LECTURE OBJECT: AN ARCHITECTURE FOR ARCHIVING LECTURES ON THE WEB

GIOSUE VITAGLIONE
CERN – University of Naples
E-mail: gio@qos.net

NORA BOUSDIRA
CERN – Ecole Nouvelle d’Ingénieurs en Communication
E-mail: bousdira@ely.enic.fr

STEVEN GOLDFARB
University of Michigan
E-mail: Steven.Goldfarb@cern.ch

HOMER A. NEAL
University of Michigan
E-mail: haneal@umich.edu

CHARLES SEVERANCE
University of Michigan
E-mail: csev@umich.edu

MICK STORR
CERN Technical Training
E-mail: Mick.Storr@cern.ch

Abstract

A new software architectural model for the archival of slide-based presentations on the Internet is proposed. This architecture is based on the concept of the “lecture object,” a persistent format, independent of the lecture production and viewing technology. The work has been undertaken in the context of the Web Lecture Archive Project, a collaboration of the CERN HR Division Training and Development group and the University of Michigan, Ann Arbor. To date some 250 lectures have been archived and are viewable worldwide using standard Web browsers and freely available video player software.

1 Introduction

Traditional lectures and seminars follow a linear pattern in which the lecturer prepares a presentation and delivers it, often accompanied by visual support material. The delivery mechanism can vary in style, with the lecturer using different techniques for displaying the visual support material, e.g. overhead projector, computer slide projection or a blackboard. Questions may be taken during the presentation, at the end, or not at all.
In each case, students must rely on their notes and/or copies of the support material to recall the key points of the lecture at a later date.

Having access to some form of reproduction of the original lecture, however, can greatly facilitate the learning process. Such a reproduction can exist in a variety of media, including audio or video recordings. Unfortunately, the dissemination of the material on audio/video tapes is cumbersome, thereby limiting access.

Recent technological developments based on the accessibility of the internet and the widespread utilization of the World-Wide Web, however, have overcome this difficulty. Audio and video recordings can be uploaded to media server facilities, making them easily accessible to a world-wide audience and requiring only standard web browsers and (often) freely available software for playback. Several software packages now exist with the ability to synchronize the audio and video recordings with the visual support material to generate an excellent reproduction of the original lecture, while requiring a small enough bandwidth to be viewed with most standard modems.

At CERN, the European Laboratory for Particle Physics, more than 250 lectures and seminars have been archived in this way by the Web Lecture Archive Project (WLAP) team and are viewable at http://www.cern.ch/wlap

This paper focuses on issues related to the archiving of lectures and proposes an architectural model for rendering the process independent of the production and viewing tools, such that the archive may be made persistent over a number of technological cycles.

2 Web Lecture Archive Project

The project started in 1999 as a pilot-project [1] funded by the U.S. National Science Foundation and the University of Michigan (UM) to examine the feasibility of the Sync-O-Matic 3000 application [2] to record and archive lectures in a variety of situations. The primary motivation was the need for large collaborations, such as the ATLAS experiment at CERN [3], to improve communication in a globally distributed environment.

Following the success of the original pilot project [4], a collaboration was formed between the CERN HR Division Training and Development group, the UM ATLAS Group and the UM Media Union, supported by CERN IT Division. The aim of the project is to demonstrate the feasibility and usefulness of archiving lectures, seminars, tutorials, training sessions and plenary sessions of collaboration meetings.

CERN has been chosen as host for this project because of its rich educational seminar program and because it provides experimental facilities to physicists from all over the world, sharing the need to communicate and disseminate information to a broadly distributed audience. In fact, the motivations are quite similar to those originally driving researchers at CERN to develop the World-Wide Web.

Today’s large particle physics experiments attract thousands of physicists, the majority of whom cannot be present at CERN 100% of the time, and who therefore need a common tool for viewing these events off-line. In addition, many of the institutes are separated by several time zones from CERN, making live participation through video or phone conferencing often difficult. Finally, although the archives are primarily targeted at the High Energy Physics community, some lectures and seminars are of wider interest and can play an important role in establishing a link between the working scientist and the scientifically curious members of the general public.
Table 1 lists the current content of the WLAP archive showing a wide variety of events that have been published to date.

