Creating the first SCORM object

Victor Gonzalez-Barbone\textsuperscript{a,\ast}, Luis Anido-Rifon\textsuperscript{b}

\textsuperscript{a}Facultad de Ingenieria, Universidad de la Republica, Julio Herrera y Reissig 565, CP 11300 Montevideo, Uruguay
\textsuperscript{b}E.T.S.E Telecomunicacion, Universidad de Vigo, Campus Universitario, CP 36310 Vigo, Pontevedra, Spain

\textbf{A R T I C L E   I N F O}

\textbf{Article history:}
Received 30 October 2007
Received in revised form 14 April 2008
Accepted 15 April 2008

\textbf{Keywords:}
Authoring tools and methods
Distance education and telelearning
Interactive learning environments

\textbf{A B S T R A C T}

The creation of the first SCORM (Shareable Content Object Reference Model) object offers some challenges and difficulties which go beyond the facilities offered by content generation applications. In particular, the creation of really reusable, searchable learning objects requires a detailed consideration of metadata, where some institutional aspects may be unclear or not available. This work describes creation of a first learning object, from software tools installation to final packaging. It aims at a wider perspective than that offered by handbooks or user guides for content generation tools, generally poor or altogether deprived of suggestions on how to go about to achieve reusability, interoperability, durability and accessibility as conceived by the SCORM standard. Only free software and internet publications are used as references. The creation of a simple SCORM package with the Reload Editor is described step by step, and the package created is then tested using Reload SCORM Player, allowing for the detection of some difficulties and alternatives of solution. Help available and some commented references are afterwards indicated. A list of suggestions finally emerges, to the purpose of solving beforehand most of the uncertainties, defining a consistent learning object creation scheme and reducing training time to master tools and metadata generation. As a conclusion, some limitations found along the work are pointed out, in particular the necessity of adopting or defining a LOM (Learning Object Metadata) application profile together with an institutional strategy to face metadata creation efficiently.

© 2008 Elsevier Ltd. All rights reserved.

1. Introduction

The cost measured in time and effort required to create a learning object showed to be much greater than originally envisioned, to the point of hindering the spread of eLearning as an alternative to traditional education. The examination of created content showed in turn a high degree of redundancy among different learning situations: different audiences should receive education or training on the same subjects with a content core which could be common to them all, plus some specifically oriented content proper of the specialization sought. Reuse of content for different learning objectives was seen as a first step in cost reduction. The essential concept to implement this solution is the sharable learning object, a complete unit of educational material with a single, narrow educational or instructional objective. The same set of learning objects adequately conceived could allow for different instructional designs with their own objectives through the definition of different aggregations and sequences of the same learning objects. These ideas led to different proposals for the creation of reusable learning objects, among which SCORM (Shareable Content Object Reference Model) seems to be the most promising one, counting with ample support in government, educational and corporate organizations (\textit{Carnegie Mellon, 2004}).
Creating a reusable content object is no trivial task: besides the adequate selection of material for the instructional goals in sight, information about the learning object must be provided in a structured way so as to allow search engines to find it and instructors to evaluate adequacy of its contents without having to look inside the object.

In many handbooks and best practice guides it is assumed, for the creation of content, the existence of a group of specialists with different roles: instructional designers, authors, content developers, programmers, subject matter experts, course managers and instructors to deliver the course (Carnegie Mellon, 2004). Each one of these specialists has a specific role, having only a limited knowledge of the other parts of the content generation process. While this may be the ideal situation, in most cases developer groups are much smaller: the subject matter expert must frequently act as author, content developer or even programmer, sometimes also act as instructor or manage the delivery of the course. Content generation tools go in this direction, offering facilities for the easy creation of SCORM content, requiring less specialization apart from the subject of learning (Carnegie Mellon, 2004).

The main purpose of this work is to examine the creation of a first SCORM object towards the goal of producing learning objects of sufficient quality as to be lodged in content repositories of wide access. This is a crucial point to reach in the real world the massive use of the capacities provided by eLearning technologies.

This article is structured as follows:

- Quality goals for the creation of effective reusable content objects are identified.
- Some eLearning terms are reviewed and the SCORM standard is introduced.
- Software content authoring tools available are reviewed and selected.
- Installation of tools is described.
- A simple, SCORM compliant learning object is created and verified with the help of the installed tools.
- Some comments and suggestions emerging from the creation process are presented.
- Issues concerning quality of SCORM objects creation are examined.
- Some final conclusions end the article.

2. Learning objects and the SCORM standard

In this section a brief review of relevant eLearning terms is presented, followed by an introduction to the SCORM standard for the creation of reusable learning objects. More detailed and specific definitions can be found in (E-Learning Guru, 2004; JISC/QCA, 2006; Kaplan-Leiserson, undated; Windeatt, 2002).

eLearning, which stands for electronic learning, applies widely to any process of learning supported by the use of ICTs (Information and Communication Technologies) such as the Internet, Intranets, local or wide area networks, stand-alone computers, CD-ROMs, video and audio tapes, interactive TV, wireless and mobile devices.

