



Project no. FP6-028038

Palette

Pedagogically sustained Adaptive LEarning Through the exploitation of Tacit and Explicit knowledge

Instrument: Integrated Project

Thematic Priority: Technology-enhanced learning

D.INF.07 – Final version of the information services

Due date of deliverable: 30 November 2008

Actual submission date: 7 January 2009

Start date of project: 1 February 2006

Duration: 36 months

Organisation name of lead contractor for this deliverable: INRIA-Grenoble

Project co-funded by the European Commission within the Sixth Framework Programme		
Dissemination Level		
R	PUBLIC	PU

Keyword List: Information services, Authoring tools, Templates, Multimedia, Reuse tools

Responsible Partner: UNIFR

MODIFICATION CONTROL			
Version	Date	Status	Modifications made by
1	20/11	1 st Draft	Irène Vatton, Vincent Quint
2	26/11	2 ^{sd} Draft	Irène Vatton, Vincent Quint, Yan Mikac, Cécile Roisin, Aida Boukottaya, Micaël Paquier.
3	3/12	3 rd Draft	Irène Vatton, Vincent Quint, Yan Mikac, Cécile Roisin, Aida Boukottaya, Micaël Paquier, Sami Miniaoui.
4	8/12	Version sent to reviewers	Aida Boukottaya
5	16/12	Integration of reviewers feedback	Vincent Quint, Cécile Roisin, Aida Boukottaya,
6	17/12	Sent to SC for approval	Aida Boukottaya

Deliverable manager

- Aida Boukottaya, UNIFR

List of Contributors

- Aida Boukottaya, UNIFR
- Cécile Roisin, INRIA-Grenoble
- Irène Vatton, INRIA-Grenoble
- Micaël Paquier, EPFL
- Sami Miniaoui, UNIFR
- Vincent Quint, INRIA-Grenoble
- Yan Mikac, INRIA-Grenoble
- Vincent Quint, INRIA-Grenoble

List of Evaluators

- Amaury Daele, UNIFR
- Christine Vanoirbeek, EPFL

Summary

This deliverable describes the final releases of information services. It details the changes made from the latest releases for authoring tools: Amaya and LimSee3, and structured document reuse services: Template-driven structuring, template-driven evolution and template-driven eXtraction. Technical implementations are detailed as well as the use of these tools within Palette CoPs.

1	Introduction.....	5
2	Authoring tools.....	7
2.1	Amaya.....	7
2.1.1	Technical evolution.....	7
2.1.1.1	The XTiger language and its visualisation	7
2.1.1.2	Editing with templates	8
2.1.1.3	Creating templates	8
2.1.1.4	Editing graphics	10
2.1.1.5	Annotating documents in RDFa	12
2.1.1.6	Sharing documents.....	13
2.1.2	Amaya in the CoPs	13
2.1.2.1	Comparing training approaches in TICEF and TICFA.....	13
2.1.2.2	Discussing teaching practices in Did@cTIC	14
2.1.2.3	Conclusion	16
2.2	LimSee3.....	17
2.2.1	Technical evolution.....	17
2.2.1.1	Document rendering	17
2.2.1.2	Exporting and packaging	19
2.2.1.3	Document publication on Palette repositories	20
2.2.1.4	New document template – annotated audio	20
2.2.1.5	Library enhancements.....	21
2.2.1.6	User interface and user preferences	22
2.2.1.7	Off-line installation.....	23
2.2.1.8	Localization	24
2.2.1.9	Conclusion	24
2.2.2	Using LimSee3 in Integrated Services of PALETTE	24
2.2.2.1	ePrep multimedia pedagogical content production.....	24
2.2.2.2	Did@cTIC annotation tool for arousing socio-cognitive conflicts.....	26
2.2.2.3	Conclusion	26
3	Structured documents reuse tools	27
3.1	Technical evolution	27
3.1.1	DocReuse unified interface	27
3.1.2	2.2 The Template-driven structuring service	31
3.1.3	2.3 The Template-driven evolution service	32
3.1.3.1	Template evolution element primitives	33

3.1.3.2	Template evolution types primitives.....	36
3.1.3.3	User interface.....	36
3.1.4	The Template-driven eXtraction service	37
3.2	DocReuse in the CoPs	41
3.2.1	Didactic Meeting Capitalization Situation.....	41
3.2.2	Didactic Meeting Reports Evolution	41
3.2.3	Didactic Meeting Reports Synthesis.....	41
3.2.4	Usability test of DocReuse Services	42
3.2.4.1	The usability test of the DocReuse Unified interface	42
3.2.4.2	The usability test of the Evolution service.....	43
3.2.4.3	The usability test of the Extraction service.....	43
3.3	Conclusion.....	44
4	Conclusions	45
5	References.....	46
6	Annexes	47

1 Introduction

Workpackage 2 of Palette aims at providing services for information production and reuse in communities of practice (CoP). The objective of WP2 is to create the efficient services and tools that are necessary for average CoP members to easily produce, use and reuse multimedia documents. More precisely, efforts in this workpackage focus on two services, one for multimedia content authoring, the other for structured documents reuse.

Regarding multimedia content authoring, the objective is to provide users with a flexible, adaptable environment for efficiently producing different types of multimedia documents, specifically targeted for use in CoPs. This goal is achieved with two complementary open source tools, Amaya and LimSee3. Given the wide range of issues to be addressed, developing a single piece of software for addressing all the needs would be risky. Therefore, each tool focuses on different issues.

Amaya is a web editor designed to ease editing, publishing, and sharing of rich web pages containing discrete media objects (text, still images, vector graphics, mathematical expressions), as well as style information. It focuses on the use of the web and its specific features: links between resources, on-line distributed content, adaptable and accessible documents, separation of style from content, etc.

LimSee3 aims at multimedia authoring in general. It is designed to ease the manipulation of time-based multimedia documents. It focuses on the time aspect of documents. It handles continuous media objects such as video and sound. It allows users to manipulate synchronization of document components, transitions between media objects, animations, etc. LimSee3 provides broader authoring perspectives than most existing tools which are dedicated to particular languages and players. It is built upon a document model called "*LimSee3 document model*" (see [1]) that allows full author-oriented template-based services that hide the complexity of multimedia formats (such as SMIL) to the author.

Although the focus is different for each tool, there is some commonality between them. Both are based on the same principle of structured documents. Both feature a graphical user interface and allow authors to interact with a graphical, familiar representation of documents, even if a more abstract structure is handled behind the scene. Both use open document formats based on XML, namely those created by the World Wide Web Consortium (W3C) for multimedia web resources: XHTML for structured text, SVG for vector graphics, MathML for mathematical expressions, SMIL for synchronized multimedia, and CSS for style sheets. This allows content created with both authoring tools to be integrated smoothly in the web environment used by CoPs.

Amaya existed as an open source project before the beginning of Palette. In Palette, the objective was to tailor it to the specific needs of CoPs and to introduce the notion of template as a help for producing well organized, reusable documents.

The first Palette release of Amaya (version 9.54, February 2007), presented in deliverable D.INF.02, introduced templates as a guide to edit web documents. Only the core functionality of template-driven editing was available, but this was enough to allow CoP members to experiment with it and to provide feedback.

The second Palette release of Amaya (version 10) was presented in deliverable D.INF.05 in February 2008. The template-driven editing feature was completed and the user interface was redesigned following the usability analysis carried out by WP1.

LimSee3 was developed as an entirely new open source project in the context of Palette. Developing from scratch had the inconvenience of being lengthy, yet it made it easier to comply with the principle of participative design. This allowed developers to integrate user comments and requests along the road.

The first release of the tool (version 0.5) was made with deliverable D.INF.03 in April 2007. Its purpose was to demonstrate template-driven editing of documents, emphasizing its convenient and easy-of-use dimension. The implementation was not complete at that time but available functionalities were sufficient to make early experiments and to gather comments from users.

The second release of LimSee3 (version 1.0) was part of deliverable D.INF.05 in February 2008. It followed an intensive collaboration phase with the ePrep CoP: the main evolutions of version 1.0 have been obtained in a participatory design way. As a more technical contribution, the document model has also evolved to allow documents to be exported into different formats.

The latest version of LimSee3 (version 2.0) released with this deliverable provides a set of improvements that are derived both from the Palette usability analysis and from experiments done by CoP users. It introduces two new categories of features:

- several common tools that are permanently present, and are active or not depending on the current context or the user request: alignment tool, resizing tool, export and packaging tool (including access to Palette services such as SweetWiki and the Palette repository).
- specific tools to cover a new multimedia service, namely an annotated audio service. It is provided thanks to a new template that has been added to the set of available templates.

Regarding structured documents reuse tool, namely DocReuse, the main objective is to ease and automate as much as possible structured documents reuse and transformations. Structure transformation has been known as a tedious and error prone process. DocReuse is designed as a set of web-based applications to ease the reuse of structured documents and to answer CoPs needs as specified in the generic scenario “Reification of Knowledge” and described in WP5 deliverables D.IMP.04 and D.IMP.08.

DocReuse existed as a result of a Phd work [2] before the beginning of Palette. The first release of DocReuse (May 2007), presented in deliverable D.INF.04 described two main services: the template-driven restructuring service and the automatic matching and transformation service. The first one aims to enable CoPs members to generate structured documents by reusing their existing resources (mainly HTML pages). A first prototype was available to enable CoPs members (Did@ctic CoP) to begin experimentation and gather their feedback. The second service extended the result of the Phd work to deal with Amaya templates (expressed with XTiger language) as input instead of XML schemas. A very first prototype was available and deployed to Did@ctic CoP members through a set of use-cases. Several trainings have been organized and a working group composed of DocReuse developers and Did@ctic mediators was formed to analyse the usability of the service and fix required and useful functionalities for CoPs members.

The second Palette release of DocReuse was done on April 2008 and presented in deliverable D.INF.06. Two main results coming from the close collaboration with Did@ctic mediators were presented. The first result was to split the automatic matching and transformation service into two new services: the Template-driven evolution service and the Template-driven eXtraction service. The core matching algorithm was used to implement the first prototype of the Template-driven evolution service. The second result consists in a set of recommendations concerning the XTiger language. These recommendations were included in collaboration with the INRIA-Grenoble team to the XTiger specification.

The Final Palette version of DocReuse released with this deliverable provides improvements derived from CoPs members’ experimentation. Essentially, the main functionalities of DocReuse services were entirely implemented. The user-interfaces were homogenised with Amaya in order to offer CoP members a unique representation of templates. Moreover, a single database for the three DocReuse services was available in order to build an efficient memory of the CoP. Finally, a single entry to the three services was implemented in order to ease their access.

2 Authoring tools

2.1 Amaya

2.1.1 Technical evolution

Version 10.0 of Amaya was released on 26 February 2008, as part of deliverable D.INF.05. Since that time several technical developments were made to complete the implementation of key features and to address specific requirements expressed by CoPs, based on their day-to-day use of the tool. Most of these new developments are related to the XTiger templates that were introduced in Amaya specifically for the Palette project. Another important new feature was added to allow members of the ePrep CoP to produce web documents containing graphics. Two new developments were also initiated to ease document sharing between CoP members and for interoperability with other Palette services, namely the common repository and SweetWiki. The following sections provide more details about these developments, made available with Amaya version 11.0.

2.1.1.1 The XTiger language and its visualisation

The XTiger language described in D.INF.01 was initially designed to define templates that could help authors to write and structure documents according to a predefined model. As such, XTiger templates were primarily intended to be used in a document editing system and they were first implemented in Amaya for that purpose (see D.INF.02). Later on, the XTiger language was considered as a good means to help CoP members to restructure and reuse existing documents, and it was therefore introduced in the DocReuse tool (detailed in D.INF.06).

This new way of using XTiger required a few modifications to the language that were introduced in Amaya 11.0:

- The `option` element was deprecated and replaced by an attribute (called `option`) that can be associated with `use` elements when they are optional. For `repeat` elements, the same behaviour is achieved through the `minOccurs` attribute: value 0 indicates that the element is optional. This was done to simplify the syntax of the language, without losing any existing feature.
- The `label` attribute was redefined as being of type NMTOKEN, to offer more flexibility to users when providing a hint to XTiger elements.
- The `name` attribute was constrained to be unique in a template, to identify unambiguously each element.
- The `component` elements were constrained to appear only in the `head` element of a template, in order to clearly separate their definition (in the head) from their use (in the body).