<table>
<thead>
<tr>
<th>Number of lectures</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSL1999</td>
<td>Summer Student Lectures 1999</td>
</tr>
<tr>
<td>ATLAS</td>
<td>ATLAS Plenary Sessions Meetings</td>
</tr>
<tr>
<td>ATLAS</td>
<td>ATLAS Software Engineering Workshop</td>
</tr>
<tr>
<td>ATLAS</td>
<td>ATLAS Software Tutorial – Shared Libraries</td>
</tr>
<tr>
<td>ATLAS</td>
<td>ATLAS DAQ-1 Back-end Subsystem Training</td>
</tr>
<tr>
<td>ATLAS</td>
<td>XML Detector Description Workshop</td>
</tr>
<tr>
<td>GEANT4</td>
<td>GEANT4 software tutorial</td>
</tr>
<tr>
<td>NOBEL</td>
<td>Nobel Prize Talk – M.Veltman</td>
</tr>
<tr>
<td>SSL2000</td>
<td>Summer Student Lectures 2000</td>
</tr>
<tr>
<td>ACAD2000</td>
<td>Academic Training Program Lectures 2000/2001</td>
</tr>
<tr>
<td>TECHTRAIN</td>
<td>C++ for Particle Physicists - P.Kunz</td>
</tr>
<tr>
<td>TECHTRAIN</td>
<td>Technical Training seminars</td>
</tr>
</tbody>
</table>

2.1 Archiving issues

The requirements for a large archive of Web-Lectures, such as the CERN Archive, are:
1. The archive must be persistent, making the content available over a long period.
2. There should be a low production workload for creating a Web-Lecture.
3. Lectures should be easily editable, to fix mistakes or to update existing material.
4. Lectures should be easily indexed, allowing efficient search and retrieval capabilities

Specifically, to satisfy point 1, the requirements are:
- No proprietary technologies, as dependence on private software companies can lead to large licensing fees, restricting the access.
- Platform independence, also necessary for a wide audience.
- Independence of external technological cycles. The archive must use technologies that can be easily converted in the future, as software development can be very rapid.

3 Lecture Object

3.1 General Architecture

3.1.1 Two phase model

A natural approach for producing a web-lecture is in two steps: Capturing and Publishing. The first part grabs by special hardware/software all the data (audio, video, slides, etc…) from the real lecture. Publishing is the transformation of this material into a format viewable on the web. Usually the Web-Lecture is also automatically uploaded on the server.
This approach is quite easy to understand and to use, and reflects very closely the way a video recorder works. During the recording the audio/video signal is “captured” by some electronic devices, and it is stored (“published”) on a tape. After that, everything is ready to be watched from the tape. On the tape you have the format usable directly by a “video player”. In this case, the format of the archive is the same format used for the presentation.

![Figure 3-1 Two steps process to create a web lecture](image)

This approach has some limitations if used for web-lectures with the currently available technologies. The main problem being that it is difficult to choose a format that is good for both presentation and archiving. In fact, there are many presentation formats available today and most of them are proprietary. The current open standards do not yet propose all the features of the private software. (Compare Flash4 and RealPix against HTML and SMIL 1.0). In this field the standards need to follow the technologies proposed by the companies, as the industry is in an early stage of development.

Furthermore the presentation format must contain information regarding the layout, which can depend on the viewing platform. For example, viewing the same lecture with a desktop PC or with a PDA (Personal Digital Assistant) with a small colour screen will require different presentation formats. It is thus desirable to propose an open standard that defines an intermediate format for archiving web-lectures. This format needs to be, as far as possible, independent from proprietary standards, and convertible into different presentation formats.

3.1.2 Lecture Object Architecture

This proposal is based on a three-phase process, instead of two.

- **Capturing** - The audio, video and visual supports used by the speaker are captured by microphones, cameras, electronic boards and computers. Some software and hardware control the recording of the timing. Material related to the lecture, such as notes and documents is also collected.

- **Archiving** – All the information captured is archived on one or more servers. This information consists of sets of data and meta-data called here Lecture
Object Architecture (or sometimes just Lecture Object). The Lecture Object is proposed to be an open standard, not proprietary, to be used with different tools for the production of web-lectures. In this way a large variety of capture tools can produce equivalent Lecture Objects.

- Delivering – All the repositories of Lecture Objects can be used by tools that understand this standard for indexing and retrieving data. So, the web-lectures can be watched with a variety of tools and formats.

![Figure 3-2 Lecture Object, three-phase process.](image)

The Figure 3-3 illustrates the schema of a capturing tool. It creates a Lecture Object, compliant to the standard, independent of how it has been captured.

An important element in the architecture is the Broker, which acts as intermediary through a web server between clients and resource servers in the retrieving operations.