Content, in an eLearning context, refers to information captured digitally and imparted to learners in formats such as text, audio, video, animation, simulation, and others. Content management comprise the technical and operational processes for managing digital educational content: learning items and tests.

A learning platform is a set of learning and teaching tools based on networked computer resources. For students, the learning platform determines the focus of their learning activities as well as delivery of content, resources and help required to support activities. A learning platform is usually considered to provide at least curriculum mapping, student tracking, communications tools, tutor and student support, learning and assessment delivery tools. One of the main purposes of a learning platform is the delivery or content to students.

A learning object is a reusable, media-independent collection of information used as a modular building block for eLearning content. A Shareable Learning Object (SLO) is a small piece of learning material with a single learning objective as purpose, independent from other learning materials, which can be used in different settings and combined as appropriate.

SLOs are most effective when organized by a metadata classification system. Metadata is the name given to information about content that enables it to be stored in and retrieved from a data repository according to different selection criteria. In eLearning contexts metadata provides details about a learning object, such as author, title, subject, date of creation, version. Metadata is most useful when it conforms to a standard. Typically metadata is recorded in XML files which are read and interpreted by delivery systems such as learning platforms.

An authoring system or authoring tool is a software application used by instructors and instructional designers to create eLearning courseware, generally media-based learning content. An authoring tool does not require computer expertise or programming knowledge; it enables non programmers to create eLearning programs. The Reload Editor is one such tool.

SCORM (Shareable Content Object Reference Model) is a collection of standards and specifications for the packaging and sequencing of learning material in the form of shareable, reusable content objects. A SCORM learning object comprises a collection of assets, which in SCORM terminology designate pieces of instructional or educational material. Assets within an object admit different organizations, different forms of aggregation and sequencing oriented towards a definite learning purpose.

SCORM proposes four basic qualities required to build effectively reusable learning objects (Carnegie Mellon, 2004; Berranga Flores & Garcia Peñalvo, 2004):
• **Reusability.** The content must be independent of the learning context, apt for use in different situations, for different audiences, on different delivery platforms with different applications or tools.

• **Interoperability.** The content must run on several application programs, environments, hardware and software, whatever the tools employed in its creation.

• **Durability.** The content must go on running without any modification upon any change or update in systems hardware or software.

• **Accessibility.** The content must be identifiable and locatable when needed, to the learning requirements needed. Its adequacy for the objectives in hand must be apparent without requiring access to the content or paying royalties for it, through the provision of sufficient information about each learning object.

SCORM intends to reach these qualities through the use of SCOs containing instructional or educational material, assets in SCORM terminology. SCORM compliant learning objects can be delivered in a runtime SCORM environment, eventually provided by a learning platform or from a learning object repository supporting SCORM compliant objects and providing the adequate search facilities. The use of metadata allows to identify an locate instructional material in learning content repositories online (Carnegie Mellon, 2004).

Creating a SCORM object able to provide the four qualities afore mentioned is no trivial task. Metadata, in particular, show the greatest difficulties: they must follow strict conditions of structure and content to be of general use. The set of metadata for a learning object is called **LOM (Learning Object Metadata).** IEEE standard LOM v1.0 describes more than 70 metadata items (Anido Rifon & Rodríguez Artacho, 2002), a number enough to show the difficulty of reaching consistent metadata generation. In addition, there are uncertainties in the specifications themselves, in particular concerning the reach and granularity a learning object must have (Berlanga Flores & García Peñalvo, 2004), of utmost importance to define the different organizations which make possible the use of the same learning object in different courses.

In the next sections, software tools for SCORM learning object authoring are selected, and a step by step installation description follows.

### 3. Software selection

The set of chosen tools is the one provided by the Reload Project (RELOAD Project, homesite), specifically the editor (Reload Editor) and the player (Reload SCORM Player). Though alternatives exist, the Reload Project tools are considered the most adequate for the generation and visualization of content available at the moment, at least within the free software community, which this work set out as its limit. The Editor and the Player can be downloaded from the Reload Project site (RELOAD Project, 2004).

The eXe Project intends to generate tools able to generate SCORM compliant content and other specifications compliant content in an easy way, apt to be used by instructors and educators. It is supported by the Tertiary Education Commission of New Zealand, with the participation of the Lincoln University and the Technological University of Auckland. It is a promising proposal, a project open to new ideas, but still in an early phase of development, the most recent version to date being 0.11 (eXe, homesite). The proposal of using DocBook, a model of technical documentation based in XML, for the creation of SCORM content deserves attention (Martínez-Ortíz, Moreno-Ger, & Fernández-Manjón, 2004), though it may discourage authors used to purely graphics environments.