In addition to these minor changes to the syntax and semantics of the language, the visual aspect of XTiger elements in Amaya was modified, with the goal of presenting these elements in exactly the same way in both Amaya and DocReuse. That way, CoP users feel more conformable and less confused when handling the same template in both tools. The same icons and the same colours are now used to display XTiger elements throughout the whole range of Palette tools:

- A `use` element is shown as a dashed blue box
- A `bag` element is shown as a dashed green box
- A `repeat` element is shown as a box with a `o` on the top left corner
- The `attribute` element is not displayed in the main view, as it only impacts the Attributes tool.

XTiger elements can also be seen in the structure view. In this view, they are displayed in gold, while the XHTML elements are in purple (Figure 1). XTiger elements are also shown in the Source view, but with no special colour.

Another way to understand the nesting of XTiger elements with XHTML elements is the selection path displayed in the status bar, at the bottom of the window (Figure 1). It displays the list of all elements in the structure of the document from the root element to the current selection. This list is updated every time the selection is changed. In the selection path, XTiger elements are displayed in gold and XHTML elements in black. For XTiger elements, it is not the name of the element that is displayed there (use, bag, repeat), as this would be ambiguous given the number of these elements in the structure, but their label, as declared in the template.

2.1.1.2 Editing with templates

The initial implementation of XTiger templates in Amaya supported only pure XHTML documents. But Amaya can also handle other XML document formats, such as MathML for mathematical expressions or SVG for structured graphics (see below), as well as generic XML documents. The support of XTiger was then extended, and templates can now be used with any XML format, including any combination of XML formats. This includes compound documents where different formats are mixed to represent web pages containing, for instance, mathematical equations and drawings in addition to structured text.

With this extension, XTiger templates can now ease the creation of complex structures that occur repeatedly in equations or drawings. All parts of a compound document can benefit from the help provided by templates. The introduction of XTiger elements within an XML structure induces some changes on document processing: the hierarchy of elements does not follow exactly the model expressed by the relevant schema and some procedures have to take that into account. In particular, the Amaya editor was updated for that purpose. It is now able to cope with the XTiger elements when editing a document with a template.

2.1.1.3 Creating templates

Amaya version 10.0 was able to edit XHTML documents following the hints provided by XTiger templates, but the templates themselves had to be created "by hand". Only expert users were able to do so. A major improvement of version 11.0 is the template editor, which allows any user to easily create and update templates.

Creating a template is done in two steps. First the user creates the skeleton of the type of document s/he wants to represent with the template. This skeleton is created in exactly the same way as any other document, with the usual features of the editor. Then, XTiger elements are added to the skeleton to indicate how real documents could be derived from there. The role of the template editor (see Figure 1) is to help users to create these XTiger elements. The basic principle is that the user selects a part of the skeleton and wraps it into an XTiger element. This can be done recursively, as XTiger elements can be nested. The parts of the skeleton that are left outside an XTiger element will remain unmodifiable in instance documents that will be produced later on with the template.

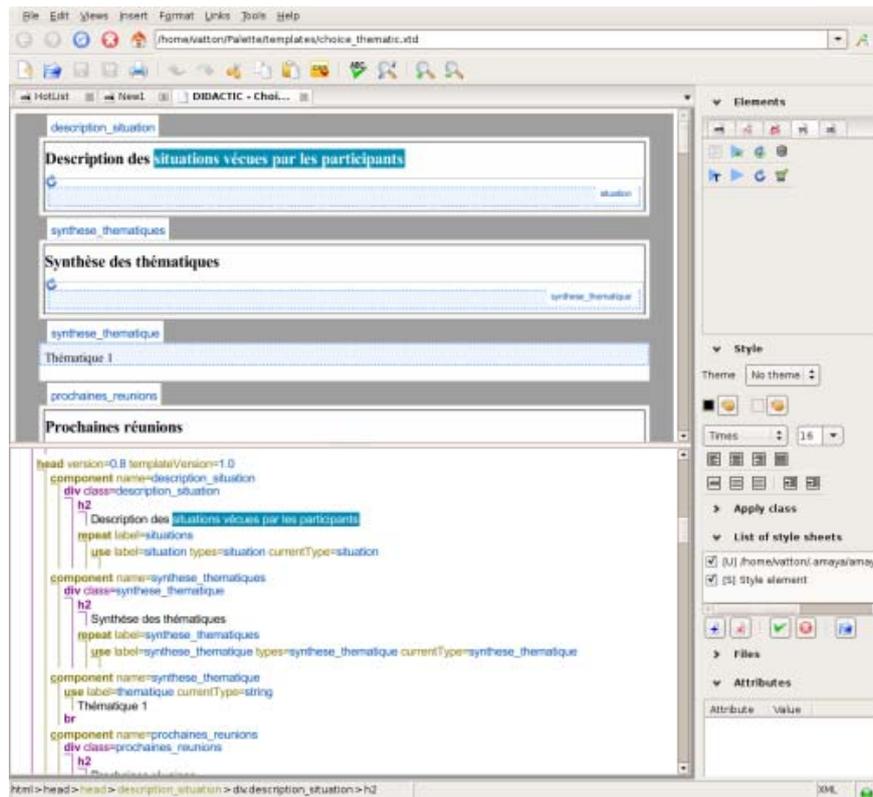


Figure 1: editing a template

XTiger elements are created through a new tab in the Elements tool (see Figure 1, top right) or through a menu accessed from the Tools menu. Both the Elements tool and the menu offer the same commands:

- *Create template from current document* turns the current document (the skeleton) into an XTiger template. This command creates an XTiger head element in the document and adds an XTiger namespace declaration.
- *Create and use a component* generates an XTiger component element that will be displayed in the head of the template document. The current content of the selection becomes the content of the new component.

A use element that allows one to insert that component is created at the current position of the document. The selection must be valid (a sequence of complete elements or a string). The command asks the user for the component name. That name must be unique in the template.

Note: The content of the component won't be edited in document instances. The user has to insert `use` and/or `bag` elements inside the component to define editable areas. These editable areas can be inserted before or after the creation of the component.

- *Create and repeat a component* generates an XTiger component element that will be displayed in the head of the template document, and a use element within a repeat element. That command is a shortcut for the two commands *Create and use a component* and *Repeat the current use*.
- *Create a new type union* generates an XTiger union element that defines a new type as a choice between basic types (number, string), elements from the target language (`div`, `h1`, `h2`, `p`, etc. for XHTML), or component elements. The current selection has no effect. The command asks the user for the union name. That name must be unique in the template.

- *Create a text area* generates an XTiger `use` element that contains a character string. The command asks the user for a label. That label can be chosen freely; it will appear in the selection path displayed in the status bar for identifying the `use` element. The selection must be a string which becomes the initial value of the new use element.
- *Use a component or an element* is used to generate a new XTiger `use` element.

If the selection is empty, a dialog box proposes the list of elements that can be inserted at that position and asks the user for the the label to be associated with the new use element. The use element is generated with an empty content.

If an element of the target language is selected, a use element that allows one to insert this element type is generated at that position. The selected element becomes the initial content of the new use element. The command only asks the user for a label.

- *Repeat the current use* is used to generate an XTiger `repeat` element whose content is the selected `use` element. The command asks the user for a label that will be used in the selection path displayed in the status bar.
- *Create a free editing box* generates an XTiger `bag` element. The command asks the user for a label that will be used in the selection path displayed in the status bar.

The introduction of the XTiger elements in the Element tool was an opportunity to reorganize this tool, which is also used for creating elements from the various languages handled by Amaya. The Element tool has now several tabs, one for each language (XHTML, MathML, SVG, XTiger, generic XML). This is a significant simplification of the user interface.

2.1.1.4 Editing graphics

In addition to an XHTML editor for handling text, Amaya version 10 offered an advanced feature for editing mathematical expressions (using the MathML language), and it was able to display vector graphics expressed in the SVG language. With its ability to handle compound documents through the XML namespace mechanism, Amaya 10 was then able to display scientific documents, such as those used in the ePrep community. But an important feature was missing regarding graphics: there was no way to create or edit graphic elements, which could only be displayed.

A graphics editing feature was then introduced in Amaya 11.0. The goal of this development was to address the needs of the ePrep CoP, rather than implementing the full SVG language. As a consequence, Amaya can not create all SVG elements and attributes, but it implements a subset of the language that is sufficient for the drawings that scientific authors include in their documents. Also, some predefined components may be created in a single click, to avoid users to repeatedly build piece by piece the most frequent shapes used in scientific illustrations.

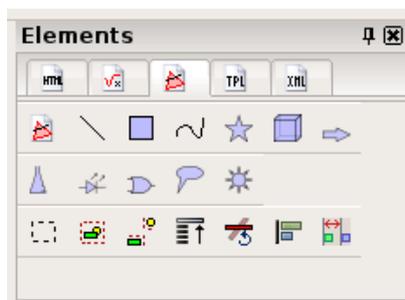


Figure 2: Elements tool for SVG

A new tab was added to the Elements tool for creating the basic SVG elements and these predefined components (Figure 2). The tool can create:

- connectors: line, simple or double arrow, zigzag.
- text or embedded HTML fragment (`foreignObject`) and image.
- basic shapes: square, rectangle (optionally with rounded corners), diamond, parallelogram, trapezium, triangles, circle, ellipse.
- polygon, polyline, open and closed Bézier curves.
- regular polygons: pentagon, hexagon, heptagon, octagon, various stars
- 3D objects: cone, cube, cylinder, parallelepiped, octahedron, pyramid
- special symbols for electronics, chemistry, etc. (Figure 3)

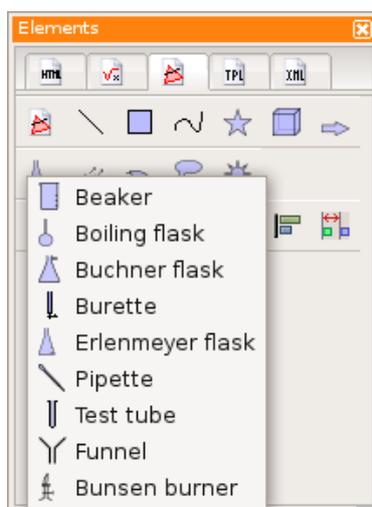


Figure 3: special symbols for chemistry

All these elements and components are created by clicking a button in the Elements tool and their position and size is indicated by direct interaction. Existing elements can be moved or resized in the same way.

Several transformations and other graphic operations can be applied interactively to existing elements (Figure 4). These include:

- selecting multiple elements
- grouping/ungrouping objects
- changing the level of objects, to control overlay: send to back, one level down, one level up, bring to front
- rotating objects.
- vertical and horizontal symmetry
- scaling, skewing
- aligning objects: left, centre, right, top, middle, bottom.
- distributing and spacing objects

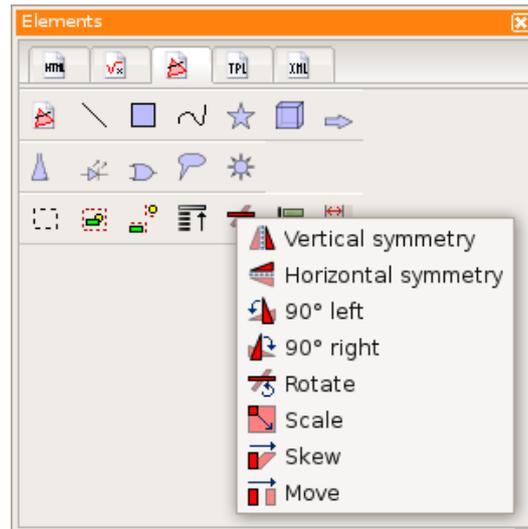


Figure 4: graphic operations

Finally, the style of all graphic objects can be set and changed at any time (Figure 5):

- colour: stroke and fill
- stroke width
- opacity/transparency



Figure 5: style palette for graphics

2.1.1.5 Annotating documents in RDFa

RDFa is a brand new recommendation from W3C (<http://www.w3.org/TR/rdfa-syntax/>). It is a specification for adding semantics in any markup language, in particular in XHTML. It is an efficient way to support semantics expressed as RDF data directly within documents.

RDFa is now supported by Amaya 11.0; the specific attributes (about, property, resource, datatype, instanceof) are allowed by Amaya in RDFa documents and they are saved when the document is modified. These attributes can also be edited through the usual user interface for attributes.

It is also possible to turn an XHTML document into an XHTML+RDFa document from the menu Tools / Change doctype. The required namespaces declarations are then generated, based on the preferences set by the user. A new tab was added to the Preference dialog for this purpose. The namespaces declarations on the html element can be edited through a specific dialog.

The availability of RDFa in documents produced by Amaya eases interoperability with Sweetwiki: various data required to post a document to Sweetwiki are extracted by Amaya from the RDFa information embedded in the document itself. This is a good example of the advantage of embedding

semantic information in a web document: the document may be used by several applications, and each of them can find securely the data it needs.

