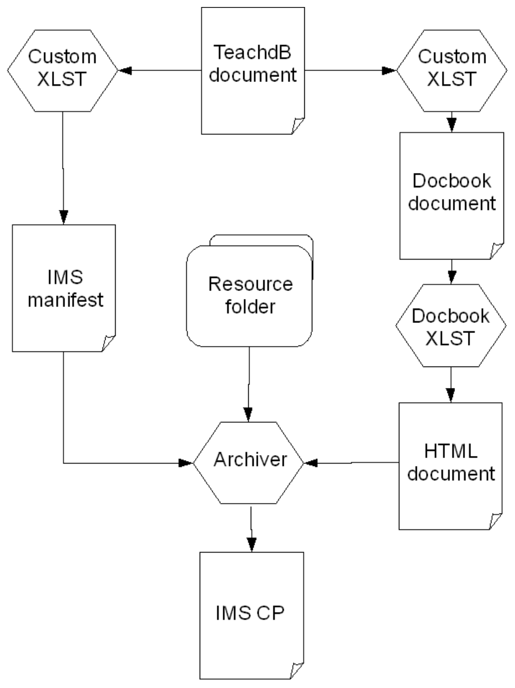


Figure 4. IMS CP generation procedure.

The exercises customization performed in TeachdB allows marking exercises to be solved, unsolved or reserved and then this characterization permits to filter what set of exercises will be published in the Content site or in the Exams site (as we will shown in the next section) and what exercises are reserved to produce written exams.

In order to get from a TeachdB subject theme file the theory and the solved exercises and to pack them into an IMS CP to be imported in PoliformaT we perform the procedure described in figure 4.

We have implemented two XSLT style sheets. One of them produces an intermediate Docbook document from TeachdB in order to use the standard Docbook XLST, properly customized, to produce a single HTML page. The other XLST style sheet produces the manifest document for the content package.

Finally an archiver packs the manifest, the HTML page and the local resources into a ZIP file that will be delivered to PoliformaT.

B. *IMS QTI genaration*

The implementation of QTI import/export procedure in Sakai 2.4, corresponding to PoliformaT at present time, is rather rudimentary. The automatic importation procedure only accepts a QTI valid file, with some additional restrictions, and any other auxiliary material (figures, graphics, applets, etc) should be uploaded manually to a public folder inside PoliformaT. Considering this situation our QTI assessment generation procedures is depicted in figure 5.

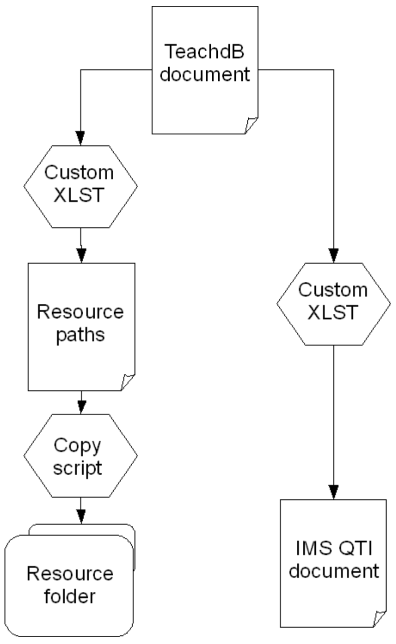


Figure 5. IMS QTI generation procedure.

The custom XSLT that produces the IMS QTI document takes from the TeachdB input file <qandaentry> elements that include an <options> children and that are included inside <qandadiv> elements with *role* attribute set to “class”. The Docbook hierarchical elements are translated to QTI and content markup is converted into HTML by reusing templates from Docbook XSLT. This is implemented by importing the main Docbook HTML style sheet into our QTI producer style sheet.

The other XLST style sheet produces a resource description text file, containing path location for resources (i.e. still images), that is generated by parsing the source TeachdB file. This description file is later read by a copy script that makes a copy on a resource folder of every resource used by the <qandaentry> elements included in the IMS QTI file.

The QTI file is imported in PoliformaT by the import procedure available in the subject Exams section and the resource folder is uploaded by WebDAV to a previously stated public folder inside PoliformaT.

C. *XXE Customization*

We have implemented inside XXE editor the procedures describe in the two previous sections. XXE has very powerful customization capabilities that allow adapting it to any XML compliant markup offering a friendly editing interface based on CSS [19]. In the professional edition the application menu can be customized with new entries that correspond to XML code able to call external XSLT style sheets and a set of command implemented inside the editor [20].

The customization is performed by including inside a customization folder the DTD for TeachdB, the CSS style sheet to get a friendly editing interface, a set of document templates, the XSLT style sheets that perform format generation and the customization file that integrates all the previous elements. This configuration file is XML compliant according to a schema created by XMLmind specifically to configure XXE. XXE itself has an addon that allows editing configuration file.