Moodle (Moodle, homesite), ATutor (ATutor, homesite), Claroline (Claroline, homesite) and Docebo (Docebo CMS, homesite) are free software learning platforms supporting SCORM 1.2 objects. Learning platforms’ sites usually provide directions to create SCORM objects using several different content creation tools, the Reload Editor being generally a preferred choice. Claroline offers alternatives such as Dreamweaver (proprietary), JavaScript, or tools such as the Reload Editor, eXe and others (Claroline, homesite). The Reload Project tools are, therefore, widely supported by learning platforms.

The installation described below was done over the Linux operating system. At the Reload Project’s site there are software and instructions to install the editor and the player in the Microsoft Windows operating system (proprietary).

### 4. Software installation

The necessary files to install the Reload editor and the Reload SCORM player are:

- Setup_ReloadEditor202_lnx.bin for the editor.
- Setup_ReloadSCORMPlayer121_lnx.bin for the player.

Both files are executable under Linux. They create a folder (directory), extract the files and place them inside the created folder. It is necessary to choose a higher folder under which the tools are to be installed, though:

- For individual usage installation, the user’s home folder would be the most appropriate place.
- For a system wide installation, where all the tools should be available to all users, there are two usual alternatives:/usr/local and /opt. As the applications to install are self contained software packages, generally alien to Linux distributions, it is recommended to use/opt.
The installation requires access to the Linux windows system, generally blocked for the superuser, so the installation must be carried on by an unprivileged user operating under a folder of his/her own or with appropriate permissions. An installation at the system level is described in what follows. An individual installation only requires invoking the installer, which creates a folder under the user’s home folder by default. Installation must be carried out on a command line terminal.

4.1. Folder for system installation

Creating the /opt/ReloadTools folder usually requires root privileges. The following commands access root, create the folder /opt/ReloadTools and assign it to user victor, group victor (change as necessary), taking the operator back to his usual unprivileged user state.

```
su
mkdir /opt/ReloadTools
chown victor:victor /opt/ReloadTools
exit
```

The rest of the installation is done by the operator as an unprivileged user.

4.2. Installing the editor

The installer recommends shutting all running applications before starting installation. Assuming files have been downloaded into folder /export/software/Reload, the command to install the editor is:

```
/export/software/Reload/Setup_ReloadEditor202_lnx.bin
```

Following the steps indicated by the installer, the operator must accept a license agreement, adjust the installation folder to /opt/ReloadTools/Editor, indicate where to make links for an easy invocation (may not be done or done afterwards), opt for an already installed Java virtual machine or install one from the installation package. A verification summary is shown before installation is actually done; here the operator can go back to correct anything not to his/her taste. Finally files are installed and success or errors are reported.

Whatever the installation folder, the installer creates a subfolder reload under the installing user’s home folder, where customization and log files are placed. Errors can be seen in the installation log file named ’Reload Editor.log’ located in the user’s subfolder reload/reload-editor.

The editor can be started directly in a terminal by giving the complete path:

```
/opt/ReloadTools/Editor/ReloadEditor
```

As it is usually a package alien to the Linux distribution users must create their own menu option and or invocation icon according to the window manager they are using (KDE, Gnome, others). To create the icon an image called reload-editor.gif can be found in the installation folder.

4.3. Localization

There are several localization packages available at the Reload Project Site. As an example, Spanish localization installation is described. Spanish localization package is i18n_ES.zip; others go by similar names. Localization must be done after editor installation. It is convenient to download localization files into the same folder where the editor was installed, or place it there for decompression. The following commands change the name of the existing i18n folder to i18n-en (English localization), then uncompressed the Spanish localization files:

```
cd /opt/ReloadTools/Editor
mv i18n i18n-en
unzip i18n_ES.zip
```

An i18n for Spanish localization appears in place of the original English one. Keeping the original i18n as i18n-en allows for the restoration of the English localization; localization folder in use is always named i18n, whatever its localization contents.

Now start the Reload Editor, choose menu option Tools/Options/Appearance/Language and select Spanish or any Spanish language country variant. The editor must be restarted to make changes effective.

Localization in Spanish is complete for all menu options and “balloon help” seen when placing mouse pointer over icons. Explanations of items in lower left area are not translated, and neither is online help.
4.4. Installing the SCORM Player

The SCORM Player installation follows closely the Editor’s installation. Within the same previous assumptions, installation command is

```
/export/software/Reload/Setup_ReloadSCORMPlayer121_lnx.bin.
```

Installation folder is now adjusted to /opt/ReloadTools/ScormPlayer.
To start the player a full path must be given:

```
/opt/ReloadTools/ScormPlayer/ReloadScormPlayer.
```

Menu options or icons can be created according to the user’s preferred window manager. The installation creates in the user’s home folder a subfolder reload/reload-scorm-player for customization and log. Installation log can be found in the file named ‘Reload Scorm 1.2 Player.log’ under ~/reload/reload-scorm-player.

The installed version does not offer localization or language adjustment options; the ScormPlayer seems, at the moment, limited to its original English version.