2.1.1.6 Sharing documents

For sharing documents within a CoP, Amaya users have several options. They can use the Common Repository, or a web server. Amaya can also be used to edit documents resulting from collaborative work on SweetWiki.

Thanks to the HTTP PUT method supported by both Amaya and the Palette Common Repository, a document being edited with Amaya can be saved on the repository in a single click. This operation is as simple as saving the document locally. Conversely, a document available on the repository can be downloaded in Amaya for further editing.

Some documents resulting from collaborative work on SweetWiki may need some editing work in the final step, and Amaya is a tool of choice for this task. The document must first be transferred from SweetWiki to Amaya, where it is edited and polished, and then it must be sent back to SweetWiki. The first operation is performed through a HTTP GET, which is supported both by Amaya and SweetWiki. The second operation is performed by a HTTP POST, as SweetWiki does not support PUT.

2.1.2 Amaya in the CoPs

With version 11.0 the template feature is fully available in Amaya. In particular, CoP members can now create templates that precisely fulfil their needs. Early experiments have been made with three CoPs to adjust the details of the template editor and make sure that real users can take advantage of it.

A few examples are given in this section to demonstrate how CoPs can design their own templates. In all cases the templates are defined to allow teachers or students to provide feedback on their experiences. When a teacher or a student creates a document based on such a template, s/he sees immediately where information has to be inserted and what kind of information is expected. As a result, the information they provide is better structured and can therefore be processed more securely. Also, as the template stimulates them to provide detailed information, more information is available in the documents they produce.

2.1.2.1 Comparing training approaches in TICEF and TICFA

The first example (Figure 6) is a template for the TICEF CoP. A similar template has also been created for the TICFA CoP. Both CoPs are groups of Master students in the Education Sciences domain. Students have to analyse different training approaches with two pedagogical models (one for TICEF, one for TICFA). The template contains areas where students have to insert information under the form of short character strings, like in a form (year, name, email address, all in blue on the figure). Other areas accept free content that can be organized with usual elements such as paragraphs, lists, etc. (green areas on the figure).

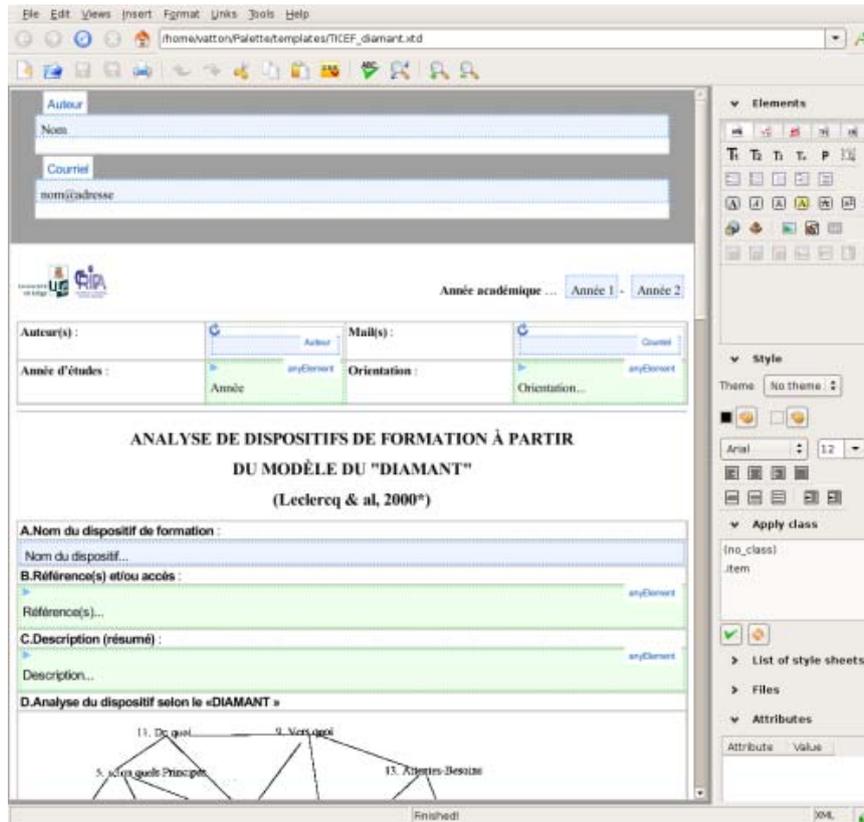


Figure 6: a template for the TICEF CoP

The template designer decided to format the document as a large table, because that format is very common for these students. A diagram is also added as part of the template, to help students to remember the pedagogical model they are working on. Clearly the table, the diagram, and other fixed parts of the document are only there to help the student and to create a pleasant context for his work. But that extra information is noise for the run on student works. Thanks to the templates, the important areas of document instances, those where students have inserted information, are clearly identified and can be automatically and safely extracted by a XSLT transformation or by the Palette DocReuse tool.

The final objective is to compare different training approaches with the same pedagogical model and compare the analysis of the same training approach with two different pedagogical models. That objective can be easily reached by using these templates and Palette tools.

2.1.2.2 Discussing teaching practices in Did@cTIC

The second example (Figure 7) is extracted from a set of templates created by the Did@cTIC CoP. These templates are used by a group of University teachers during the meetings where they discuss their teaching practice. The template of Figure 7 is used during the first meeting, to decide what thematic will be analyzed by the group during future meetings. The template is a support for taking minutes during that first meeting.

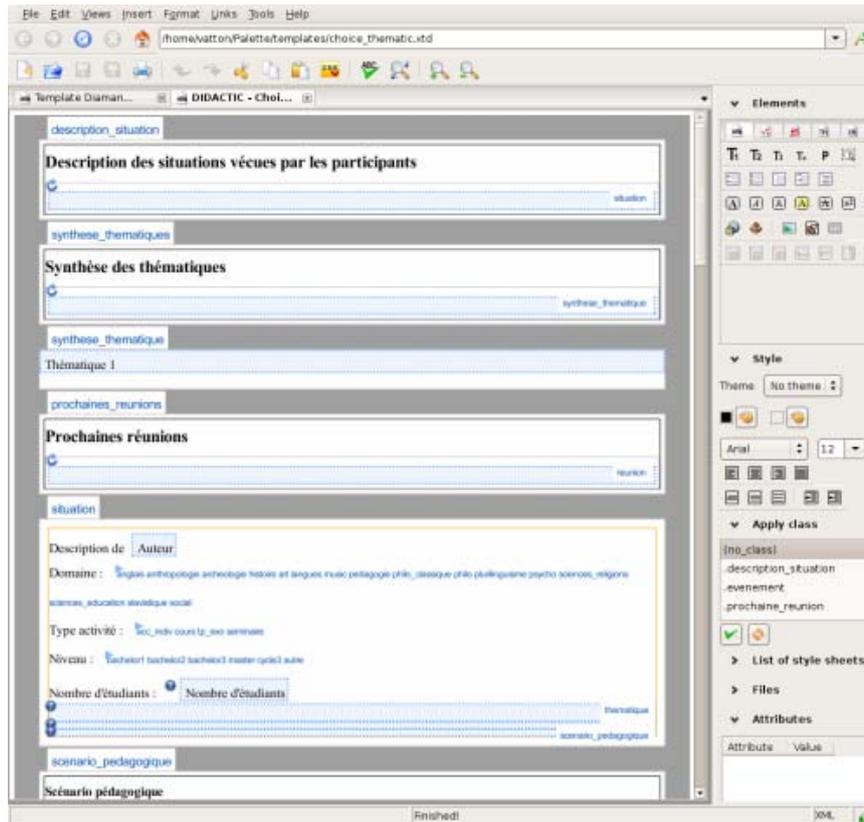


Figure 7: a template for the Did@cTIC CoP

The template is presented as a classical document, as shown in the template instance (Figure 8) with few decorations (some titles and black or yellow frames). But the template brings a lot of hints on what has to be entered where during document editing. For example it allows only a limited set of predefined strings for the domain, the activity type, and the level. These predefined values are presented to the authors, who have just to choose one of them. Some fields are limited to only one paragraph others allow repeated paragraphs. As in the first example, important fields (inserted information) of the document are clearly identified and can be easily extracted for future work.

The final goal is to feed a data base with these structured documents to later find out and analyze the information entered by authors. For example a team of teachers, may quickly find out some years later that a thematic has already been discussed, or they can retrieve examples of situations that have already been experimented and what solutions were suggested in that context.

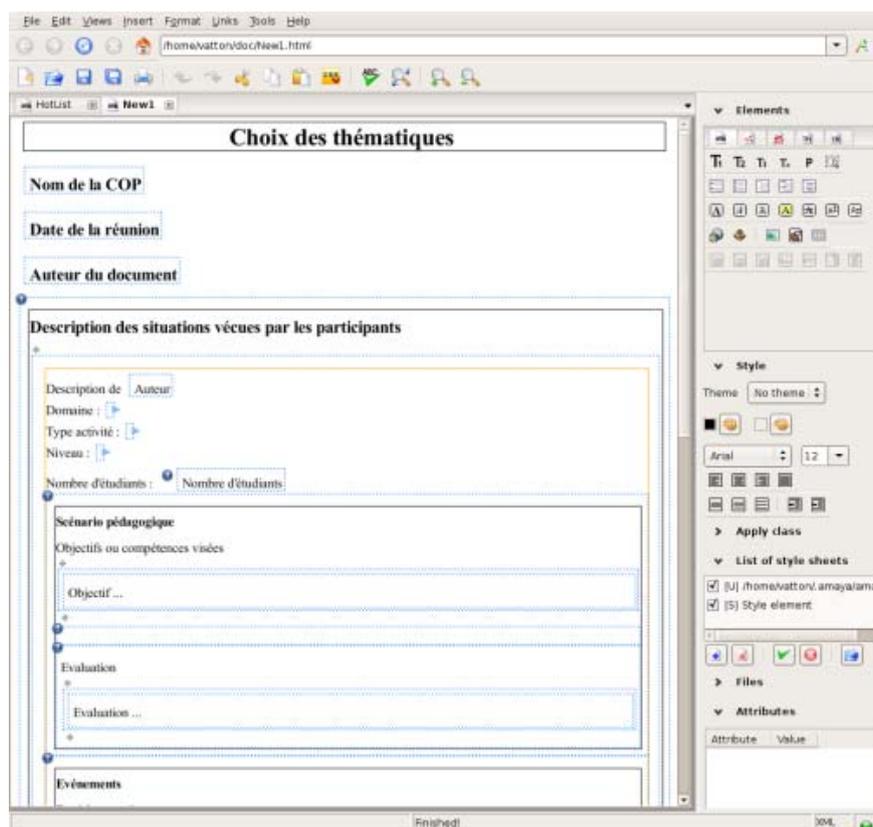


Figure 8: a document instance

2.1.2.3 Conclusion

The above examples demonstrate clearly the advantages of XTiger templates, as opposed to the templates offered by most word processors or HTML editors.

Documents created by Amaya with XTiger templates are structured XML documents, and Amaya makes sure that they are both well-formed and valid (in the XML sense). As a consequence, the whole range of XML tools can be used to process documents in many different ways. They can be filtered and transformed using XSLT (this is also what DocReuse does for reusing documents). They can be searched using both their content and their structure with XQuery or XPath. They can be stored and retrieved in XML databases.

In usual document editors, templates only provide limited help, by proposing a typical document where some parts are unmodifiable and the rest completely free. With XTiger templates, a very fine structure may be defined where it is required, and some constraints may be put on the parts that need to follow certain rules. Obviously, total freedom is also possible on other parts of the document. With such a broad range of possibilities, the structure of documents may be better controlled, thus making documents more (re)usable. But users are not required to use sophisticated tools. Amaya can actually be used as simply as a word processor (see deliverable D.INF.05). The XTiger templates are seen by users as a guide for the construction of a richer document.

2.2 LimSee3

2.2.1 Technical evolution

Version 2.0 of LimSee3 is released along with the present deliverable. In this section we present the evolutions of the application since the version 1.0, which is described in deliverable D.INF.05, and we provide a summary of the current state of work.

2.2.1.1 Document rendering

When authoring a multimedia document, it is very important to have a clear view of the document under construction, i.e. with a quality of the rendering as close as possible to the final output. Therefore, a steady effort was spent on improving the document rendering and on providing easy ways to manipulate it. The first evolution from versions 1.0 to version 2.0 is a brand new display engine which ensures smooth rendering with better handling of transparency, layer stack and text rendering. The user has the possibility to choose the document view among several options, ranging from a preview to a completely outlined layout, especially suitable for detailed authoring. Switching from one view to another is instantaneous.