The service architecture is illustrated in figure 3-4.
A client, via the web, requests from the broker a specific web-lecture. The role of the broker is to find the corresponding Lecture Object and all of the components associated with it (video, slides, scribbling, etc…). Once the broker completes its job, the material is transformed into a presentation format, usable by the client. This transformation can be done either server-side or client-side, depending on the characteristics of the format and the server’s configuration. In addition, the broker can implement special optimization politics, so that it assigns a specific resource, chosen in a set of copies and versions on different servers, to balance server load and network traffic, thus improving performance.

3.1.3 Transformations

An application must be provided to perform the transformation from Lecture Object to presentation format. A prototype of such an application has been developed to transform a Lecture Object into SMIL, the standard defined by W3C (World Wide Web Consortium) for Synchronized Multimedia Integration Language. In a similar manner, one could transform to other presentation formats. This is possible because the Lecture Object contains the information necessary to reconstruct the events of the original lecture. Information, such as presentation layout, can be chosen by the broker, depending on the preferences and the capabilities expressed by the client.
3.2 Meta-data architecture

Meta-data are data which describe data. As such, meta-data can be used to archive information about a lecture, part of it or a set of lectures. For example to the object: “Lecture” can be assigned the meta-data: “Author”, and so on.

3.2.1 Existing standards

Some standards are available for describing learning material. We do not discuss them in detail here, but present some general considerations.

The standards related to this work are:

- IEEE LOM (Learning Object Meta-data)
- IMS (Instructional Management System)
- Gestalt (Getting Educational Systems Talking Across Leading-Edge Technologies)
- Ariadne (Alliance of Remote Instructional Authoring and Distribution Networks)

These are all meta-data architectures designed to describe the characteristics of generic Learning Objects and are all based on the first one (IEEE LOM).

The approach, common to all, is to define a single meta-data structure to describe whatever learning object we use. This is a practical solution for programmers developing software to handle the learning resources, as it handles all resources as instances of the same class. One need not be concerned with the classification of all objects into the variety of possible scenarios.

On the other hand, this very general approach can be inefficient in some cases, making it difficult to interpret the semantics of a field of the meta-data structure for a specific case. For this reason, the documentation of the standard must be very detailed regarding the meaning of the fields and their correct usage. Otherwise, we risk having a very rich archive of misunderstandings.

Also, if the structure is general enough to describe all the cases of learning objects, it can be difficult to determine the desired level of abstraction. The archivist can decide the granularity of the meta-data description, but this must be consistent in the whole archive and with archives of the same kind to allow interoperability. It would be useful, for the most common cases, to give implementation examples. These can be used as references to show (and to agree) which granularity and abstraction level should be used in each case. It would provide for a simpler interoperability between similar systems.

Furthermore, the description must be extendible. Adopting one single structure for all the possible learning objects means that there will be cases where one would like to add a specific field to describe a specific propriety of our objects not present in the general structure.

3.2.2 Lecture Object Meta-data

The proposed architecture is focused on describing web-lectures, which are primarily slide-based. The information about the structure of a lecture is considered as data, and not meta-data. This includes the sequence of the events (start talking, show slide #3, etc…) occurring during the lectures. In this case, the lecture may be treated as an object and considered to be data.

Also associated with the lecture are data, such as the video file, the slides, information pertaining to the actions of the speaker, for example pointing to slides or parts...
of slides, etc. All these are considered as data. In addition, there are other entities related to the lecture, for example the abstract, keywords, educational pre-requisites to the lecture, etc. These are considered to be meta-data of the lecture because this is information pertaining to another object, and not strictly standalone information with meaning on their own.

Note that the distinction between data and meta-data depends strongly on the context. In a word processor document, for example, the text of the abstract, if it is included in the document, is data and not meta-data. There is often no clear distinction.

Figure 3-6  Data and meta-data structure.

4 Implementation issues

Building an archive structure can be difficult, as there are many technical aspects to be taken into account. It is always good idea to have an understanding of the domains involved, that is, about what you’re going to archive. Defining a standard, however can be even more difficult, as one must consider the general case and to think in abstract terms. Rather than attack the general issues here, we propose to discuss a specific case, admitting the need for eventual extensions of the ideas presented.

4.1 Lecture description in XML

We have chosen to implement the lecture object using XML (eXtensible Markup Language)[5]. The main reasons being that XML is a simple, straightforward standard for structuring data, and that it is (as advertised) extensible.

We model the lecture as an object (in the sense of an XML “ENTITY”). That is, the Lecture Object represents a lecture. The lecture is then composed of a set of structured data describing the events of the actual lecture. The whole set of data and meta-data will
be called the Lecture Object Architecture (sometimes simply referred to as the Lecture Object). We then attempt to archive the entities as they exist in reality.