5. Creation of a SCORM Object: the essentials

The Reload Editor is a content package and metadata authoring tool: both content and metadata describing content can be managed within the editor. Creating content packages enables sharing educational content. Metadata describes this content and enables to find it in a thematic search, allowing for an assessment of suitability without actually examining the material. Any type of educational material can be included in a content package, from simple HTML or text files to video clips and animations.

In this section a description of SCORM content packaging is provided, a simple example described, and the main stages of object creation examined. This sections serves as an introduction to the actual creation of the SCORM compliant learning object which will be carried on immediately after.

5.1. SCORM content packaging

A content package is a single zipped archive containing all the files needed to deliver an educational content. A special file, imsmanifest.xml, known as “the Manifest”, included in the package, serves as a guide to the rest of the files. The manifest contains information on:

- Resources: any of the files included in the package, their type and mutual dependencies (e.g. an HTML file and its stylesheet, images or linked HTML pages).
- Metadata: information on the content files and the package itself.
- Organization: a sequence of resources serving an educational purpose. Many organizations are possible in a single package, distinguished by name. This feature enables the package to serve different educational needs on the same subject in different courses or learning instances.

Before the creation of the learning object can be undertaken, all resources must exist and be finished. The creation of a learning object is essentially a packaging process.

5.2. Object creation stages

The creation process involves the following stages:

1. All resources that must go into the package, finished and ready, are gathered together. The simplest way to do this is placing them all in a single folder.
2. Create a new, empty object.
3. Create at least one organization. Several organizations are possible; they may differ in the resources involved or in the order of presentation.
4. Add all required resources to the new object.
5. Create a simplest metadata set.
6. Optionally add more complex metadata. This generally done following a template adopted by the learning institution to which the object will belong.
7. Package all files into the SCORM compliant learning object.
5.3. A simple example object

Our example object will contain just a bunch of HTML pages, named One to Eight; these will be our assets, in SCORM terminology. Any other kind of content is of course possible; the creation process is the same whatever the type of material.

Our only organization will include all these assets in the order given; the student is expected to take them in the order they are shown to him.

It must be noted that auxiliary files may need to be included as assets, though the student does not interact directly with them. If a simulation application is included in the package, and the application relies on a simulation engine to run, the simulation engine must be accessible, i.e. within the package, for the simulation application to work.

6. Creation of a SCORM Object: the steps

The following sections show the steps to create a simple SCORM package using the Reload Editor.

6.1. Gathering resources

The following Linux commands create a working folder and two subfolders, one for the existing resources and another as a working place for building the new SCORM package. The same can be achieved using any file manager, i.e. MS Windows Explorer or any one of the various Linux file managers offered by the different Linux distributions.

cd takes user to his home folder, wherever he may be.
mkdir FirstPack makes a new folder where all files will be placed.
cd FirstPack
mkdir ResourcePkgs makes a subfolder where pre existing resources will be found an taken from.
mkdir ScormPack makes a working subfolder where files for the new SCORM package will be placed and created.

A set of test resource files can be downloaded from the Reload site in a compressed zip file named testpkg.zip (RELOAD Project, 2004d). Assuming the file was downloaded in ~/FirstPack/ResourcePkgs, the commands

cd ResourcePkgs
unzip testpkg.zip

Uncompresses some resource files including a subfolder called supp. Resources for the new SCORM package will be found and taken from this folder, ~/FirstPack/ResourcePkgs.

6.2. Creating the SCORM package

The following steps create a very simple yet SCORM compliant package.

1. Start the Reload Editor, by giving a command like /opt/ReloadTools/Editor/ReladEditor or clicking an icon or choosing a menu option, according to the installation performed.
2. Create a new SCORM package through the Reload menu options File/New/ADL Scorm 1.2 Package (Fig. 1).
3. Select the folder for the new content Package, /home/victor/FirstPack/ResourcePkgs (Fig. 2).
4. In the working window by the title SCORM Package, used for package creation, several panels or areas may be recognized (Fig. 3):
   - A list of files under the chosen folder ScormPack, showing some new files have been created, in particular imsmanifest.xml.
   - A Content Package area showing the state of the content package under creation, headed by a MANIFEST-... name and showing two containers, Resources and Organizations. The letters and numbers after the word MANIFEST are a unique identifier.
   - An Attribute – Value table, down in the middle panel.
   - A help box, which content changes according to item selected in Content Package area, down to the right.
5. Add a default organization. Right click on Organizations, choose Add organization. Make sure a one line text box appear right over the Attribute table, if necessary enlarging the area over the table (Fig. 3). Change the word Organization by Main, which is the conventional name for the default organization.
Fig. 1. Reload Editor showing file New menu options.

Fig. 2. Folder selection for new package creation work.

Fig. 3. SCORM package creation working window.
6. Import resources. Right click on ScormPack, choose Import Resources, select folder ResourcePkgs; all files in folder are imported by default. Check Include dependent files if necessary: it is always convenient to do so. A new folder of the same original name, ResourcePkgs, appears under ScormPack. When opened, a collection of HTML and other files are shown.