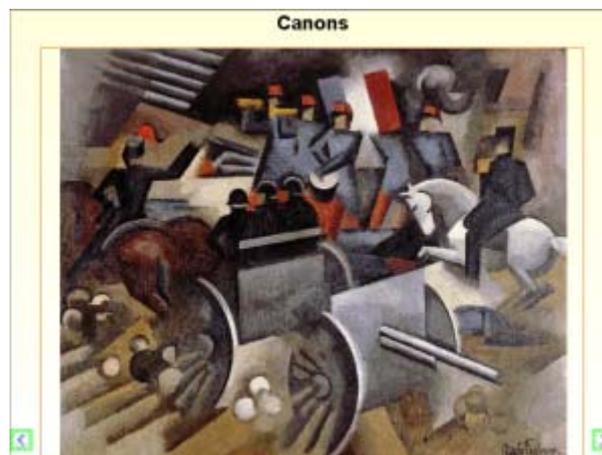


Figure 9: Editing with current element outlined

Several tools have been added since version 1.0 to ease document authoring:

- The **alignment tool** is intended to modify the pre-defined alignment of a media in its container. The media is attached to a point inside its container and a policy is defined on how the media fills the allocated space: it can keep its proper dimensions; it can be automatically resized to fill the container, etc.



Figure 10: Alignment tool

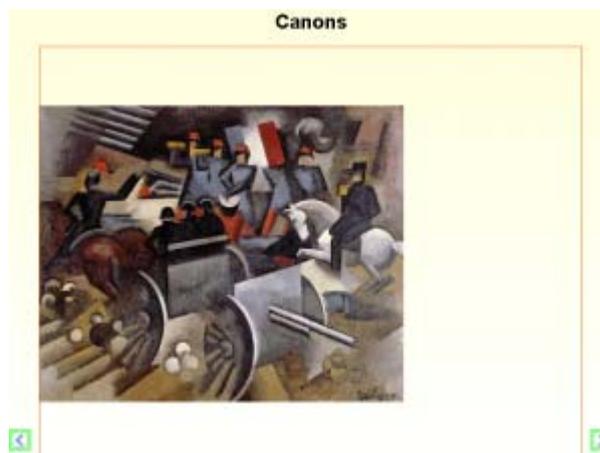


Figure 11: Example of modified alignment

- The **resizing tool** makes it possible to resize and move space containers in a document. This function executes in a full WYSIWYG manner inside the displayed document, so that changes can be checked immediately. When moving an object, magnetic guidelines are provided on its original location, so that it is possible to move it safely along one axis only.

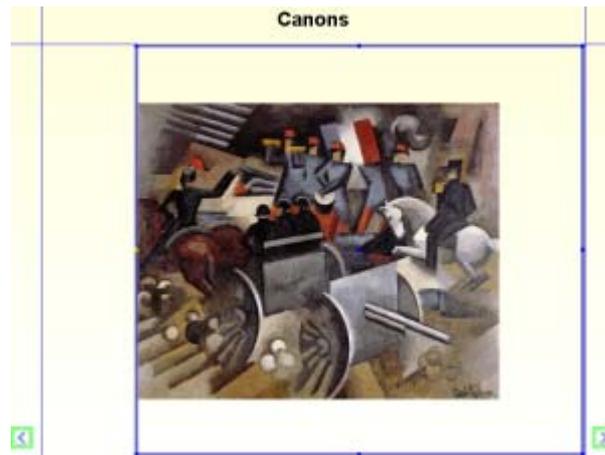


Figure 12: Resizing a container (increasing the left margin)

- The **space hierarchy** presents a document as a hierarchy of containers containing media, under the form of a tree. This representation allows to quickly overview inclusion of containers (parent-child relation) and to manage the **visibility** of the displayed elements. For instance, this tool can temporarily undisplay an object in order to reveal the underneath layers: this can be useful when editing the background of a document.

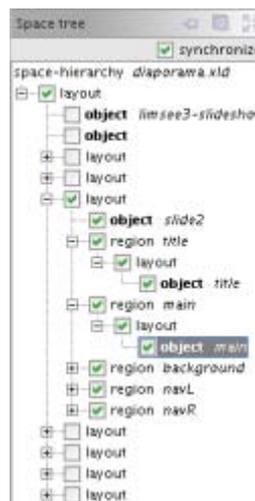


Figure 13: Space hierarchy (unchecked elements are hidden)

These three general-purpose tools join the tools that were already present in version 1.0: **document tree**, **attribute editor** and **document timeline**. The new rendering mechanisms with all the cited tools form a palette of comfortable displaying and authoring facilities that make document editing easy and efficient.

2.2.1.2 Exporting and packaging

Version 1.0 allowed LimSee3 documents to be exported in the SMIL language and, in a rather limited way, in XHTML. Version 2.0 proposes an improved SMIL export (facilitated by the previously mentioned better document rendering) and a complete XHTML+Timesheets export. The latter makes it possible to play documents created with LimSee3 in web browsers that cope with XHTML and standard JavaScript, for instance Mozilla Firefox and Google Chrome. Therefore the wide availability of these browsers ensures a wide access to multimedia documents authored with LimSee3. The export

architecture was presented at the ACM DocEng'08 conference [3] and the particular Timesheets executing library was presented to the SMIL Timesheets working group at W3C.

Moreover, a packaging procedure was developed for version 2.0. This functionality allows safely sharing documents, exported files and all necessary resources between users, thanks to packaged archive files. Packaging ensures users that all included media will be shipped along with a document, so that when unpacked on some other computer, the document is still fully usable.

2.2.1.3 Document publication on Palette repositories

Since version 2.0, LimSee3 is able to inter-operate with two Palette repositories, namely the Common Repository (CoRep) and SweetWiki. Primarily, documents (or packaged documents) can be published on the CoRep directly from LimSee3 in a push-button manner.

Moreover, published documents can be additionally referenced on SweetWiki : a wiki page describing the document is automatically generated and uploaded by LimSee3. The description is based on document meta-data, which are partly calculated by LimSee3 and partly provided by the user. The description also depends on the document model.

2.2.1.4 New document template – annotated audio

Version 1.0 was shipped with two specific applications, corresponding to the two default document templates, *Slideshow* and *Lesson*. Although some improvements were added to those templates, such as an outline wizard for the slideshow, the main progress in version 2.0 has consisted in adding a third template, the *Annotated audio*.

The new template was elaborated during several participatory design sessions with an ePrep CoP member. The initial requirement was to be able to annotate - in real time - an oral examination which is being recorded. The aim is to create one document per oral exam, so as to populate a bank of annotated performances that can be viewed by other students.

The *Annotated audio* template structures the document into three main zones:

- a title zone at the top, which is visible throughout the whole play-time of the document
- a navigation zone beneath, to go directly to a particular annotation
- the annotations themselves at the bottom, visible one at a time; individual annotations are automatically synchronized with the corresponding areas in the navigation bar



Figure 14: An Annotated audio document, with a 2-column layout for annotations

The template comes with a specific tool for annotating. The tool allows to browse, edit, add and remove annotations, to tune the temporal behaviour, all of this in real-time (during the oral exam) or in a stopped-time mode (during post-production).

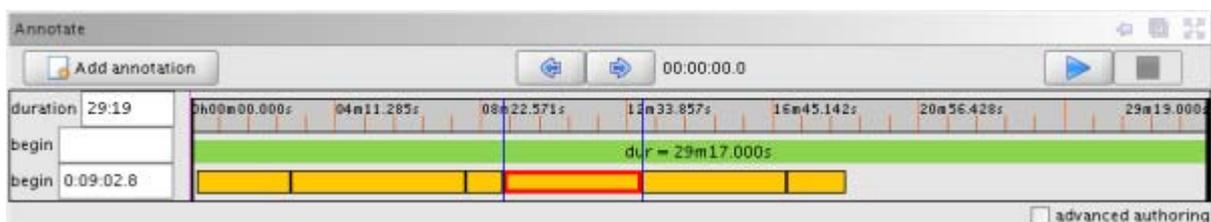


Figure 15: Annotation tool

A real-life test session proved the usability and usefulness of the new template.

2.2.1.5 Library enhancements

Media libraries are provided to users as easy-to-use repositories of media: one can easily organize local and distant files into logical sets (libraries) and then reuse them in documents. Since version 1.0, media libraries have received two main enhancements: it is now possible to preview the contained media under their original aspect before inserting them into a document, and media contained in a library are presented in a way that reflects their origin and current availability.

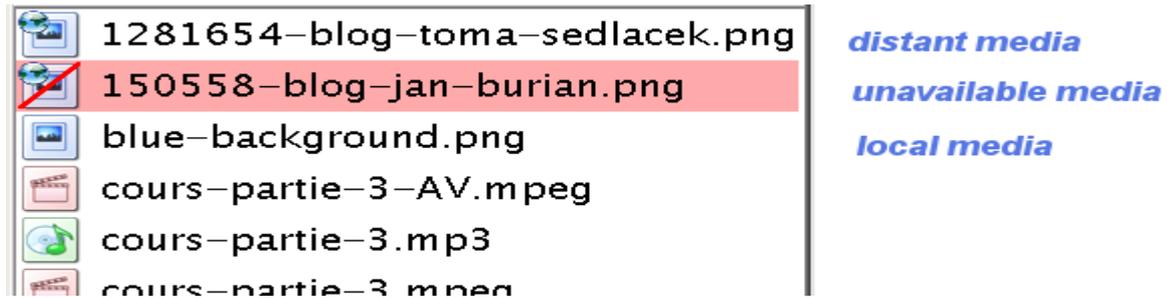


Figure 16: Detail of a library listing

Thus, in version 2.0, media libraries provide an interesting set of functionalities for media management (add, delete, copy-paste, preview, drag&drop, locate). Even if libraries are not strictly necessary for document authoring, since media can be inserted directly from external sources, they provide a higher level of usability.

2.2.1.6 User interface and user preferences

The second usability analysis (presented in deliverable D.PAR.07) allowed us to address a number of usability issues. The modifications that were implied by the analysis are perhaps more visible on the evolution of the user interface between versions 1.0 and 2.0.

The general screen organization was clarified: the library, document and tool zones were better delimited and they were given a new resizing behaviour. The toolbar was separated into a generic part and a document-specific part, toolbar buttons were reorganized and some were redesigned. As a result, the current screen layout is clearer and more predictable; fewer tools are proposed by default, rather they appear as necessary or on demand. A particular effort was spent on uniformization of icons and graphics in general.

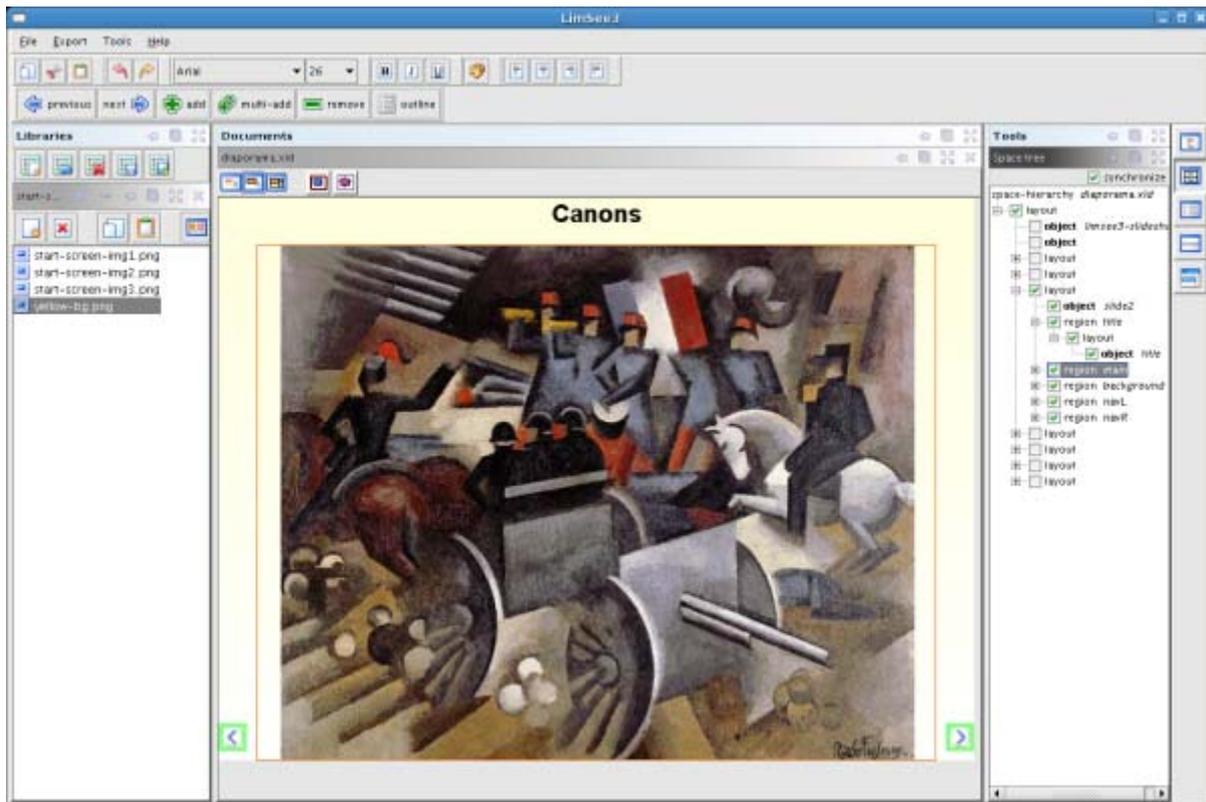


Figure 17: A LimSee3 screenshot

Newly in version 2.0, LimSee3 can be customized through user preferences. Some properties can be explicitly set by the user (e.g. preferred library view, or on-quit behaviour), others are gathered automatically by observing user behaviour (e.g. preferred window size and position, list of recently used documents, etc). Preferences are user-based, thus the same installation of LimSee3 can be customized differently for each of its users.

