La logique d'ouverture, normalisation et standardisation au service de la pérennité du LCMS

Sébastien CHAUMAT, Sébastien PILLOZ, Gérard VIDAL

PR@TIC, École Normale Supérieure de Lyon, 46 allée d'Italie, 69364 Lyon Cedex 07
ERTé ACCES (Actualisation Continue des Connaissances des Enseignants de Sciences)
INRP, Place du pentacle BP 17, 69195 Saint-Fons Cedex
Sebastien.Chaumat@ens-lyon.fr, Sebastien.Pilloz@ens-lyon.fr, Gerard.Vidal@ens-lyon.fr

Résumé

Le service PRATIC de l'École Normale Supérieure de Lyon a appliqué une démarche d'ouverture, de normalisation et de standardisation lors de la mise en place de son système de gestion de contenus pédagogiques. Cela concerne aussi bien les modèles de données et de métadonnées, les implémentations, les formats de fichiers, les architectures et infrastructures logicielles de gestion et stockage, les interfaces homme-machine, et même le mode de développement logiciel. Nous assurons ainsi la pérennité, l'interopérabilité et l'évolutivité de l'ensemble de nos ressources pédagogiques numériques et dispositifs associés. Les développements logiciels et l'exploitation du système sont largement améliorés par cette approche.

Mots-clés : Produire et gérer les contenus éducatifs, architecture et interface utilisateur

Abstract

The PRATIC department of Ecole Normale Supérieure de Lyon applied openness, normalization and standardization to the development of its Learning Content Management System. Models for data and metadata, files formats, architectures and software infrastructures for management and storage, human-machines interfaces and even software development style are affected. Thus the durability and the evolutivity of the whole set of our learning resources and connected systems are assured. Software development and system exploitation are improved by this approach.

Keywords: Producing and managing educational content, architecture and user interface.

Introduction

L'ouverture, la normalisation et la standardisation sont les clefs de la pérennité des objets numériques et de leurs catalogues. Le Pôle de Ressource et d'Assistance pour les Technologies de l'Information et de la Communication (PRATIC) de l'École Normale Supérieure de Lyon (ENS Lyon) a souhaité appliquer cette démarche à son extrême lors de la mise en place du Système de Gestion de Contenus Pédagogiques (Learning Content Management System, LCMS) et des systèmes périphériques : l'Espace Pédagogique Intégré (EPI) [1], les sites d'accompagnement des enseignants CultureSciences-Physique [2] et Planet-Terre [3], le campus numérique ESCALES [4] et le site de diffusion de la connaissance scientifique DICOS [5].

Cette démarche d'ouverture, normalisation et standardisation, concerne aussi bien les modèles de données et de métadonnées, les implémentations, les formats de fichiers, les architectures et infrastructures logicielles de gestion et stockage, les interfaces hommes-machines, et même le mode de développement logiciel. Nous assurons ainsi la pérennité, l'interopérabilité et l'évolutivité de l'ensemble de nos ressources pédagogiques numériques et dispositifs associés. L'agréable effet secondaire est que les développements sont à l'optimum de la facilité et de l'efficacité dans ce cadre.

Données : Contenus Scientifiques et Pédagogiques

Nous avons balisé la mutation des enseignements de l'ENS Lyon vers le numérique en plusieurs étapes. D'abord collecter l'existant dans une structure de travail collaboratif ; ensuite analyser l'offre actuelle en matière de contenus numériques puis proposer et implémenter des structures types pour les nouveaux contenus et la migration des anciens. Parallèlement proposer un système d'indexation des ressources et ensuite le basculement des systèmes de stockage vers le véritable LCMS centré sur les métadonnées.

Diversité des Formats de Fichiers

L'absence de directive aux auteurs sur les objets mis à disposition des apprenants conduit nécessairement à alourdir à la fois la gestion de ces objets mais aussi l'usage qui en est fait par l'apprenant. À condition d'encapsuler le document pédagogique dans un objet de plus haut niveau, on peut éliminer les difficultés de gestion liées aux formats hétérogènes.