7. Add resources to the package, simply dragging each file from the resources folder ResourcePkgs to the Main organization. The final state is shown in Fig. 3. Files selected, including those referenced by them, are automatically imported into Resources, which can be verified by opening this container and subsequent files. Right clicking on each resource under Main allows to move the resource up or down, thus adjusting the sequence of presentation. Selecting the menu option View/Preview Content Package, the default browser is opened and the HTML files can be viewed in the order stated by the main organization.

8. Add a metadata container by right clicking on MANIFEST-..., choosing Add Metadata. Right clicking on the Metadata container recently created, choose Add Schema and Schema Version; ADL Scorm and 1.2 appear under the Metadata container.

9. To add metadata values, right click on Metadata and choose Edit Metadata. A number of metadata fields are ready to be filled (Fig. 4). None of these metadata are required to create a SCORM compliant package, though, so in a simple example metadata edit can be omitted. The edit metadata window allows for a Form view and a Tree view. A metadata application profile may be chosen among three: IMS LRM Profile, LTSN profile and UKCMF profile; fields to be filled change accordingly. Metadata can be exported as a standalone file (Export button) or a standalone metadata file imported (Import button).

10. Reload menu option File/Save saves changes made until now. To package the files into a SCORM object, Reload menu option Files/Zip content package asks for a name and location for the new SCORM package. In this example, ~/First-Pack/firstscormpak.zip was chosen as file name and location.

This is almost the simplest SCORM package that can be created. Other contents, such as PDF or JAR files, would invoke the adequate tools to work with. Metadata should be inserted, following institutional directives or a carefully prepared plan. Different organizations are possible and can coexist, as well as establishing prerequisites for an item, i.e. the previous completion of one or more items in the same package. Though simple as it is, SCORM compliance of this example package can be tested with the Reload SCORM Player.

6.3. Testing the new SCORM package

The following steps test the recently created SCORM package using the Reload SCORM Player.

1. Start the Reload SCORM Player, by giving a command like /opt/ReloadTools/ScormPlayer/ReloadScormPlayer or clicking an icon or choosing a menu option, according to the installation performed.

2. To import a SCORM package into the player use the Reload Scorm Player menu option File/Import SCORM Package. Select ~/FirstPack/firstscormpak.zip. A name for the package is asked for; no name was given as metadata during package creation. The name “Some Pictures” was chosen in the example. The new package appears under Imported Scorm Packages. A click on the imported package opens it and shows its organization in the middle area of the main window, enabling to choose a particular one if there were several organizations available (Fig. 5).
3. Menu option Run/Run Scorm Package opens the default browser and shows all HTML pages as browseable items, in the sequence established by the Main organization, or the organization chosen if there were several (Fig. 6). Once a page has been seen, trying to review it indicates that such item has already been seen, and is not shown. If the Quit option is chosen, and the browser window subsequently closed as indicated, trying to see the package again only shows a list of items all ticked as completed (Fig. 7). The option Scorm Player menu option Run/Reset Scorm Package allows to see the contents again, as if it were the first time.

In the following sections, some comments on the whole process follow, together with a review of some points and warnings deemed useful for a first contact with SCORM package creation.

7. Lessons learned

This section reviews the creation process adding some comments, warnings and suggestions of help to create consistent SCORM compliant packages.

7.1. Previous training

Even with a knowledge of the use of metadata in learning objects proposed by SCORM or IMS, the Reload Editor cannot be considered an intuitive tool. However the efforts of presentation and composition of the user interface, the intrinsic nature of
SCORM objects imposes some complexity. For an author with general knowledge of metadata and content packaging the following readings should probably suffice:

- **Reload Editor and content packaging**: A quick start guide ([RELOAD Project, 2004a](#)). An introduction to reusable objects and the Reload Editor (two A4 size printed pages).
- **Reload Editor**: Step by step to a first content package ([RELOAD Project, 2004b](#)). A quick ten step guide to create a simple learning object (two A4 size printed pages).

A less proficient or more demanding author will find useful to read and follow the tutorial included in:

- **Reload Editor introductory manual** ([RELOAD Project, 2004c](#)). This manual of 31 A4 size printed pages reviews the IMS specification, editor's installation and working area appearance. One of the tutorial chapters describes in detail a learning object creation from a set or source files provided by the Reload Project in the package test-src ([RELOAD Project, 2004d](#)). Its content are HTML pages and images. The tutorial also describes the creation of a standalone manifest file, customization of the editor, the SCORM elements and how a SCORM package can be created.