2.2.1.7 Off-line installation

As part of usability-related requirements, a new off-line installation procedure has been developed for LimSee3. The existing on-line one is kept; the off-line installer simply provides an alternative. The off-line installer presents several advantages:

- The procedure consisting in downloading an installer, then executing the installer which guides the user through a series of screens in order to install LimSee3 at the desired place, is familiar to most users.
- The off-line installer is able to put shortcuts on the desktop and/or in the Program menu.
- The off-line installer can be automatically executed in the same manner on several machines, thus providing the same installation on all of them.



Figure 18: Welcome screen of the off-line installer

2.2.1.8 Localization

Since version 1.0, localization to French has been provided and maintained. A general localization mechanism allows seamlessly adding other localizations as well.

2.2.1.9 Conclusion

Since its version 1.0, released nine months ago, LimSee3 has undergone some major modifications and improvements. The main objectives in this evolution were usability (better authoring experience, easier use), usefulness (new functionalities added through a participatory design process) and interoperability with other Palette tools. Thanks to these guidelines, LimSee3 has reached version 2.0 which offers a mature multimedia authoring tool.

2.2.2 Using LimSee3 in Integrated Services of PALETTE

The main objective of LimSee3 is to provide users with multimedia content authoring in the context of their activities in a CoP. While the authoring step is often a single-user activity, the resulting multimedia documents aim at being retrieved, shared and reused by CoP members. For that purpose, we have developed in LimSee3 the means for using the common resource repository and SweetWiki (see section 1.2.1.3 "Document publication on Palette repositories"). The integration of LimSee3 services inside the palette of services of PALETTE is described in deliverable D.IMP.08.

In this section, we show how LimSee3 provides innovative and pedagogically consistent services for multimedia authoring. Two CoPs have been involved in the use and development of these services: ePrep and Did@cTIC. The initial requirements, issues raised by experimentations and technical solutions are reported below for each of these CoPs.

2.2.2.1 ePrep multimedia pedagogical content production

The multimedia course production service firstly experienced with version 1.0 of LimSee3 has proven its usefulness in the following directions:

- The LimSee3 authoring services automate as far as possible the production process (homogeneous layout provided by templates, generated clickable table-of-content, easy synchronization between audio/video and slides), freeing time of teachers.

- Teachers are able to create and share multimedia courses that go beyond the “simple lecture”: rich content (fully synchronized, navigational content, keywords addition), document produced in several standard formats, easily reusable media content.

These experiments inside the ePrep CoP have given the opportunity to an ePrep member to define a new kind of multimedia content production service that was developed in LimSee3 version 2.0 (see section 1.2.1.4 “New document template – annotated audio”). We briefly describe below the specific requirements of this CoP, the realization process of the new service and its first experimentation.

ePrep teachers have to prepare students for competitive oral examinations that follow rules which depend on the subject. For instance in history, the student gives a 20 minutes presentation that is followed by a 10 minutes questions and answers session. The preparation consists mainly in training the students with several oral tests during the year. There is therefore a large amount of topics that are dealt with by the students and that could constitute a multimedia collection useful for both students and ePrep teachers. Not only student oral presentations are of interest but also teacher comments that are given as text annotations. More precisely the requirements for a new authoring service are the following:

- Authoring live text annotations while the student oral presentation is recorded as an audio media.
- Possibly editing the annotated audio for adding annotations and adjusting their synchronization with the audio track.
- Generating a multimedia document in a standard format to enable publication on multimedia platforms.

All these needs are covered by the new document template - annotated audio and the associated tools provided in LimSee3. No equivalent tool has been found by CoP members. Note that commercial tools like [Articulate Presenter](http://www.articulate.com/products/presenter-annotations.php) (<http://www.articulate.com/products/presenter-annotations.php>) or Audio Annotation tool from Photoshop give the opposite service: annotating static media (images, texts, etc.) with audio content. We can find some audio annotation services that partly cover the needs expressed by the ePrep CoP in some recent research tools. Among them we can cite *Project Pad*, *videoANT*, *Ligne de Temps* and *MediaMatrix*. [Project Pad](http://dewey.at.northwestern.edu/ppad2/index.html) (<http://dewey.at.northwestern.edu/ppad2/index.html>) from Northwestern University proposes several online tools for media annotation and collaboration: one is a transcription tool (with fine-grained synchronization between text and audio fragments), some others provide a form-based metadata edition service to digital objects. The closest tool is the [Audio Tool](http://dewey.at.northwestern.edu/ppad2/07timeline.html) (<http://dewey.at.northwestern.edu/ppad2/07timeline.html>) that provides a similar interface for the annotation process but does not allow the production of a resulting document in a standard format. Similarly [videoANT](http://ant.umn.edu/) (<http://ant.umn.edu/>) from the University of Minnesota proposes a Flash-based web application for annotating videos. The tool "[Ligne de temps](http://www.iri.centrepompidou.fr/)" (<http://www.iri.centrepompidou.fr/>) developed by Centre Pompidou in Paris is devoted to film annotations. Finally [MediaMatrix](http://matrix.msu.edu/~mmatrix/) (<http://matrix.msu.edu/~mmatrix/>) from Michigan State University allows to grab and annotate media stuff picked from the web in order to constitute structured personal media portals (similarly to the media library provided by LimSee3), while [Kora](http://www2.matrix.msu.edu/kora/) (<http://www2.matrix.msu.edu/kora/>) from the same university is devoted to the annotation of audio/video collections using database-driven metadata. As a conclusion of this rapid overview, none of these tools do cover live annotation needs where simplicity and rapidity are of high priority.

The resulting documents of the LimSee3 Annotated Audio service are useful for ePrep CoP members as a means for sharing their pedagogical experiences (for instance what are the comments and the questions that have been asked to each student?) and as pedagogical stuff also (teachers can use them as good / bad use cases of competitive oral examinations). Finally, considering students of a same classroom as a CoP, the shared collection of the annotated presentations is a means for sharing their individual experiences inside the group in order to make progress: the students can access to a wide variety of topics covered by all the oral presentations, they can "replay" the presentations and finely analyze them in the light of the teacher's annotations and questions, and they can compare their

performances and pick ideas and methods from each other. The service proposed here by LimSee3 makes it possible to realize actions of type "enrichment of resources with semantic information" of the first generic scenario "reification scenario" as defined in WP5 in deliverable D.IMP.05.

To really fit with the ePrep CoP, the LimSee3 Annotated Audio tool has been used in a real oral tests session during a whole afternoon. The developers of LimSee3 together with an ePrep member have attended the session (with the authorization of the students) in order to watch the way the teacher uses the tool (see the [report](#) made for the ePrep CoP available at <http://www.eprep.org/communaute/activites/CR151008.html>). The resulting observations have allowed several adjustments of the tool, for instance a 2 column layout for the annotations (see figure 14).

For the ePrep CoP members and students, the added value of this new service relies on:

- its rapidity and simplicity in both authoring and publication steps
- and in the navigation facilities inside the resulting multimedia documents.

2.2.2.2 Did@cTIC annotation tool for arousing socio-cognitive conflicts

The use of LimSee3 in activities related to the reification scenario goes one step further with its experimentation in the Did@cTIC CoP. In this CoP, practice is recorded as reusable traces that are then analyzed and compared in order to let new practice merge out. For instance, in order to understand and enhance teaching practice (how to start a session in order to capture the audience? how to make a course more lively and interactive?), short teaching sessions are simulated between teachers to be then analyzed and discussed. Up to now part of the material was lost because while the simulation is recorded the notes and discussions are not stored (they are on paper sheets).

With LimSee3, this activity can change as follows:

- A teaching sequence is recorded as a video (typically 10 minutes).
- Before the didactic meeting, each participant annotates this video with LimSee3 using a slightly different template (basically allowing a video media instead of an audio media to be annotated). The resulting multimedia documents constitute a first level of practice analysis.
- LimSee3 provides a service for merging the annotations of the participants in order to use the merged document during the meeting: the video can therefore be displayed together with the different annotations that have been set during each period of the video fragment. Moreover it is possible to use LimSee3 to annotate again the document to capture the discussions and conclusions of the didactic meeting.
- The resulting document that integrates these two levels of annotations is published in a standard format (SMIL, XHTML+Timesheets) to be available for every Did@cTIC CoP member.

The use of LimSee3 for instrumenting this activity brings the points of view of the participants face to face and paves the way of keeping traces of the emergence of the cognitive conflicts as required by learning theory.

2.2.2.3 Conclusion

LimSee3 is a new tool that has been developed during the PALETTE project and has therefore heavily benefited from the participative design methodology promoted in the project. Both the authoring services (through templates) and the resulting interface have been designed and implemented in close interaction with PALETTE partners (usability analysis) and CoP partners (usefulness in enhancing their practices).

3 Structured documents reuse tools

DocReuse tool aims to ease the reuse of structured documents (documents produced in respect to an XTiger template). To achieve this goal three services were implemented:

- *Template-driven Evolution*: ensures the automatic adaptation and validity of instances when templates evolve. Moreover, it enables CoP members to modify the template without being familiar with the XTiger syntax. The template-driven evolution service performs structure transformations in a complete transparent manner.
- *Template-driven Extraction*: ensures performing queries on structured documents without being familiar with a specific query language. Users precise what parts they want to extract and under which constraints.
- *Template-driven Structuring*: to produce structured instances, CoPs members have just to upload both the HTML input file to be structured and the target template. Then, the users are able to drag and drop content from the HTML file to instantiate the template appearing as a form that defines the skeleton of the document.

The latest release of DocReuse services was done as part of deliverable D.INF.06. Since that time several new changes and developments have been done. The first major one consists in the design and implementation of a unified interface to access DocReuse services. This unique access entry helps CoPs members to share the same structured documents between DocReuse services and to build a coherent database of structured documents. The second major change is the design and implementation from scratch of the *Template-driven Extraction service* in order to address a specific CoPs need formalized in the generic scenarios (D.IMP.08). New developments and adjustments were made after the trials of the Template-driven Evolution service.

3.1 Technical evolution

3.1.1 DocReuse unified interface

The major goals of the unified interface for DocReuse services could be summarized as the following:

- The interface acts as a unique access point to DocReuse services.
- The interface offers a way to represent in a user-friendly manner the CoP structured resources (templates, instances and unstructured documents). CoPs members have then a coherent view on their structured resources (as they have when they use Moodle for example). The interface makes clear the link between templates and their instances (this link is important in the structured world since it represents the validity of instances regarding a given template). The proposed interface makes also explicit the evolution trace of a given template.
- The interface enables CoP members to manage their structured resources. As shown in Figure 19, CoP members have for example the possibility to preview, rename, export, import and remove templates.

The unified interface for DocReuse services consists of three parts (Figure 19):

- The first one represents the three DocReuse services. A small guide is available for each service (see Figure 20). In order to access the Template-driven Extraction service for example the user should select a template and at least one related instance should be available. A more detailed help is also available (see Figure 21). The help is organised as a FAQ. This FAQ was made incrementally through CoPs trials of DocReuse services. Questions are classified by service; however some common questions are also available.