Mais proposer cinq à six formats de fichiers différents pour les documents de base reste, du point de vue des usages, une option discutable. Non seulement car cela implique forcément des problèmes d'interopérabilité, mais aussi car les formats habituellement utilisés dans notre établissement ne sont pas forcément les plus pertinents pour l'usage pédagogique dans le contexte technologique moderne.

Les problèmes d'interopérabilité ne sont pas forcément où on les attend. Par exemple, nous avons

été confrontés à un fichier, rédigé sous MacOS avec le logiciel Microsoft Word, et déposé par un enseignant dans le groupe de travail de la préparation à l'agrégation de Sciences de la Terre et de l'Univers dans notre EPI. Bien que parmi les plate-formes mises à dispositions des étudiants dans l'établissement, l'une dispose de la dernière suite office de Microsoft sur PC, le fichier s'avérait illisible pour les usagers. Nous avons rétabli la situation en convertissant le fichier au format PDF via le logiciel libre OpenOffice. Par la suite OpenOffice a été installé sur les plates-formes clientes, ce qui constitue une amélioration de l'interopérabilité mais aucunement une solution définitive au problème.

Plus généralement il est inenvisageable et inutile de présupposer de la plate-forme matérielle et logicielle du client. Cela est d'autant plus vrai dans le contexte européen de la réforme LMD (Licence Master Doctorat) qui ne fait qu'accroître la diversité des plate-formes terminales (et cela est intrinsèquement bon). L'interopérabilité ne peut alors être assurée que si elle est initiée au niveau de la plate forme de mise à disposition des contenus.

Structures des Média

Au delà du format du fichier contenant le média pédagogique, on peut aussi s'interroger sur le type structurel de ce dernier (mono ou multi-média, linéaire ou non). La réflexion devient alors plus pédagogique que technique mais les répercussions sur les choix de formats de fichier et logiciel sont immédiates. Par exemple, on comparera une présentation linéaire de transparents de cours sous formes de diapositives (formats et logiciels de créations usuellement propriétaires) à l'utilisation d'un document multimédia contenant les images des transparents, la vidéo du professeur et la navigation non linéaire via une table des matières (format normalisé SMIL 2.0 [6], logiciel de création libre LimSee2 [7]).

Modélisation de la Structure des Contenus

Ces exemples montrent la nécessité de la réflexion au niveau du producteur de contenu sur les formats et structures de données.

Mais avant même d'entamer une telle démarche au niveau d'un établissement, il convient de s'interroger préalablement sur la possibilité d'un modèle abstrait commun pour nos contenus duquel dériveront ensuite les implémentations structurelles et les formats de fichiers.

L'inventaire des ressources numériques existantes à l'ENS a permis de dégager rapidement quelques grandes familles de données. L'analyse nous a permis d'identifier la brique de base de nos contenus scientifiques et pédagogiques : l'illustration, composée (de manière simplifiée) d'une image, de sa légende et de ses métadonnées (auteur, donateur, droits, etc). Dans les disciplines telles que les sciences de la Terre ou les sciences de la vie, les illustrations sont très souvent exploitées sous la forme de planches de plusieurs illustrations, a l'instar des planches des ouvrages de botanique.

Au delà de ces briques de base nous avons réussi à identifier un modèle de structure classique pour les documents scientifiques proche dans ses possibilités des classes LaTeX proposées par les éditeurs de revues. Au nombre des variantes apportées, on citera la possibilité d'appuyer du texte de plusieurs façon différentes (par des couleur notamment) comme cela est utilisé par exemple pour faire ressortir la structure d'une formule mathématique.

Implémentation de la Structure en XML

Nous avons choisi d'implémenter cette structure abstraite sous forme d'un schéma XML pour d'évidents soucis de pérennité et d'interopérabilité, comme imposé au point 3.2.3 de l'annexe « Interopérabilité » du Schéma Directeur des Environnement Numériques de Travail (SDET)[8]. Nous avons estimé que XML est un langage assez souple pour permettre tous les usages nécessaires à la production des documents pédagogiques de base sans qu'il soit nécessaire de lui substituer un langage ad-hoc.