In Spanish there are also some references, similar in content to the short English references and eventually some add-on showing how to upload the object onto a particular learning platform:

- **Taller Práctico ''Creación de un LO IMS/SCORM''** ([Dieguez Cobo, 2005](#)). Describes shortly the creation of a SCORM object (LO, Learning Object), how to verify it with the player and how to upload it to the Moodle learning platform.
- **"Tutorial para crear paquetes SCORM y usarlos en Moodle"** ([Queralt Gil, 2005](#)). Describes shortly the SCORM protocol and SCORM objects, offers a ten step guide to create a SCORM object with the Reload Editor and how to create a Moodle activity based on the created object. Screen snapshots illustrate the different steps.
- **Several LMSs** include, in their documentation, instructions on how to create SCORM objects and upload them to the learning platform. Sometimes several authoring tools are considered ([Docebo CMS, homesite](#)).

## 7.2. Content

As seen in the example case, creation of a content package is a simple task in the Reload Editor. However, a first contact with no previous reading or lack of understanding of some key aspects will delay familiarization with the Editor or produce inconsistent packages. Some concepts deserve to be emphasized:

- **Assets or resources** can be of any type: PDF files, HTML pages, images, video, Java simulations in.jar files or any other form of digital content.
- **Assets must preexist**, all finished and complete, in locations accessible to the editor, were them local or remote.
- **The creation of a learning object** starts with menu option File/New/ADL SCORM 1.2 Package, giving it a name and location (folder) where all necessary files will be placed. At the top of the tree, node MANIFEST < identifier > represents the core file where all the package information is retained, including content description, packaging and presentation of the package. This node is in correspondence with file imsmanifest.xml.
- **Resources must be “imported” into the editor**, using menu option File/Import Resources or right clicking on the package under creation name, up on the left vertical panel. Only after importing resources will they be seen so as to select among them which ones will be part of the object. This import operation does not place resources inside the object, it only makes them available to be chosen as content of the object under creation.

![Fig. 7. Status of contents all seen shown in browser.](#)
Before placing content into the object, an organization must be created, right clicking on “Organizations” in the middle panel, selecting Add Organization. An organization defines a set of resources and a sequence of presentation. At least one organization must exist; the default organization is called “Main”; this will be the organization used when delivering the material in a learning instance if no other organization is chosen or exists. Warning: to give the organization a name, a flaw in the presentation of the window makes necessary to enlarge an almost hidden text box placed over the lower middle containing the Attribute – Value table. Once the text box is visible, the word Organization must be changed to Main or the name desired. The organization identifier, a self generated code in the Attribute – Value table should not be altered (unless the operator knows what he/she is doing).

The variety of organizations gives versatility to the object: all or part of the same contents may be presented in different order sequences according to different learning goals in different courses addressed to different target audiences.

A resource is inserted into the package dragging it from the left panel on to the organization in the middle panel. This operation makes the resource appear under the folder Resources of the middle panel, as well as under the organization folder. If the same resource is assigned to different organizations it appears in the Resource folder only once, but on each of the organizations where the resource is assigned. Right clicking on the organization and selecting Add Item has the same effect, but a name and location (called Referenced element) must be inserted. In this way, more control is gained at the cost of some additional complexity.

Resources and organization are not enough for a reasonable SCORM object. Essential metadata should be filled in.

Organizations and resources included in the package can be verified in a preview of the content package, selecting the menu option View/Preview Content Package. The object contents are shown in a browser. In a left frame an organization can be selected; organizations and resources are shown here. A selected resource will be displayed in the middle frame, if a browser can display it (e.g. it is an HTML page), or the adequate application will be started (e.g. a PDF viewer).

7.3. Metadata

A minimal set of metadata can be easily filled in. However, it is in metadata where the hardest difficulties for SCORM package creation reside if really useful SCORM objects are to be created (Carnegie Mellon, 2004). The Reload Editor allows for the creation of a basic set of metadata in a simple way:

• Right clicking on the Manifest node, Add Metadata: node Metadata appears under Manifest.
• Right clicking on the Metadata node and selecting Add Schema: the definition schema for metadata is inserted. For a SCORM object the mandatory schema is ADL SCORM.
• Right clicking once more on the Metadata node and selecting Add Schema Version: the schema version is inserted; at this moment 1.2 is inserted by default.

Though far from what SCORM metadata allows, this basic set of metadata is SCORM compliant and allows for packaging.

7.4. Packaging

This simple operation is achieved by selecting a menu packaging option, File/Zip Content Package. By giving a name and location a.zip file is created. This is the SCORM package.

7.5. Verifying SCORM compliance

The Reload SCORM Player provides a straightforward way to verify correctness of SCORM packaging. The most basic use entails importing the package using menu option File/Import SCORM Package and then viewing the presentation with Run/Run SCORM Package. Option Run/Reset SCORM Package turns the object back to its original state forgetting the eventual progress a viewer has made through it in a previous run. In the Reload SCORM Player middle panel the organization to be presented can be selected, and its condition changed to make it the default organization. A complete description of the Reload SCORM Player is available at

• “Reload SCORM 1.2 Player v 1.2 Introductory Guide” (RELOAD Project, 2004e). An introductory guide to the SCORM player.