The TeachdB customization for XXE Professional Edition, in the state as it is at the time of this writing, can be downloaded from [21]. Figure 6 shows how XXE looks like when the customization is active.

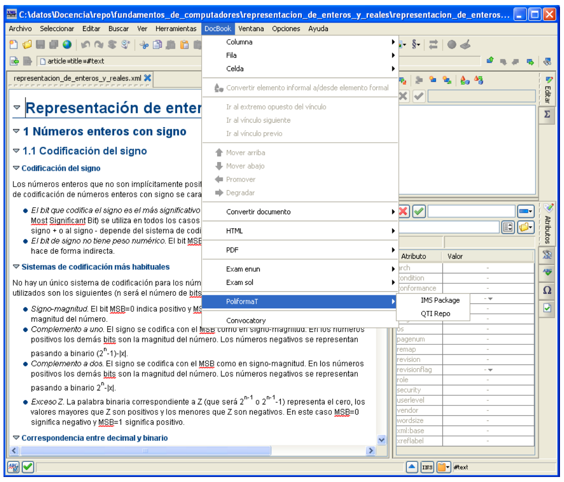

C:\datos\Docencia\repo\fundamentos_de_computadores\representacion_de_enteros_y_reales\representacion_de_enteros...
Archivo Seleccionar Editar Buscar Ver Herramientas DocBook Ventana Opciones Ayuda
article title #text
representacion_de_enteros_y_reales.xml
Columna
Fila
Celda
Convertir elemento informal a/desde elemento formal
Ir al extremo opuesto del vínculo
Ir al vínculo siguiente
Ir al vínculo previo
Mover arriba
Mover abajo
Promover
Degradar
Convertir documento
HTML
PDF
Exam enun
Exam sol
PoliformaT
IMS Package
QTI Repo
Convocatory
Editar
Atributos
Atributo Valor
Representación de enter
1 Números enteros con signo
1.1 Codificación del signo
Codificación del signo
Sistemas de codificación más habituales
Correspondencia entre decimal y binario

Figure 6. XXE with customized format generation menu entries.

V. Environment Use and Learning Impact

We have been using the proposed e-learning production environment since course 2005-06. We have authored three subjects with it, one of them was created just reusing content from the other two by means of XInclude.

A. *Environment use*

The procedure to produce a subject is to decompose it into modules or themes. Every module is created using the <article> template of TeachdB inside XXE. The content of a module is organized into sections, commonly up to a three level deep section tree. Inside a section the content is organized into <simplesect> elements that are terminal sections in the document tree. A <simplesect> can be seen as a slide and contains an optional title and a sequence of content elements like paragraphs, items lists, tables, images, etc.

Exercises are included inside a <sect> or <simplesect> element using the <qandaset> container. The organization of exercises is explained previously in section III.A. Exercises are authored filling the <question> element with content elements (paragraphs, item lists, tables, images, etc) that define the exercise enunciate. If the exercise is a test question then the last element in <question> has to be one <options> element that defines the multiple choices for the test question. The first item of the <options> element has to be the true answer. When test exercises are imported in PoliformaT they are randomly reordered when students start the exam. After the <question> element we can optionally include one <answer> element with a response template (incomplete tables or images, clues for the solution, etc) and one <answer> element with the complete solution. The attribute *type* of <answer> element distinguishes both cases as we have described in section III.A.

To produce a module to be imported in PoliformaT we select Docbook menu, PoliformaT and IMS Package (figure 6). This produces an imscp.zip that is imported in the Content section of the subject in PoliformaT without any further manual operation.

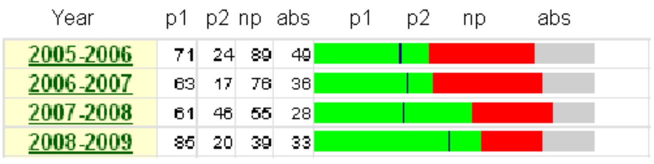
A similar procedure is followed to produce a QTI exam. We do Docbook, PolformaT and QTI Repo (figure 6) then a qti.xml file is generated that is imported in the subject Exams section in PoliformaT.

A video demo is available at [27] showing the previously describe procedures.

B. *Learning impact figures*

The switch from using presentation slides and exercises collections provided only in PDF format, into our new approach has had a noticeable impact in the learning process. We thing that there are three main reasons for this improvement, first the new IMS Content Package format allows including more active elements like videos and applets.

Then we are able to make use of the new streaming platform PoliTube started in 2007 by our university to deliver learning videos. They can be included in modules by just a PoliTube link or embedding the video using the <embed> HTML element. Another reason is related to QTI as a more dynamic way of publishing exercises, together with the capability of publishing exercises with or without solutions from a single content source. Finally, the procedure to produce written exams automatically by reusing exercises already published and tested has improved quite noticeably the quality of written exams and this is greatly appreciated by students.