S'est posé alors la question des transformations que nous souhaitions faire subir à nos contenus. Dans un soucis d'efficacité et de respect des standards existants il nous a paru économe de nous appuyer sur l'énorme bibliothèque de transformations et d'outils existant pour le format DocBook [9] (qui possède une implémentation XML en plus de son implémentation SGML). Dans la suite nous dénommerons notre schéma général « LivreDoc ».

Deux raisons nous ont poussés à ne pas utiliser directement le DocBook.

Tout d'abord, même la sous partie du DocBook dite DocBook-simple[10] est excessivement complexe à appréhender car beaucoup trop riche et lâche au niveau des contraintes de structuration. Or si les membres des communautés de physique, mathématique ou informatique ont l'habitude de structurer leurs documents en utilisant un langage de marquage comme LaTeX, à l'opposé les biologistes et les géologues ont l'habitude du texte formaté mais non structuré (la faute aux outils WYSIWYG qui n'imposent pas de structuration à priori). Nous souhaitions une introduction douce aux notions de séparation structure/fond et de formatage a posteriori, ce que DocBook ne permet pas.

L'autre raison de l'inadaptation de DocBook aux usages de nos producteurs de contenus est l'ambiguïté du vocabulaire des balises. A contrario, à l'échelle d'un établissement il est facile de mettre au point un vocabulaire pour les balises qui soit en français et compréhensible par tous très rapidement.

Il ne restait donc qu'à garantir que notre schéma XML de données était structurellement identique à une sous partie du DocBook (si possible par une transformation conforme pour simplifier les transformations vers DocBook et l'utilisation des outils connexes comme les feuilles de style applicatives) et à

implémenter la transformation de LivreDoc vers DocBook. Ce travail a été réalisé par Eric Van der Vlist (société Dyomedea [11]) en suivant une approche légère et innovante pour réduire et traduire DocBook en une seule opération [12].

Concernant les formules mathématiques, pour pallier le manque de diffusion actuel de la recommandation MathML [13], nous permettons tout simplement deux formes pour les équations : décrites selon MathML ou bien sous forme d'image (comme c'est le cas dans les traductions automatiques de LaTeXvers HTML). Le format utilisé est sélectionné au niveau du système de publication ou de transformation. Nous éviterons ainsi tout travail de conversion vers MathML lorsque ce dernier sera plus largement utilisable. Ainsi nos contenus en XML peuvent très facilement être publiés sous n'importe quel format classique, actuel et à venir.

Pour finaliser notre démarche d'ouverture, le schéma XML LivreDoc est bien entendu public.

Manipulations des Données

Un autre des avantages d'une implémentation XML de nos modèles de données est le large choix de logiciels utilisables pour l'édition de nos contenus. Du XML dans le texte (notepad, XEmacs) à l'utilisation d'éditeurs avec couche de présentation (Authentic d'Altova [14], XXE de Pixware[15]) en passant par les éditeur XML spécialisés qui insèrent automatiquement les balises (Oxygen de Synchrosoft [16], Jedit de Slava Pestov (GPL) [17]), toutes les variations sont permises. Évidemment le codage d'applications autour de nos documents est aussi largement simplifié par l'existence de nombreux outils élémentaires et API autour du XML.

Enfin la raison majeure du passage en XML est l'existence de bases de stockage natives qui décomposent le document et permettent toute sorte d'application comme l'introduction de critères sur la structure des données dans les recherches. Le serveur XML natif utilisé à l'ENS Lyon est décrit ci-après.

Conclusion sur les Données

En résumé les documents pédagogiques produits par l'ENS Lyon sont conformes à un modèle abstrait complètement identifié.