The final verification of correctness must be done on the learning environment to be used, which must specifically provide support for SCORM objects. Well known delivery platforms such as Moodle (Moodle, homesite), ATutor (ATutor, homesite), Claroline (Claroline, homesite) and Docebo (Docebo CMS, homesite) are free software learning platforms which support SCORM 1.2.
8. Quality of SCORM objects

As seen in the former sections, creating a SCORM learning object with only basic metadata is a relatively simple task if the Reload Editor is used. Some previous training and caution are required to minimize time and effort. On the other hand, to achieve the qualities promised by the SCORM standard is far more challenging. The desired qualities of reusability, durability, interoperability and accessibility demand institutional decisions and strategies beyond the goodwill of an instructor or development group.

Among the essential elements required to achieve these quality goals the following can be mentioned:

- A sound institutional decision to generate good quality learning objects, together with an adequate support in human resources and materials. Definition of institutional objectives is part of these decisions.
- Recruitment of a team of education and information technology specialists really involved in the project. Allow for courses and training if necessary, in the use of tools as well as in specifications or best practices. In most of the cases, cost of training is rapidly recovered.
- Study of standards, specifications, recommendations, best practices, and their assessment to determine if they are useful for the institutional objectives.
- Adoption, modification or elaboration of a best practices guide for the purposes of the institution. This includes the selection of a catalogue system, adoption of some criteria to obtain unique identifiers preferably through specialized services, determining the degree of granularity of learning objects (not an easy task, for sure), defining a design process for courses and learning instances, adoption or definition of a style guide for presentation.
- Generating partial templates for metadata, at different levels (institution, department, course, object), to unify and reduce the realm of doubts during the content generation process. Once these metadata values have been defined, they should not be modified, the less as the higher the level (i.e. metadata at the institutional level should not vary at all).
- A careful plan, whenever possible following a process model, considering available resources, adaptable ones or others to be created, as flexibly as possible, to be of use in the new learning objects of a certain course or learning instance, keeping always in mind a use beyond the course at hand.
- A good instructional design, precise objectives definitions. This is always essential, whatever the form of learning or teaching, and frequently overlooked.
- Follow up control, effort statistics, cost logs, problems and solutions logs, so as to accumulate a history capable of offering support to planning.
- Periodical revision of recommendations, cautious consideration of changes but reflecting the dynamics of development exhibited by eLearning, institutional objectives and the public.

Though lengthy and difficult to put into practice, the former list does not exhaust the necessary cares to reach a successful production of learning objects. There exist, moreover, difficulties intrinsic to the specifications, standards and recommendations themselves. A brief analysis of these difficulties follows.

8.1. Criticism

There is no agreement in the reach of a learning object, its extension and granularity (Berlanga Flores & García Peñalvo, 2004; Carnegie Mellon, 2004). This forces an institutional decision on these points. A recommended practice is to define a learning object for each learning objective, to allow for flexible combinations of organizations (e.g. courses) useful to different purposes.

Something of the sort happens to assessment objects, where it may be convenient to create learning objects with the sole purpose of assessment, even with a fine granularity. The present SCORM specification does not allow references from within a SCORM object towards another SCORM object, which compels sequencing to be done at learning platform level. Very specific assessments force the student to come back to the learning platform to go on with the following assessment task, perhaps a minor inconvenience but something to be remembered on decision making (Carnegie Mellon, 2004).

Pedagogical orientation of IEEE LOM has been considered troublesome when filling in the values of metadata and when searching; it has also been pointed out the scarce use of education labels by the educators themselves (Berlanga Flores & García Peñalvo, 2004). SCORM does not have a specific pedagogical orientation in itself, and declares to be “pedagogically neutral” (Berlanga Flores & García Peñalvo, 2004; Carnegie Mellon, 2004). Pedagogical orientation is left to the content generators. The excellent Best Practices Guide elaborated by the Architecture and Learning Systems Lab at Carnegie Mellon University addressed specifically a student interacting with the learning material, leaving aside other approaches such as group working (Carnegie Mellon, 2004).

The addition in SCORM 2004 of sequencing rules, though it brings the possibilities of pre and post assessment, remedial work and conditional bifurcations, also imposes a new level of complexity for developers, adding to the existing worry on the intrinsic complexity of SCORM in its metadata (Godwin-Jones, 2004).

Other pieces of criticism refer to the lack of uniformity in presentation style when objects are reused by different institutions, security concerns on evaluation questions when JavaScript is used, rejections on security grounds by browsers when objects are not drawn from the same site where the learning platform resides (cross-domain scripting), and also philosophical and political questionings concerning SCORM as a USA Department of Defense initiative (Godwin-Jones, 2004). Though
these questionings are not to be minimized before serious consideration and pondering, use of SCORM is spreading, authoring tools improve their creation, and there are initiatives of adapting it to different uses and communities.