The Lecture Object is defined simply by describing the sequence of events of a lecture. For example: lecture begins; speaker starts talking; at the same time, the first slide is projected; after 2 minutes and 13 seconds, the second slide is presented; meanwhile, the speaker keeps talking; and so on, until the end of the lecture. In addition, one might add details about where our eyes were focused or what the cameras were recording, being careful to record the information necessary to reproduce the event. An example would look like the following:

```xml
title="Welcome to WLAP" region="speaker-face"
src="rtsp://webcast.cern.ch:5540/giosue.rm"
    seq
title="Sequence of slides" region="slide"
    slide title="First" type="image/gif" begin="00:00:00"
        region="slide" src="http://webcast.cern.ch/img001.gif"
    slide title="Second" type="image/jpeg" begin="00:00:12"
        region="slide" src="http://webcast.cern.ch/img002.jpg"
</seq>
</par>
</lecture>
```

The tags `<par>` and `<seq>` define the temporal order. Each element included in `<par>` happened at the same time. Each element in `<seq>` happened in sequence, implicitly defining an ordered list. The attribute `begin` gives the information required about when that slide appears. In our case there is a speaker talking and, at the same time, a sequence of slides.

While the attribute `region` may appear suspiciously to be an application-specific direction, it is meant to provide information pertaining to a region in the room where the lecture has been given, not a region on the screen of a PC. So in the room there was a place where the slides were projected and we called that region: “slide”. Another place was the face of the speaker. In this case, the speaker location is mobile as she or he may walk about the room, however the view of the speaker qualifies as a region, as our eyes typically follow the speaker. In this manner, one can describe where an object was placed in the room of the real lecture.

There are indeed many possible variations to this scheme. For example, the attributes `type` and `title` may be considered as meta-data pertaining to a single slide, and not data of the lecture. However, we have chosen the slide information to be at the primary level of data granularity, and have thus embedded the slide meta-data in the Lecture Object. One could also choose to define separately a slide object (with its slide meta-data), referencing it from the Lecture Object. As with any language, there are many possible variations in syntax.

### 4.2 Meta-data representation, RDF

Rather than creating our own representation for the meta-data in our XML model, we have chosen to comply with the RDF (Resource Description Framework)
recommendation of the W3C for writing meta-data. It’s a general standard for the representation of meta-data. This provides a unique context for the meta-data.

In RDF there are statements, formed by a subject (duck) a predicate (leg colour) and the object (white). This statement can be represented by a triple \{s, p, o\} and be used to describe a property of the subject. Using the namespace mechanism of XML, an example RDF statement would be:

```xml
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:a="http://www.animals.net/schema/parts#" />
<Description about="http://www.lakes.org/one/duckID1224">
    <a:legsColor> white </a:legsColor>
</Description>
</rdf:RDF>
```

RDF can thus be easily included in the XML representation, with small variations to allow for less typing. The information is recorded, using a standard API, and it is unequivocal. As with any useful standard, an important advantage for adopting it is that there is an entire world of developers working with it, and building tools that interpret it.