Ce modèle a ensuite été implémenté selon un schéma XML, LivreDoc, topologiquement équivalent à une sous partie du format DocBook. Nous disposons des outils de transformation des données vers le format DocBook puis vers tous les formats atteignables depuis ce dernier, ce qui couvre quasiment tous les formats structurés existants.

Métadonnées : LOM

L'intérêt du Learning Content Management System est de garantir une gestion pertinente des documents pédagogiques. Cette gestion doit s'appuyer sur la pédagogie qui entoure l'objet et non pas sur l'objet lui-même. Cela revient à baser le LCMS non pas sur une base de données mais sur une base de métadonnées.

Modèle et Implémentation

Dans ce domaine aussi nous avons choisi la voie de la normalisation en adoptant le standard IEEE Learning Object Metadata (LOM) et son implémentation officielle en XML.

Nous avons choisi de respecter strictement la norme et de déporter sur les interfaces de saisies les problèmes d'adaptation à nos usages. Le standard LOM est assez souple dans sa définition des champs et vocabulaires obligatoires.

Ce choix stratégique permet d'éviter l'écueil du codage d'une application spécifique pour l'édition des métadonnées. Aucun logiciel de ce type n'est viable car il faut sans cesse en modifier le code pour s'adapter aux évolutions de la norme ou aux changement d'usages. Nous garantissons aussi que nos métadonnées seront échangeables en l'état avec nos partenaires.

Usages

N'importe quel éditeur XML permet d'éditer les fiches de métadonnées ce qui s'avère suffisant (et parfois souhaitable) pour les utilisateurs habitués aux langages de structurations comme latex.

On peut améliorer l'ergonomie de saisie avec des logiciels qui ajoutent une couche de présentation (CSS le plus souvent) à l'interface de saisie. Ce sont exactement les mêmes logiciels que pour l'édition des données, d'où une économie supplémentaire, notamment sur la durée d'apprentissage.

On peut encore raffiner le système en ajoutant la possibilité de gérer des masques de saisie. Par exemple pour une première indexation rapide d'un document on propose un masque de saisie de la fiche LOM qui cache toutes les balises sauf les 15 que nous avons déterminées comme fondamentales. Pour une indexation plus poussée nous proposons un masque de saisie maximal (aucune balise cachée).

L'optimum d'ergonomie s'obtient en ajoutant les fonctionnalités suivantes (voir figure 1) :

- possibilité de gérer des templates (fiches type pré remplie) qui remplissent partiellement ou complètement le masque de saisie
- génération automatique de certaines métadonnées (ex : la taille du document est calculée automatiquement par le serveur lors du dépôt initial)

Une alternative à cette ergonomie est la notion de scénario d'indexation, où plusieurs acteurs interviennent (déposant, auteur, indexeur), chacun ayant à sa disposition un masque de saisie différent des autres et ne pouvant intervenir qu'après l'acteur précédent. Ce système est avantageux lorsque ces acteurs sont clairement identifiés dans l'établissement. Dans les configurations gérées par l'ENS Lyon, les workflows correspondants ne contiennent en général qu'une ou deux étapes.

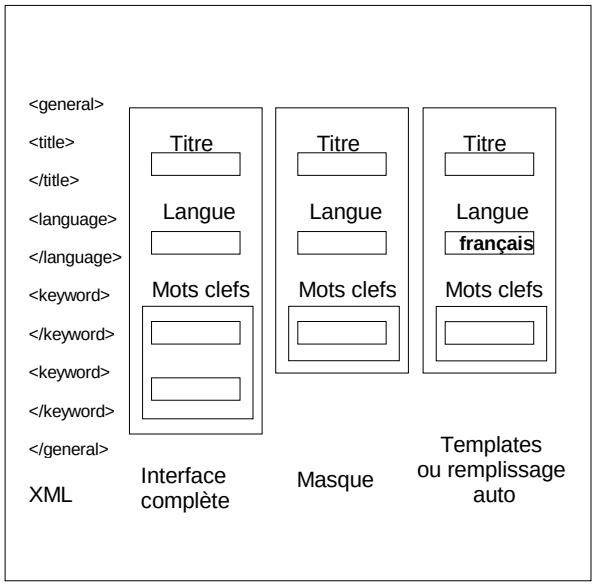

FIG. 1 : Édition de métadonnées utilisant masques et templates..