The second part gives an overview of available templates. Templates evolution trace (as described in D.INF.06) is presented using the column view metaphor of Mac OS X file browser. If the user selects an evolved template from one column, another column will show which templates (if any) have been evolved from this one and so on. The last column on the right summarizes the operations a user could perform to manage templates. In fact a user can preview a template (independently from a used DocReuse service), rename and remove a template, export and import a template. Users can import/export templates from their local desktop (file upload/download) or from the Palette common repository as described in D.IMP.08. The Palette common repository acts as a support service to federate resources from different Palette services. The import/export to Palette common repository functionality was implemented in respect to WP5 technical guidelines for Palette services integration (Figure 22).

- The third part gives an overview of available instances/documents; this part depends on the selected template (of part two). Instances that are associated to the currently selected template are shown (these instances can be selected to access the extraction service for example). In a second tab available unstructured documents are also available (in order to access the template-driven structuring service). The last tab visualises the potentially transformable instances which are the instances related to parent template, i.e. template from which the currently selected template has been obtained. An import/export functionalities (from local desktop or Palette common repository) are also available for instances. Preview, rename and remove operations exist for instances.

Once required resources for a DocReuse service are selected, user can just access the service by a simple click. Figure 23 describes an example of use of DocReuse unified interface, where a user has selected a template which has an associated instance(s). The template-driven extraction service is then ready for use.

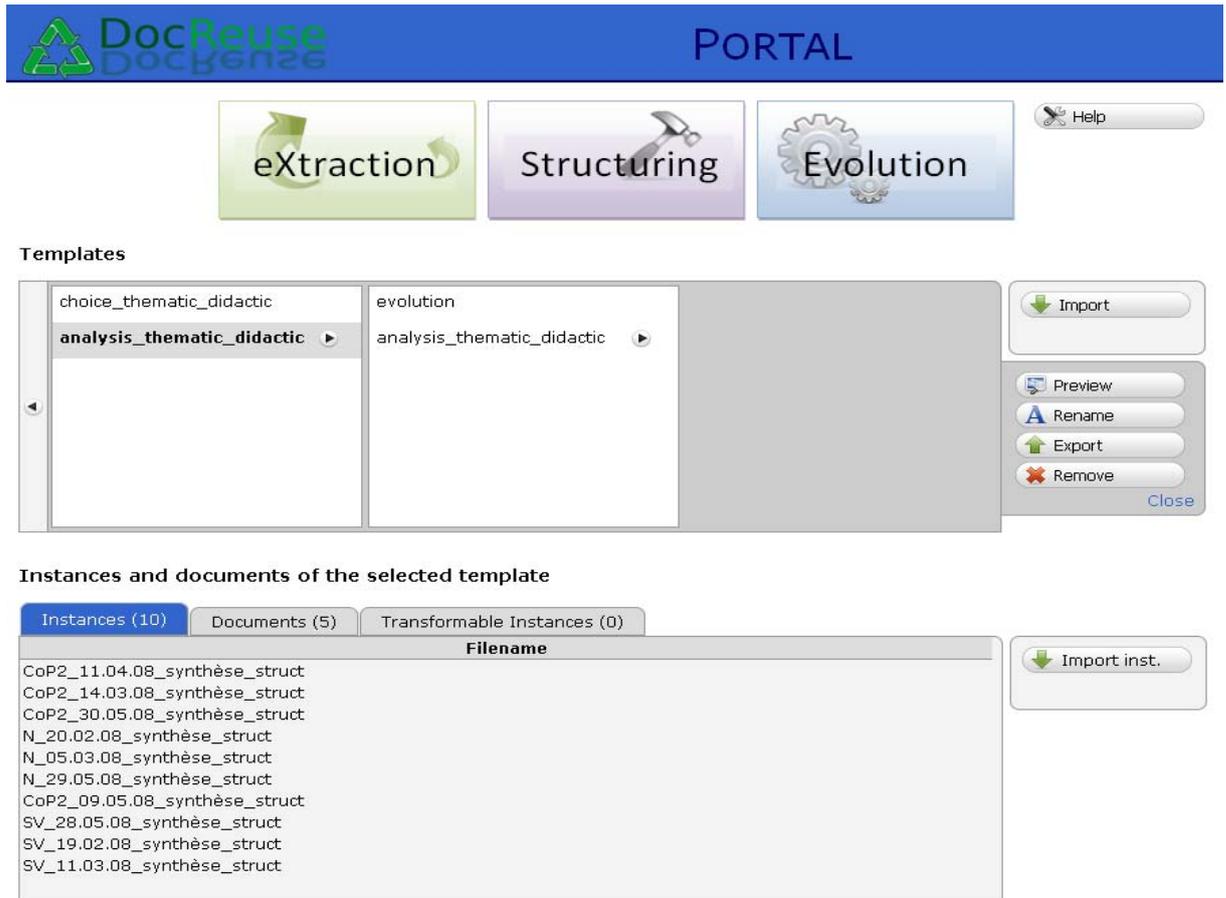


Figure 19: DocReuse unified interface

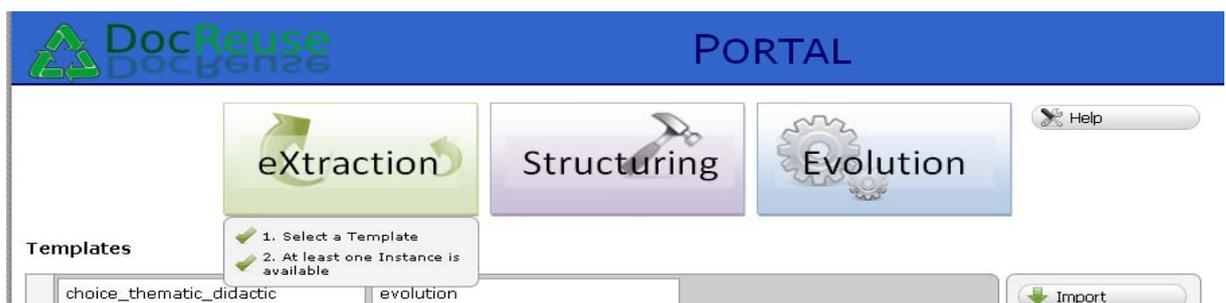


Figure 20: DocReuse: a small help example

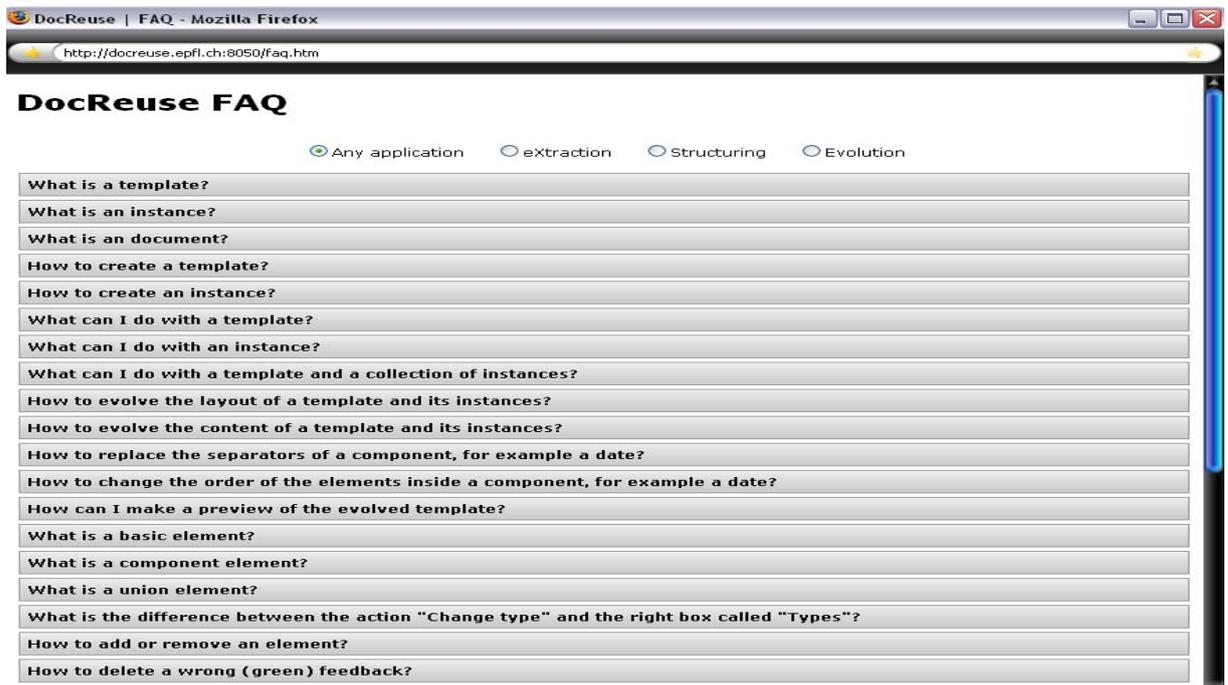


Figure 21: DocReuse services: FAQ

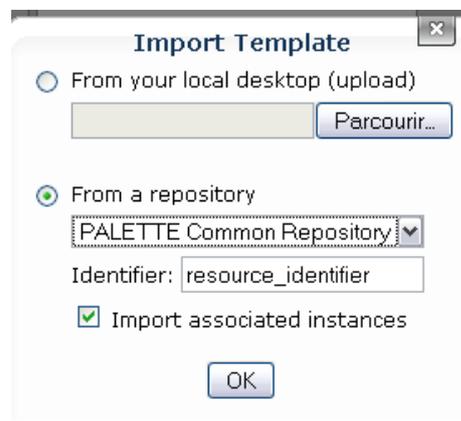


Figure 22: An import template example

DocReuse PORTAL

eXtraction **Structuring** **Evolution** Help

Templates

- 1. Select a Template
- 2. At least one Instance is available

choice_thematic_didactic evolution

analysis_thematic_didactic analysis_thematic_didactic

Import

Preview

Rename

Export

Remove

Close

Instances and documents of the selected template

Instances (10) Documents (5) Transformable Instances (0)

Filename

CoP2_11.04.08_synthèse_struct

CoP2_14.03.08_synthèse_struct

CoP2_30.05.08_synthèse_struct

N_20.02.08_synthèse_struct

N_05.03.08_synthèse_struct

N_29.05.08_synthèse_struct

CoP2_09.05.08_synthèse_struct

SV_28.05.08_synthèse_struct

SV_19.02.08_synthèse_struct

SV_11.03.08_synthèse_struct

Import inst.

Preview

Rename

Export

Remove

Close

Figure 23: Service activation example

3.1.2 2.2 The Template-driven structuring service

The last release of this service was done on September 2008. Several trials were made by the Did@ctic CoP (in the context of the trials of situations related to the reification generic scenario). Based on Did@ctic CoP members' feedback, few developments were made and a new release has been done on October 2008. New developments include the following:

- a common Database of all DocReuse services (structuring, evolution, extraction),
- integrating the newest XTiger specifications,
- improving performances by using the LibXML library for parsing documents rather than REXML,
- integrating the structuring service to the overall architecture of DocReuse,
- improving the user interface according to CoPs members suggestions (Save and preview are done directly on the front page),
- making homogeneous the display of templates with Amaya (icons, colours).

Figure 24 describes the current user interface. The template-driven structuring is currently used by Did@ctic CoP members to support the migration to the structured world.

Figure 24: Template-driven structuring interface

3.1.3 2.3 The Template-driven evolution service

The last release of the template-driven evolution service was made on 28 April 2008 as part of D.INF.06 that describes the proposed functionalities of the service as well as a first prototype. Trials were made with the Did@ctic CoP in the context of the implementation of the situations related to the reification generic scenario. Based on trials results, several developments have been done. As mentioned in D.INF.06 and based on real CoP members situations (described in D.IMP.08) as well as discussions with CoP members (Did@ctic), several requirements were fixed and implemented:

- the service proposes a taxonomy of atomic template evolution primitives that provides a system independent way to specify changes to XTiger templates,
- high level primitives (as described in D.INF.06) are implemented in order to answer specific CoP needs (examples of such primitives are the merge and split operations),
- two forms of system integrity are ensured during the evolution process: *legal* templates (respecting XTiger specifications) and the production of *valid instances*. The first one is ensured since the system controls the applied operations. The second one is ensured through the automatic generation of XSLT scripts.
- an efficient memory of templates evolution was built in order to keep track of the evolution process and to enable CoP members to perform transformation whenever they need: this requirement was implemented at the DocReuse unified interface level. The memory acts as a versioning tool for CoPs structured resources. the developed prototype is compatible with other DocReuse services (same search facilities, same repository, same FAQ, etc..) and other PALETTE services (according to the scenarios of WP5 task 4).