Year	p1	p2	np	abs
2005-2006	71	24	89	49
2006-2007	63	17	78	38
2007-2008	61	46	55	28
2008-2009	85	20	39	33

Figure 7. Learning results for a sample subject.

Some figures that measure this improvement in the learning process are shown in figure 7 and in figure 8.

Figure 7 shows how the percentage of passing students has got bigger and bigger and that the contrary stands for the percentage of non passing students. The percentage of absent students is more o less the same.

Figure 8 shows that since the proposed authoring environment is in use (2005) there is a jump of 1.5 points in the mean lecturer opinion pool score (upper line over lower line).

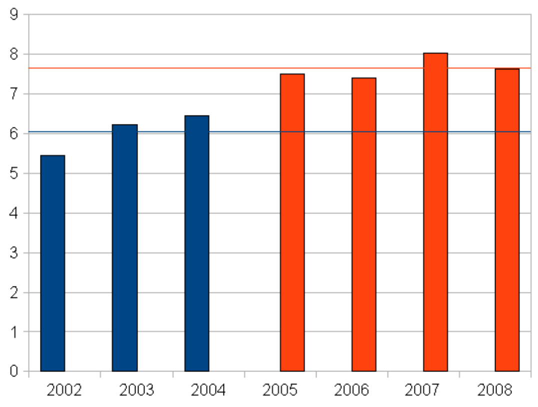


Figure 8. Lecturer opinion pools scores.

VI. Comparing to Other Approaches

The technique that we have presented has to be compared with the following available alternatives:

- Editors embedded in Sakai 2.4.
- Common word processors.
- Specific IMS authoring tools
- E-learning editor eXe.

Sakai 2.4 includes edition tools to produce content modules and assessments [10]. These tools can not be used outside the platform and then their use requires Internet connection which nowadays can still be an inconvenient. The editors are implemented as Java applets and then they suffer from high startup time and lack of functionality compared to common editor applications. Sakai editors neither support good quality printing format.

The previously mentioned shortcomings are not present in commonly used word processors like MS Word and OpenOffice Writer. These editors have another inconvenient, their XML formats are not structured (flat) and very complex, particularly they include a lot of formatting information in their markup [18]. This makes very difficult to implement automatic tools to translate documents to IMS formats, particularly QTI. Our Information Systems Support Department has developed a macro to produce QTI from MS Word documents but it has many restrictions that forces authors to pay a lot of attention in the editing process and limits the type of content in the QTI documents produced. The Open University [22] has also developed an MS Word customization that allows using a custom markup language inside Word in order to facilitate the generation of IMS formats [23]. This markup can be seen as a quite limited Docbook subset. Furthermore, XSLT has to be built from scratch loosing the capability of reusing the powerful and very well tested Docbook XSLT style sheets.

For the present work the more relevant IMS formats specific tools are RELOAD [24] as Content Packaging tool and AQuRate [25] as QTI editor. RELOAD is just an IMS CP editor which means that it is not a content item editor. Content items should be created with other tools and RELOAD allows integrating them into an editable IMS CP.

AQuRate produces QTI 2.1 assessments with a friendly interface and it supports the set of items that we are interested on. However, it lacks from good quality printing generation and we still depend on paper format. AQuRate neither includes at the moment a friendly table editor. Another shortcoming for us is that only the assessment part or our learning content can be produce with AQuRate, then theory content and exercises have to be kept appart. QTI produced with AQuRate generates errors when importing it into PoliformaT, then an additional adaptive transformation must be applied.

Finally eXe editor is an open source project (exelearning.org) that intends to offer a free e-learning oriented content editor that enforces compliance with IMS CP and QTI. The main problem with this tool is that it is not easily customizable, for example it is difficult to get good printing format. The native format is XHTML that is good for the web but really inferior to Docbook in relation to automatic document processing and management.

Our framework built on Docbook has all the benefits of the alternatives mentioned and it avoids their drawbacks. The XML editors we have mentioned, XXE and oXygen Author, are Java applications that are installed locally and do not have functionality restrictions. Docbook markup is well structured and simple, facilitating XML conversion. Furthermore, Dosbook XSLT style sheets allow full control on the production of screen format (HTML) and paper format (PDF). We certainly appreciate this feature and in fact this was one of the main reasons for our initial choice of Docbook. Finally our authoring environment allows producing theory and exercises together which is quite convenient in order to have a consistent content view. When we generate IMS CP and QTI from our customization they match the patter that PoliformaT imports without errors.