De gauche à droite : la structure LOM en XML, vue de cette structure via une interface (formulaire), masquage des balises non souhaitées au niveau de l'interface, pré remplissage automatique de certaines balises.

Outils de Manipulations des Métadonnées

Nous avons donc fait développer un système générique d'éditions de métadonnées LOM par Kapil Thangavelu (http://Zope.org/Members/k_vertigo). Ce système permet la création de masques de saisie et la gestion de templates. Pour garantir le caractère ouvert et interopérable de ce logiciel, celui-ci est libre et indépendant du schéma XML des métadonnées. Un point important est que nous développons systématiquement des API (Application Programming Interfaces) entre nos différents systèmes. Ceci permet de segmenter les développements et de rendre chacune des parties du système indépendante des autres. Ainsi l'éditeur de métadonnées est indépendant du backend de stockage de celles-ci. L'éditeur repose sur les services du Content Management System (CMS) Plone [18] (que nous utilisons abondamment) mais rien n'oblige à utiliser ce dernier pour gérer les données. Nous comptons incorporer ce produit, qui fait la jonction entre Plone et une base de métadonnées externe dans le projet de LCMS libre eduplone.

Échange des Métadonnées

L'échange des métadonnées et leur syndication sont sans aucun doute les aspects les plus dépendants de la standardisation. À terme c'est principalement cette fonction qui sera assurée par le LCMS. En effet, les flux de données sont destinés à saturer une fois que tous les enseignements dispensés par l'établissement sont couverts et que l'on entre dans la phase de maintenance et mise à jour des contenus. À l'opposé, les flux de métadonnées vers l'extérieur doivent continuer de s'accroître puisque celles-ci seront de plus en plus consultées pour déterminer la présence de ressources dans le LCMS. En particulier l'ouverture de la base de métadonnées doit être mondiale pour permettre aux étudiants de tout pays de sélectionner les cours qu'ils souhaitent suivre. Il s'agit aussi de la partie la plus simple et la plus balisée du LCMS. Les propositions de normes internationales en la matière sont publiées par l'Open Archive Initiative (OAI)[10] qui développe et cherche à promouvoir les standards d'interopérabilité qui facilitent l'échange des contenus. La mise à disposition et la collecte des métadonnées sont l'objet du standard Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH 2.0) qui s'impose naturellement à nous. L'offre logicielle libre est abondante de ce côté et le positionnement du serveur « OAI » dans l'architecture globale du LCMS est relativement peu contrainte.

Infrastructure/Architecture Logicielle de Gestion/Stockage

Au delà de l'utilisation de formats standards ou ouverts pour le stockage et la manipulation des (méta)données, il faut garantir aussi la pérennité de l'exploitation de celles-ci. Pour répondre à cette contrainte nous nous appuyons uniquement sur des logiciels libres pour notre infrastructure. Ce choix garantit l'indépendance du dispositif technologique et n'est conditionné que par la disponibilité des logiciels adéquats (ou de la possibilité de les développer facilement). Le coeur du système est une base de donnée XML native : eXist [19]. Nos applications web sont construites sur le serveur d'application Zope sur des serveurs Linux. Initialement, nous écrivions directement les application au dessus de Zope (en DTML ou ZPT). Ensuite nous avons déployé des systèmes intégrés comme UnivPortal (dérivé du cartable-électronique, développé initialement par l'université de Chambéry [20]) et le CMS Plone.

Serveur sous Debian GNU/Linux

Sans rentrer dans les détails, les serveurs que nous utilisons utilisent tous le système d'exploitation Debian GNU/Linux. Nous l'apprécions pour sa stabilité, sa robustesse et sa puissance de traitement. La gestion du RAID est assurée logiciellement et permet de surperformer les solutions matérielles classiques.