3.1.3.1 Template evolution element primitives

As shown in figure 25, several primitives to make evolve elements were implemented, these primitives include:

- Element insertion: the primitive enables users to add elements to the template by giving its name and type. Messages to make the user aware about the impact of such change are also available; they cover the cases where the system is not able to automate the generation of valid instances (these cases have been discussed in D.INF.06). Figure 26 shows the used interface for this primitive.
- Text insertion: since an XTiger template is a combination of structural information (expressed using the XTiger statements) and target language elements (XHTML), the evolution service offers the possibility to insert XHTML text using a light text editor.
- Move-up/Move-down elements: enables users to change the position/order of elements in an automatic manner.
- Quantify elements: enables the users to change elements cardinality according to Xtiger specifications (see Figure 27 for an example).
- Modify an element: enables the modification of the default value and/or the label of the element (Figure 28).
- Change element type: enables users to change element types using either basic or user-defined types as shown in Figure 29.
- Duplicate element: enables to create a copy of an element.
- Remove-Element: removes an element.
- Merge/split primitives: enables users to merge/split elements. This is high level primitive as described in D.INF.06. Figure 30 describes the splitting of a date to day, month and year fields.



Figure 25: Elements primitives

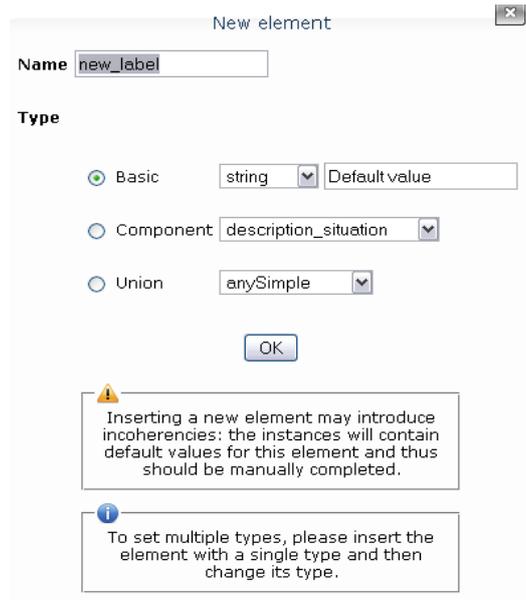


Figure 26: Element insertion primitive

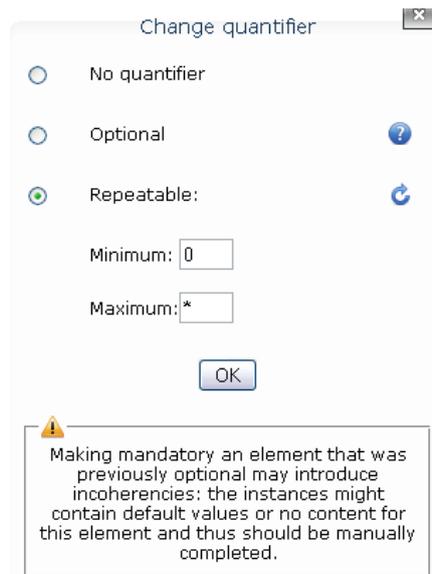


Figure 27: Element cardinality change primitive



Figure 28: Element modify primitive

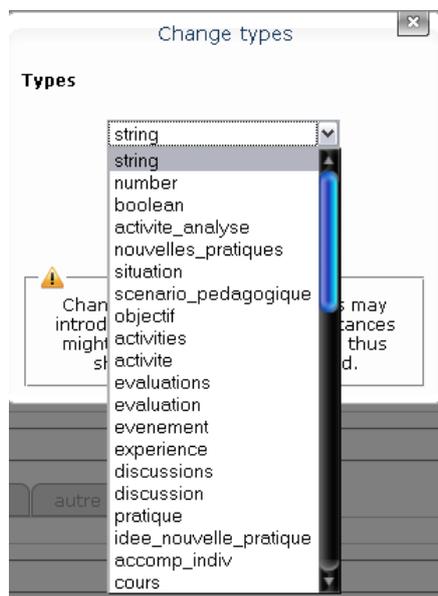


Figure 29: Type change primitive

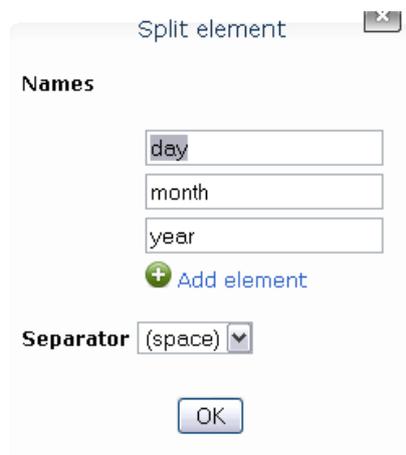


Figure 30: Split/Merge primitive

While applying evolution primitives, the user has the possibility to preview the template as well as an example of an evolved instance in order to better visualise the changes he made. This constitutes the user workspace as shown in figure 31.



Figure 31: Workspace

3.1.3.2 Template evolution types primitives

As described in D.INF.06, template evolution type's primitives were implemented for more experimented CoP users (Figure 32). These primitives concern basic types as well as user-defined types. Users can rename, edit, duplicate and remove types. When a change is done on a type, all the elements of this type are affected. Moreover, users could aggregate several template elements (having the same structural level) to create a new type.

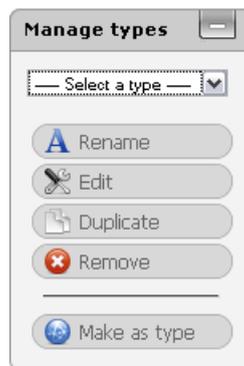


Figure 32: Type management

3.1.3.3 User interface

A new user interface (see figure 33) was implemented, the latter user interface respects the agreement done with the Amaya team on how to display graphically template components and elements.

The template-driven evolution service is integrated to the DocReuse unified interface and is tested and used within the Did@ctic CoP according to the situation described in section 2.5.

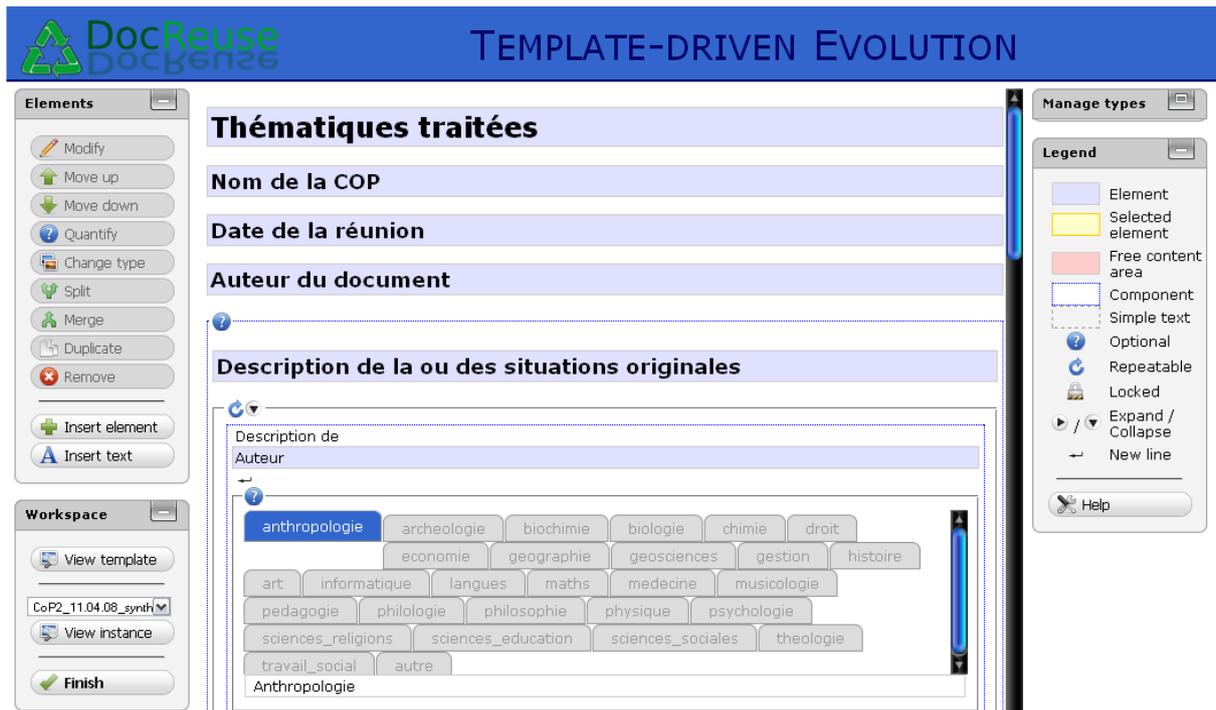


Figure 33: Evolution service User interface

3.1.4 The Template-driven eXtraction service

During the trial of the template-driven structuring and evolution services, discussions with Did@ctic CoP members lead them to express the need to query the structured documents databases capitalised using DocReuse services . The queries should be based on the used template without being familiar with querying languages syntaxes. In order to answer this need, the template-driven eXtraction service was implemented in narrow collaboration with Did@ctic CoP mediators. It enables user to:

- select elements he/her wants to extract and express under which conditions. Conditions are expressed as constraints in respect to the element type (mathematical operators are for example only applied to numerical elements, Boolean are only equal to true or false...). Figure 34 gives an example where users want to extract three elements (Thématiques traitées, Nom de la CoP, and Date de la réunion) for Did@ctic CoP and for meetings occurred the 28th of November;
- combine several conditions using AND/OR operations. Figure 35 gives an example of such combination;
- extract instances parts where two elements have the same values. Figure 36 gives an example of a query extracting instances where the two field “question” and “experience” are identical;
- produce the result in both HTML and XML (see figures 37 and 38).



Figure 34: A query example



Figure 35: And/Or query example

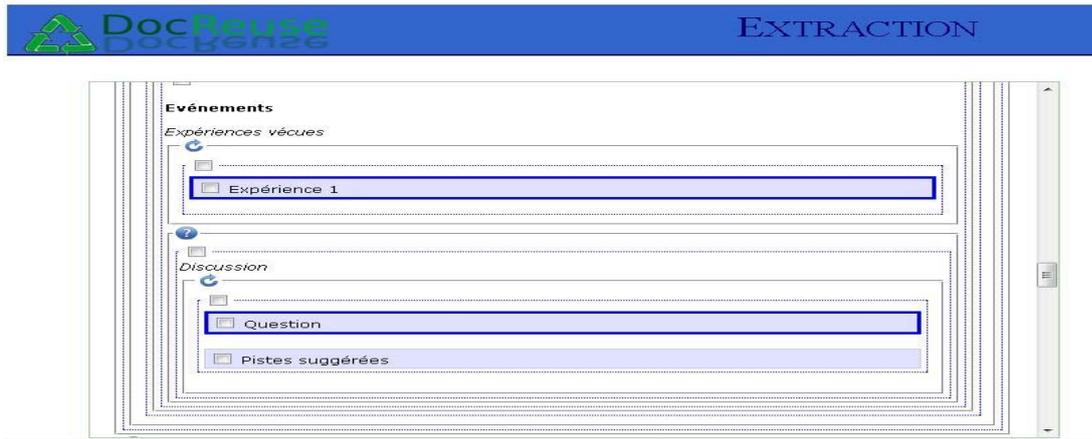


Figure 36: Identical field query example

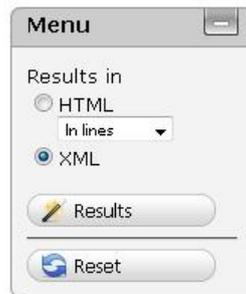


Figure 37: Result menu

Results	
Structuration didactique	
CoP 2	
11 avril 2008	
Sara	
Prise en compte des caractéristiques des étudiant-e-s	
CoP 2	
14 mars 2008	
Bruno	
Comment susciter la participation des étudiant-e-s ?	
CoP 2	
30 mai 2008	
Raphaëla	
Rapport assistant(e) - professeur(e)	
CoP Nautilus	
20 février 2008	
Amos	
Evaluation des compétences	
CoP Nautilus	
5 mars 2008	
Yviane	
Magali	
Marie	
Conseil pédagogique aux étudiant-e-s	
CoP Nautilus	
29 mai 2008	
Yviane	

Figure 38: Result visualisation as HTML document

The template-driven eXtraction service shares the same template visualization with Amaya and others DocReuse services. Produced results are either on XML (can be used with a given stylesheet for a publishing purpose) or in HTML (we define two default stylesheets, the first one presents the results as lines (see Figure 38). The second one presents the results as columns).