8.2. Application profiles

The extension and number of metadata elements in the SCORM proposal has led to the creation of application profiles. An application profile is a set of metadata elements selected from one or more metadata schemes combined in a new compound scheme supposedly more appropriate for the functional requirements of a particular application, while retaining interoperability with original base schemes (UKLOM, 2004). The application profile concept does not contradict standardization, since it maintains interoperability with base standards, just leaving blank non essential information. The application profile is usually simpler, requires less mandatory metadata elements, those required are more precisely defined, and not all metadata elements present in base schemes must be filled in. Application profiles are frequently created for the use of a community or institution in particular, as is the case of CanCore, an application profile for Canadian educational institutions (Fernández Manjón, 2006). The Dublin Core application profile includes only 15 metadata elements (Dublin Core Metadata Initiative, 2003). It is a common practice to reduce required metadata by managing a set of required metadata elements plus another set of optional ones (UKLOM, 2004).

CanCore (CanCore, homesite), contrary to other application profiles, provides a detailed guide for the interpretation and implementation of each metadata element in its LOM. UK LOM is an application profile addressed to the United Kingdom educational community, with the explicit purpose of reaching a minimum core of common metadata elements and an associated domain value for each of them (“a minimum common core of LOM elements and associated value spaces”). To achieve its application profile, UK LOM took as references twelve existing metadata schemes, IEEE LOM in the first place for considering itself an application profile of this standard, and CanCore, to which it owes a recognized strong influence (UKLOM, 2004).

In spite of these simplification needs, UK LOM advice is that practical applications implement the whole of LOM conceptual data, even if these remain hidden to the final user, as a means to secure interoperability with other application profiles based on the UK LOM Core and the IEEE 1484.12.1 LOM standard (Learning Objects Metadata Standard) (UKLOM, 2004).

8.3. Application profiles in Spanish

It is not possible, at the time of writing, to know for certain if there is an application profile for the Spanish speaking educational community or some work in progress, though it is considered of the utmost importance to involve some organization in its development. There are, however, some institutional experiences.

The Image Processing and Multimedia Technologies Service of the Scientific and Technological Centre at Oviedo University (Universidad de Oviedo. Servicio de Proceso de Imágenes y Tecnologías Multimedia, http://www10.uniovi.es/spi) describes a learning materials development process oriented to science teaching where a set of metadata elements based in CanCore was chosen, giving precedence to elements descriptive of general, educational and cataloging aspects within UNESCO’s knowledge classification system (Sampedro Nuño, Sariego Ferrero, Martínez Nistal, Martínez González, & Rodríguez Ruiz, 2004). Initiatives of this kind would ripen the benefits or an application profile common to the Spanish speaking community, ensuring the interoperability in a wide area, essentially defined by language, even though each institution may particularize aspects of its own interest based on the general proposal.

In Chile, the FONDEF Project called Learning through Learning Objects Repositories (Aproa, 2005), proposes the creation of learning objects over a platform of their own based on Macromedia Flash. First an objective for the learning object is defined by filling a template form in the Aproa platform, where cataloging data are collected. Based on these data the platform generates metadata and packages the object according to SCORM requirements (Aproa, 2005). This solution simplifies metadata work using an implicit application profile included in the platform and the form which defines the object creation. Aproa presents itself as an object repository strongly oriented to multimedia, targeting objectives of interest to Chilean Education. Dependency over a single proprietary tool, Macromedia Flash, is considered a serious limitation. This proposal diverges in respect to the orientation of the present work, where open options, variety of tools and free software are deliberately privileged.

9. Conclusions

Generating SCORM compliant learning objects is relatively simple when using the Reload Project Editor. It is free software, installation on the Linux free operating system is simple and can be done for an individual or at system level, location packages are available, documentation is good. If not an intuitive process, previous training for an effective use of the Reload Editor is not costly, though some points require insistence to avoid some common pitfalls. The Reload Project itself provides a viewer, the Reload SCORM Player, which can show the created SCORM object, though definite tests must be carried out on the target learning platform, if possible trying more than one.

Creating SCORM objects with the desirable qualities of reusability, interoperability, durability and accessibility is far more difficult, in particular because of the intrinsic complexity of the IEEE LOM metadata standard, the all enhancing definitions typical of standards aiming at general use, and some limitations of the standard itself, such as the lack of a well bounded definition of the size and granularity learning objects should have. Application profiles, described by the standard itself,
in particular CanCore and UK LOM, respectively addressing the educational communities of Canada and the United Kingdom, seem to be a practical and valid answer to these problems. The annotated difficulties are traceable to the intrinsic complexity of creating reusable learning objects, and even to learning materials preparation in general. The tools and techniques available for SCORM packaging are, in themselves, relatively easy to use and friendly enough to be tried by most teachers with just a little training. Adopting or adapting a useful, effective metadata set requires institutional effort and involvement. Learning materials preparation is hard work; creating sharable learning objects requires some additional effort, but reusability makes the task worth. Hopefully, as the number of available learning objects increases, the benefits of standardization will become apparent, favoring a more widespread use of eLearning on a cost effective basis.

References