Serveur XML Natif eXist

EXist [19] est une base de données XML native libre, programmée en Java. Elle gère les requêtes XQuery, ainsi que XUpdate et XInclude. Elle communique via XMLRPC, XML:DB API, SOAP, WebDAV ou HTTP. Elle peut utiliser plusieurs types de backend (interface native XML, MySQL, Oracle, PostgreSQL) pour stocker les données XML. EXist peut fonctionner en processus indépendant ou en servlet. Les performances sont excellentes. Il a été calculé que sur une journée d'exploitation typique en production les bases XML natives concurrentes (produits propriétaires connus compris) sont 10% moins performantes. On mesure

l'intérêt de cette solution lorsque l'on compare son coût (nul pour la licence, insignifiant pour le déploiement et l'apprentissage) avec celui des solutions propriétaires du même segment. La vivacité de la communauté autour du développement de ce logiciel est un gage de pérennité et de qualité.

Web Application Server Zope

Zope [21] est un serveur d'application web libre, programmé avec le langage Python et orienté objet. Il s'appuie sur une base de données objets transactionnelle (ZODB) pour assurer la persistance des objets. Zope propose deux langages de développement de pages web dynamiques, le DTML et le TAL/METAL, et sait en intégrer d'autres (XML, CGI...). Zope est facilement extensible grâce à l'ajout de « produits » écrits en Python et il sait communiquer via plusieurs standards (XMLRPC, FTP, WebDAV...) en plus de sa fonction de serveur web. Il peut déléguer l'authentification et le stockage des rôles des utilisateurs à n'importe quel système d'annuaire. Utilisé aussi bien par l'OTAN que le ministère français de l'intérieur ou des entreprises comme SGI, c'est un outil de choix pour constituer un socle conforme au SDET. Pour garantir la robustesse d'une application web, il est indispensable de s'appuyer sur une structure correctement modélisée en terme de développement et de sécurité. Ainsi l'utilisation d'un serveur d'application s'impose face à un simple langage de programmation web comme PHP. Le choix du serveur d'application est dicté principalement par les contraintes d'efficacité et la base applicative préexistante. Deux raisons nous ont fait préférer Zope aux serveur d'application basés sur Java. Tout d'abord, Zope s'appuie sur le langage Python qui est largement plus efficace que Java (temps de développement beaucoup plus courts, code beaucoup plus concis, exécution plus rapide) [22]. Ensuite Zope est en phase de croissance exponentielle et dispose d'un catalogue d'applications de tout premier ordre, en particulier des systèmes de gestion de contenu complets soutenus par une communauté d'une exceptionnelle vigueur.

UnivPortal

UnivPortal (dérivé du Cartable Électronique) a été développé par la cellule TICE de l'Université de Savoie en 1999, suite à un appel à projet du ministère de l'éducation nationale. Il s'agissait de créer un Environnement Numérique de Travail (ENT), qui a été réalisé à partir des propositions de plusieurs chercheurs du laboratoire Syscom. Le but est de donner un accès intégré à l'ensemble du système d'information à tous les personnels de l'établissement (ou de l'ensemble d'établissements) où UnivPortal est déployé. La plate-forme fournit les services de gestion de contenus (workflows de publication, gestion de groupes d'utilisateurs, système de journal...), de communication (messagerie, chat, casier intra-portail...) et des outils de scolarité (agenda, livret de notes...). Elle s'appuie sur l'annuaire de l'établissement pour la gestion des utilisateurs. UnivPortal est basé sur Zope, et en tire son modèle de sécurité. De nombreux développements spécifiques en Python, DTML et JavaScript ont été effectués. En production le système passe très bien à l'échelle et l'équipe de développement (maintenant au sein de la société Pentila [23]) est extrêmement réactive.