The IEEE LOM structure is divided into 9 categories: General, Lifecycle, Metadata, Technical, Educational, Rights, Relation, Annotation and Classification. We do not describe them here in detail, but note that each one is used to describe an aspect of the Learning Object. All the fields are not mandatory, we use just those which are necessary to describe the lecture or its component. The following is an example description in RDF of a video:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
    xmlns:ims='http://www.imsproject.org/metadata/
    xmlns:graph="http://home.cern.ch/giosue/2000/06/SizeOnScreen#" />
<rdfs:Description rdf:about="pnm://webcast.cern.ch/cannettes.rm">
    <ims:technical rdf:parseType="Resource">
        <format application/vnd.rn-realmedia</format>
        <size>1032353</size>
        <location pnm://webcast.cern.ch/cannettes.rm</location>
        <graph:width>160</graph:width>
        <graph:height>120</graph:height>
        <requirements rdf:parseType="Resource">
            <type rdf:parseType="Resource">
                <string>Player</string>
                <lang>en</lang>
            </type>
            <name rdf:parseType="Resource">
                <string>RealPlayer</string>
                <lang>en</lang>
            </name>
        </requirements>
        <duration rdf:parseType="Resource">
            <datetime>0000-00-01:20</datetime>
        </duration>
    </ims:technical>
</rdfs:Description>
</rdf:RDF>
```
Note that three different name spaces are defined above. The second is used for the IMS meta-data structure and the third is to extend IMS to include the \texttt{width} and \texttt{height} fields to describe the size of the video, fields not present in the normal structure.

4.3 Capturing tools

4.3.1 PowerPoint macro

A VBA macro has been developed to produce a lecture object from a PowerPoint presentation. It captures the time during the presentation, and creates automatically a \textit{Lecture Object} with information pertaining to the slides’ titles and the timing. This macro can automatically export the slides in gif and/or jpeg, optimizing disk usage. The generated files are automatically uploaded to the server.

4.3.2 PDF viewer

This is a tool to show Acrobat files that produce a Lecture Object. It’s similar to the PowerPoint VBA macro described above.

4.4 Transformations

A set of server-side and client-side transformations have been developed to demonstrate the ability to dynamically transform a \textit{Lecture Object} into various presentation formats.

4.4.1 Server-side transformations

4.4.1.1 PHP script lo2smil

This script has been developed using the PHP scripting language on a Linux server. One version of the software uses SAX \cite{8} to parse the \textit{Lecture Object} and to retrieve all the information about the lecture. This script sends to the client an html page containing a form and some javascript code. This code fills the form’s fields with information pertaining to the screen size and returns the form to the server. The server can now use this information to create SMIL \cite{10} code with a proper layout, maximizing the size of the slide region. It then sends this information to the client.

Another version has been developed using the DOM \cite{9} interface, for obtaining SMIL code editing the \textit{Lecture Object} code.

4.4.1.2 Lo2print

A prototype script has been written to print the slides archived in a lecture. This PHP script is invoked, passing as parameter the location of the lecture object. The script takes the URL of the slides and creates a single HTML file, printable in a single operation.
4.4.2 Client-side transformations

4.4.2.1 Lect-o-matic Java Applet

This is a more complex transformation that does not use SMIL. Rather, the transformation is performed by a Java Applet loaded in an HTML frame, on the web browser of the client.

The Applet loads the Lecture Object xml file and parses it, retrieving its information. It then talks with the RealPlayer [11] plug-in and starts playing the video. Via a callback function, the applet is informed by the plug-in about the time position of the video clip. The applet shows the right slides at the right moment.

This solution is very flexible and powerful. The applet could provide many features, including jumping back and forward through slides and video. In addition, many graphical effects are possible.

5 Further development

There is much room for the development of applications designed for both input and output conversion to and from the Lecture Object. This includes the tools to capture high quality videos with minimal effort, as well as tools to make automatic media conversions, satisfying the growing variety of viewing applications. Our research currently is concentrated on the development the lecture object architecture, taking into consideration the latest available technologies.

6 Conclusions

We have presented a proposal for a new architectural model for the archival of lectures on the Internet. This architecture is based on the Lecture Object, a format to archive lectures which is independent of how the lecture is recorded and viewed. Prototype software has been developed as proof of concept for capture and transformation of the Lecture Object. The full implementation of this architecture will facilitate the building of standard archives of web-lectures, available in different presentation formats, with enhanced search capabilities and useable over a long period of time.

7 Acknowledgements

We would like to thank all the people who contributed in the development of the Lecture Object Architecture and who supported the Web Lecture Archive Project. In particular the CERN IT division for hosting the webcast server and for providing expertise and support. We also want to acknowledge that the Sync-O-Matic 3000 software was developed with the support of Michigan State University and that Clipboard 2000 was developed with the support of the University of Michigan.
8 References

http://webcast.cern.ch/Projects/WebLectureArchive/Project/Proposal99.pdf
http://www.syncomat.com
[3] The ATLAS experiment is being constructed by 1850 collaborators in 150 institutes around the world.
http://atlasinfo.cern.ch/Atlas/Welcome.html
http://webcast.cern.ch/Projects/WebLectureArchive/Project/Summary99.pdf

9 Bibliography

3. CALIBER-NET: Quality in European Open and Distance Learning http://www.eadtu.nl/activities/projects/CALIBERNET/default.htm
7. Distance Education Clearinghouse, University of Wisconsin, http://www.uwex.edu/disted/
9. DUNE: Distance Education Network of Europe http://www.eadtu.nl/activities/projects/DUNE/default.htm
12. IMS meta-data extensions, http://www.imsglobal.org/metadata/metadatafaq2.html#extending
16. Quality of Service networking, QoS.Net Project, http://www.qos.net/
22. Video Codec WareHouse, http://www.geocities.com/codecwarehouse