VII. Conclusions and Future Work

After having used Docbook extensively as our framework for developing and deploying learning/teaching materials [7][15], we have presented in this work an extension for Docbook interoperation with IMS learning formats in order to easily produce e-learning content that can be automatically imported in our Sakai based LMS platform PoliformaT. Our work is being considered by our Information Systems Department and it received a good appreciation at the September 2008 IMS Quaterly Meeting [26]. A video demonstration of how our tool produces IMS learning formats inside XXE and how these formats are imported into PoliformaT can be seen at [27]. This extension of format delivery from our learning content pool has been fairly easy to implement and it has proven to be very effective, consolidating Docbook as a solid e-learning production tool alternative.

As IMS formats management continue to improve in future Sakai releases we will keep track adapting our environment to the new features. Particularly it is foreseen that Sakai will support IMS Common Cartridge that mixes IMS CP, QTI and other IMS standards into a common standard for e-learning content delivery and interchange.

Docbook is also evolving to version 5, nowadays version 5 is not mature enough to replace version 4 in production but it is worthwhile to consider migration at the right time. Docbook 5 is a complete rewrite of Docbook based on RELAX NG schema validation and includes new features like using XLink as the single linking mechanism, namespaces that allow embedding into Docbook documents other XML makups like MathML and SVG, and easier customization. Porting Docbook 4 documents to Docbook 5 requires little changes that can be automatically performed by simple XLST code.

There are other markup languages that share some of Docbook nice features like DITA. But at the moment we have not found any reason to leave Docbook. If this has to be done in the future the simplicity and good structure of these XML markups make their interoperation easy.

References

[1] Docbook main sites. http://www.docbook.org, http://docbook.sourceforge.net.

[2] IMS Global Learning Consortium site. http://www.imsglobal.org.

[3] Martínez-Ortiz, I., Moreno-Ger, P., Sierra-Rodríguez, J.L., Fernández-Manjón, B. "Using DocBook and XML technologies to create adaptive learning content in technical domains", Internationa Journal of Computer Science and Applications, vol. 3, num. 2, pp. 91-108, 2006

[4] Koper, R., Manderveld, J., "Educational modelling language: modelling reusable, rich and personalised units of learning", British Journal of Educational Technology, vol. 35, pp. 537-576, 2004

[5] Walsh, L., "Using extensible markup languages for the single source delivery of teaching resources via print and the web: a practical example" Australian Society for Computers in Learning in Tertiary Education ASCILITE, Perth, pp. 913-923, 2004

[6] Delgado, C., Pardo A., Muñoz M., De la Fuente, L., "E-LANE: an e-learning initiative based on open source as a basis for sustainability" Int. J. Cont. Engineering Education and Life-Long Learning, Vol. 17, No. 1, 2007.

[7] González, A., "Teaching document production and management with docbook". II International Conference on Web Information Systems and Technologies, WEBIST 2006. Setúbal (Portugal)

[8] IMS CP site. http://www.imsglobal.org/content/packaging/index.html

[9] IMS QTI site. http://www.imsglobal.org/question/index.html

[10] Sakai project site. http://sakaiproject.org/portal

[11] R. Mengod. "PoliformaT, the Sakai-based on-line campus for UPV - history of a success". 5th Sakai Conference, Vancouver, BC, Canada, 2006.

[12] Docbook to SCORM conversion tool site. http://pyxx.org

[13] Walsh, N., *DocBook: The definitive guide*. O'Reilly. 2003. http://www.docbook.org/tdg/en/html/docbook.html

[14] Stayton, B., *DocBook XSL: The complete guide*. Sagehill Enterprises. 4th Edition, 2007. http://www.sagehill.net/docbookxsl/index.html

[15] González, A., "Authoring reusable slide presentations". III International Conference on Web Information Systems and Technologies, WEBIST 2007, Barcelona (Spain)

[16] XMLMind XML Editor site. http://www.xmlmind.com/xmleditor

[17] oXygen XML Editor site. http://www.oxygenxml.com/

[18] González, A., "Making, organizing and reusing learning material with OpenOffice" SEFI Annual Conference. 2004, Valencia, Spain), 2004.

[19] Shafie, H., *XMLmind XML Editor – Configuration and deployment*. Pixware, 2007.

[20] Shafie, H., *XMLmind XML Editor – Commands.* Pixware, 2007.

[21] TeachdB customization for XXE Professional Edition. http://www.disca.upv.es/agonzale/teachdb/teachdb_xxe.zip

[22] The Open University site. http://www.open.ac.uk/

[23] Greenberg, J., "Open University consider changing course", Microsoft Office System, Customer Solution Case Study. 2004.

[24] Reload tool site. http://www.reload.ac.uk/editor.html

[25] Aqurate tool site. http://aqurate.kingston.ac.uk/index.htm

[26] IMS GLC Quaterly Meeting, September 2008. http://www.imsglobal.org/sept2008meeting.html

[27] Video demonstration. http://www.disca.upv.es/agonzale/educon2010/demo.html