3.2 DocReuse in the CoPs

DocReuse services were tested in the context of reification generic scenario. We participate in the implementation of two instantiated scenarios “Document Template Production and Evolution Scenario” as well as “Document reuse scenario”. More precisely three situations were tested with Did@ctic CoP. For this purpose a DocReuse unified interface was deployed for Did@ctic CoP.

3.2.1 Didactic Meeting Capitalization Situation²

“In this situation the Didactic community organizes meetings to talk about the teaching experiences of its members. The community has agreed on a document template to represent the experiences discussed as a structured document where they can easily spot the potential problems that can occur to a teacher and different solutions to handle them. The CoP animator, Anne, first creates the template with Amaya. Then, during a face to face meeting, one of the meeting participants, Pascal, takes the role of a scribe. Depending on personal preferences, he can choose to directly take the meeting minutes using the template produced by Anne using Amaya, or if he prefers, he can also type the notes in a plain textual unstructured document. In that later case, Pascal will later open the raw text document with *Template-driven structuring service* and structure the notes he had taken during the meeting using the template created by Anne”.

The latter situation was implemented. Currently Did@ctic CoP members are using *Template-driven structuring service* in order to capitalize the exchanges they have during their meetings. A valuable database of structured resources was built and made available to CoP animators.

3.2.2 Didactic Meeting Reports Evolution

“The Didactic community keeps its documents in the long term. As it progresses in its understanding of its subject matter, some members suggest additions of new section to the documents, reordering of sections or eventually renaming or deletion of sections. Usually these operations are costly to apply to a set of legacy documents, and progressively the older documents fade away as they are less and less consulted because their content does not reflect the current representation of the community interests”.

Currently, to perform such evolution, Did@ctic CoPs members use the Template-driven evolution service. In a way, this service is one of the added values that can be proposed to encourage a community to structure their documents. The main benefit is to setup an efficient repository that allows to link document template with their instances, so that by applying modifications to one template, all the instances are automatically updated. Several trainings were made and the service is under test and evaluation.

3.2.3 Didactic Meeting Reports Synthesis

“The Didactic community has now been producing meeting reports for a few months, and after 14 meetings, they estimate they have enough material to prepare a consolidated report on the best practices in teaching. To speed up the process, instead of manually collecting the 14 meeting report and cut and paste the interesting parts, they use the extraction service of DocReuse to specify which structural element they are interested in each report instance”.

To reach this goal, Did@ctic members use the template-driven eXtraction service. The result of the extraction is available under two forms. The first form is an XML document that they can directly publish on their site with a specific style sheet. The second form is an HTML document. This

2 The text between quotation is extracted from D.IMP.08 which describes the final version of generic scenarios and situations.

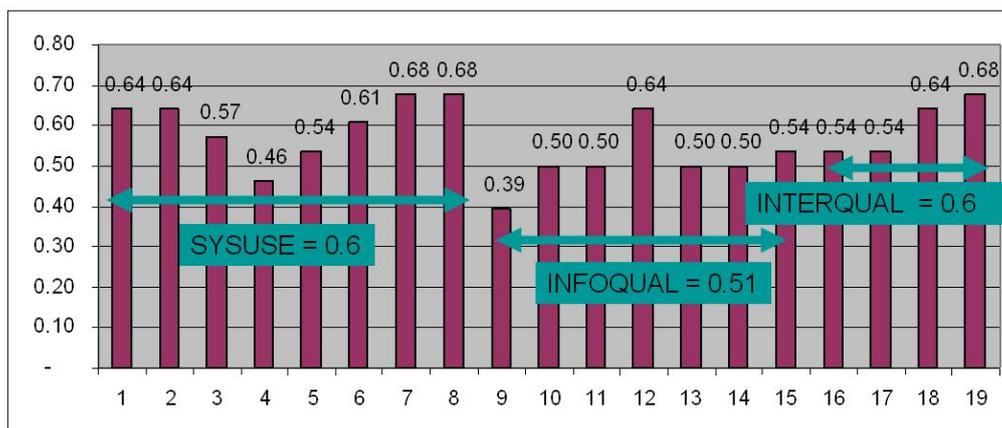
document made of pieces extracted from all the other document instances can be further edited for publishing purposes. Currently, the template-driven eXtraction service is tested within Did@ctic CoP.

The implementation of the three previous situations was also done based on the recommendations collected from CoPs members through a usability test (see next section).

3.2.4 Usability test of DocReuse Services

For evaluating the usability of the set of DocReuse services (the unified interface, the evolution service, the extraction service) we used the CSUQ Questionnaire. This methodology was introduced and also used in the D.IMP.07 in order to evaluate the usability of Palette service browser. Those tests have been performed with by 5 users (mediators and other members of the Didactic CoP) who have tested those applications and filled in the CSUQ questionnaire. This test has been done as a complementary work to WP1 Palette services usability analysis in order to accelerate the gathering of CoPs members recommendations. Those recommendations have been included in the implementation of DocReuse services as described in previous sections.

3.2.4.1 The usability test of the DocReuse Unified interface



The whole score of the questionnaire is equal to 0.56 which is globally acceptable. In addition, the score of the 3 factors SYSUSE, INFORQUAL and INTERQUAL are as follow:

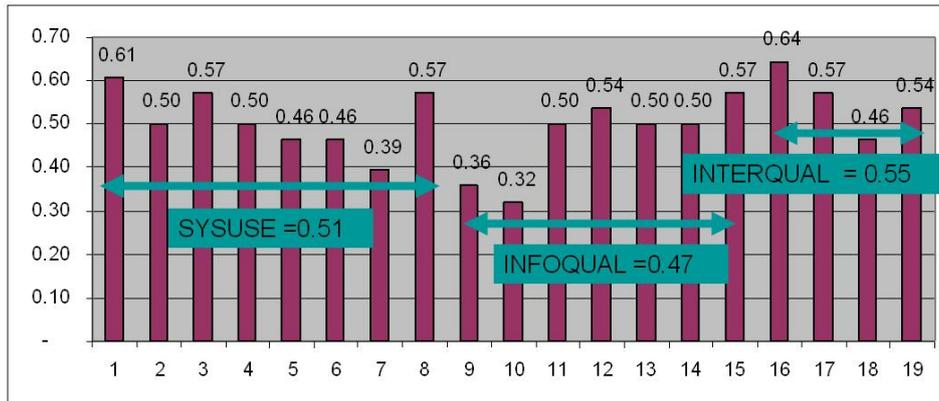
- SYSUSE (System Usefulness, Questions 1-8) = 0.60
- INFOQUAL (Information Quality, Questions 4-15) = 0.51
- INTERQUAL (Interface Quality, , Questions 16-19) = 0.6

Recommendations

Following the score of the INFOQUAL factor (0.51), two recommendations have been proposed, which have been since integrated in the Unified interface :

- Adding titles for differentiating templates, instances and unstructured documents
- The caption of the button import might be more explicit to avoid confusion for user : import document, import template, import instance

3.2.4.2 The usability test of the Evolution service



The whole score of the questionnaire is equal to 0.50 which is acceptable. In addition, the score of the 3 factors SYSUSE, INFOQUAL and INTERQUAL are as follow:

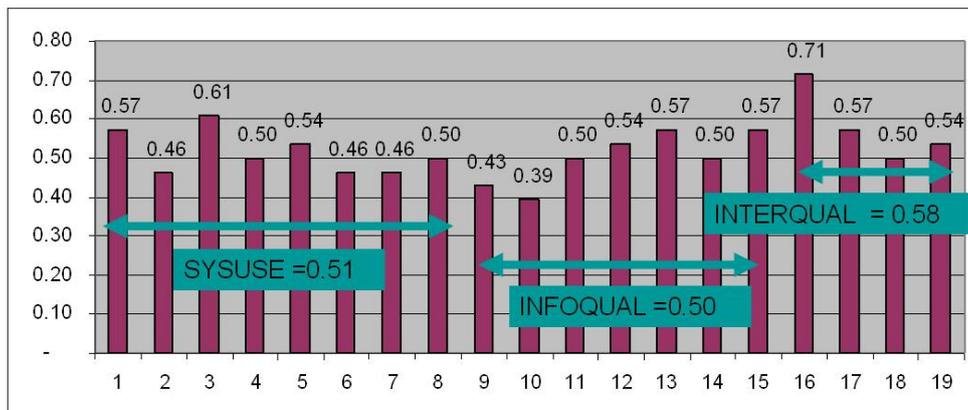
- SYSUSE (System Usefulness, Questions 1-8) = 0.51
- INFOQUAL (Information Quality, Questions 4-15) = 0.47
- INTERQUAL (Interface Quality, Questions 16-19) = 0.55

Recommendations

Following the score of the INFOQUAL factor (0.47), some recommendations have been proposed, which have been since integrated in the Evolution service:

- Add undo button to cancel the latest modification on the structure of the document
- Fix a default value for the element date : DD/MM/YYYY
- Fix the constraints on the multiple selection of element of different levels

3.2.4.3 The usability test of the Extraction service



The whole score of the questionnaire is equal to 0.52 which is acceptable. In addition, the score of the 3 factors SYSUSE, INFORQUAL and INTERQUAL are as follow:

- SYSUSE (System Usefulness, Questions 1-8) = 0.510
- INFORQUAL (Information Quality, Questions 4-15) = 0.50
- INTERQUAL (Interface Quality, , Questions 16-19) = 0.58

Recommendations

Following the score of the INFORQUAL factor (0.50), many recommendations have been proposed, which have been since integrated in the PSB:

- Adding buttons for using AND OR operators in order to formulate queries
- Integrating a combo Box for choosing from the list of domains
- Allowing the possibility to choose within a list of documents to perform the extraction
- Improving the rendering of the extraction's results : deleting the space between titles
- Adding screenshots within the help
- Adding more explanation about the merge button

3.3 Conclusion

DocReuse services implementation fully respects the participative design methodology as well technical integration guidelines. Exchanges with CoPs members lead to the release of DocReuse services that fully answer CoPs needs. DocReuse services are currently deployed and used by Did@ctic CoP. We participate to change their practices and to help CoPs members to use structured documents.

4 Conclusions

The objective of WP2 is to ease the production of multimedia documents and the reuse of structured documents. This was achieved during the Palette project by the implementation of three tools: Amaya a template-based editor, LimSee3 a multimedia authoring tool and DocReuse services to automate the reuse process.

The implementation of the three tools was done following:

- the use of standards and open document formats as recommended in Palette technical guidelines (D.IMP.01),
- the CoPs needs specified in the context of teams work (task 4 of WP5) and formalized as generic scenarios (D.IMP.08),
- the participative design methodology (specified in the context of WP1): the design and implementation was done in close collaboration with CoPs members as described in this deliverable.

All information services are currently used by CoPs and have influenced their practices.

5 References

- [1] R. Deltour and C. Roisin, The LimSee3 Multimedia Authoring Model. DocEng2006 ACM Symposium on Document Engineering, 10-13 October 2006, Amsterdam, The Netherlands, pp. 173-175
- [2] Aida Boukottaya, PHD: “Schema Matching for structured document transformations”, EPFL, October 2004.
- [3] Jan Mikác, Cécile Roisin, Bao Duc. *An export architecture for a multimedia authoring environment*. In DocEng’08: Proceedings of the eighth ACM symposium on Document engineering, pages 28-31, ACM, September 2008

6 Annexes

Overall Reaction to the Website		1	2	3	4	5	6	7		NA
1. Overall, I am satisfied with how easy it is to use this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
2. It was simple to use this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
3. I can effectively complete my work using this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
4. I am able to complete my work quickly using this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
5. I am able to efficiently complete my work using this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
6. I feel comfortable using this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
7. It was easy to learn to use this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
8. I believe I became productive quickly using this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
9. The website gives error messages that clearly tell me how to fix problems	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
10. Whenever I make a mistake using the website, I recover easily and quickly	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
11. The information (such as online help, on-page messages, and other documentation) provided with this website is clear	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
12. It is easy to find the information I need	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
13. The information provided by the website is easy to understand	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
14. The information is effective in helping me complete the tasks and scenarios	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
15. The organization of information on the website pages is clear	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
16. The interface of this website is pleasant	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
17. I like using the interface of this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
18. This website has all the functions and capabilities I expect it to have	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						
19. Overall, I am satisfied with this website	strongly disagree	<input type="radio"/>	strongly agree	<input type="radio"/>						

Annexe 1: The CSUQ Questionnaire