LECTURNITY: File format

History of the old

Author: Ulrich Kuhn

# Current format

There are a number of files which store information about a LECTURNITY recording document and possible changes.

Especially there is the .lrd file which references a number of other files and together they form the recording document.

Additionally there is the .lep file which stores edit information on one or a combination of .lrd files.

## .lrd

Here is a list of all files (formats) used for a recording. For illustration it also shows a rough estimate of the year that this format was first used.

* .lrd: References .obj, .evq. .lad, .lmd, .avi. *2000*
* .obj: A SGML file storing objects. They are implicitly numbered: The first one is “0” and so on. They have no time information. *1992*
* .evq: Text file with “events”: Every line shows a discrete point in time. There is only a line when something changes on the whiteboard. It lists object numbers. *1992*
* .lad: Encrypted form of a .wav file. 2002
* .lmd: Metadata for pages: Titles, start and end point, keywords. This data is not included in the .evq file. Only the page start and end times can be deduced from there. *2004*
* .avi: Standard format for normal video and clips. *2000* (1980)

.lmd, .obj and .lrd have a SGML format (similar to XML).

Such an .lrd recording file describes one single timeline from 0..end.

## .lep *2005*

Is the intermediate format used by the Editor. Basically it lists the cut information performed there. Here is what it contains:

* An audio stream that can be comprised of different parts of different .lad files.
* A video stream that can be comprised of different parts of different .avi files.
* A clip stream that can contain several clips which in turn can come from different parts of different other clips (.avi).
* A whiteboard stream that can be comprised of different parts of different .evq, .obj and .lmd files
* A slides/metadata stream with the page metadata for the used pages (at first redundant to the whiteboard stream; can be changed)
* A marks stream listing all stop and jump marks. *2007*
* An interaction stream with all the objects used for interactions. These are the same objects as in .obj only that these here have timestamps for beginning and end. *2007*

So from a file perspective the .lep would be more natural for storing additional streams like for example a second audio stream.

# Problems

There are two basic problems with the current files:

1. There is much redundancy: .lep is a replication of the .lrd data only that it has cut/time information.
2. All of the files come from very different times and target different functions. This makes interoperability cumbersome.

The .lep is a “good” example for the general problem: First it contains cut information and only references other files. And second it has plain objects **with** time information.

## Translation between formats

1. A third problem is that the Publisher (all output formats) only can read .lrd files. Thus there must be an export step (flattening of cut information) which must occur before every publishing. This requires a recompression of video data and is therefore time consuming and introduces quality loss.

# Future format

Any changes or solutions for these problems might take advantage from the following fact:

* For new versions of LECTURNITY it is ok if only Flash is supported as an output format. (This was different in the past.)
* This does especially mean that no “legacy” output is needed or useful: It is not useful to output any old (.lrd) information for the old not-supported formats. This makes new code for new functionality more complicated and as the old formats lack the support for new features it would bring no advantages.

## Possible solution

There should be one file format which replaces .lrd and .lep. It might be based loosely on the existing .lep but it should contain all (textual) information related to a recording and its changes.

This means:

* Recording writes this format
* Editor reads and writes this format
* Publisher (Flash) reads this format
* There is a legacy import which converts from .lrd or .lep to the new format.
* .avi and .lad are still used
* The new Flash code must then be able to read and combine different (parts of) .avi and .lad files before converting them to the Flash media files. This is similar/the same as the existing preview/export code in the Editor.

The main problem with this solution (or any solution) besides the definition of the format itself: The Publisher is still mostly Java. The preview code in the Editor is C++ (DirectShow). These must meet or be adapted in some way.