Plone

Plone [18] est un système de gestion de contenus basé sur Zope. Il s'installe comme un produit et ajoute de nombreuses fonctionnalités à Zope. De même que Zope, il est extensible via des produits écrits en Python. Il est très simple d'usage et ne nécessite pas de connaître Zope ou un quelconque langage de programmation. Plone est utilisé par de nombreuses organisations dont la NASA ou encore des ministères. En particulier le ministère des affaires étrangères lui fait confiance pour son système de communication inter-ambassades. A noter que Plone sert de base de développement à la prochaine version majeure de Zope (qui donc incorporera plus de fonctionnalités de CMS de haut niveau).

Connecteurs

Au delà des API nous utilisons des middleware (une fois de plus pour se conformer au SDET) dans les relations entre les serveurs impliqués dans notre architecture. Par exemple, comme indiqué précédemment, nous avons implémenté un connecteur Zope pour eXist. Celui-ci permet de rendre les applications web indépendantes de la base XML utilisée pour le stockage. Il permet aussi l'agrégation transparente de plusieurs serveurs eXist derrière le même connecteur Zope

API

Nous augmentons la robustesse de l'ensemble du système de gestion en rendant les différents composants indépendants via une séparation par des API (Application Programming Interface). Nous avons rédigé d'une part une API Python pour interagir avec le serveur eXist de manière générale et une API Python pour permettre à Plone2 d'utiliser eXist comme backend pour les métadonnées. Il va de soit que cette dernière API s'appuie sur la précédente.

Développement en Sources Ouvertes

Au delà des aspects logiciels et architecturaux, la démarche d'ouverture, normalisation et standardisation n'est efficace que si les principes de développement et les modes humains qui l'accompagnent sont de la même philosophie. Ainsi en ouvrant et partageant nos codes nous avons pu développer toutes les briques logicielles manquantes avec très peu de moyens (2 personnes) et une efficacité maximale. Nous avons pérennisé notre travail en reversant nos améliorations logicielles à la communauté du logiciel libre. En utilisant les outils et les méthodes de développement de cette communauté nous avons pu maintenir une qualité optimale de notre système de gestion de contenus pédagogiques et des temps de réponses aux

sollicitations des utilisateurs hors de portée des logiciels propriétaires.

Disponibilité des Logiciels et du Schéma XML

La publication des logiciels et schéma XML présentés dans cette article sera officialisée en septembre 2004. Les produits pour Zope et Plone seront annoncés sur zope.org, plone.org et sur eduplone.net. Les produits pour la base XML eXist seront disponibles sur le site de celle-ci sur sourceforge.net [19].

Conclusion

La démarche d'ouverture, normalisation et standardisation présentée ici est une bonne assurance pour les projets dont la pérennité est essentielle. Son application est simple et d'une efficacité redoutable. Elle permet de potentialiser au maximum les ressources dédiées au projet. Ainsi, malgré une équipe restreinte, le service PRATIC de l'ENS Lyon a pu mettre en place un LCMS taillé pour le long terme, complètement interopérable et disponible, du modèle à l'implémentation, pour la communauté. La nature ouverte du système permet d'apporter les améliorations nécessaires rapidement et efficacement en fonction des contraintes des utilisateurs. Ces derniers ont été continuellement associés dans les phases de maîtrise d'ouvrage et de beta test. La démarche d'ouverture étant particulièrement bien perçue, nous avons pu sans difficulté recadrer en permanence les évolutions souhaitées. Ainsi les différents acteurs de l'ENS Lyon impliqués dans la réalisation du LCMS sont confiants pour la phase de mise en production qui débutera à la rentrée 2004 et surtout pour l'avenir des solutions proposées, qui dans le pur esprit du logiciel libre, sont destinées à être enrichies et redistribuées par la communauté.

Glossaire

API : Application Program Interface. Une API définit la manière dont un composant informatique peut communiquer avec un autre. Cela permet de réaliser une couche d'abstraction entre deux systèmes, masquant au système appelant les détails de fonctionnement interne du système appelé.

CSS : Cascading Style Sheets. Candidate W3C Recommendation actuellement en version 2.1. Langage utilisé pour décrire la présentation d'un document structuré écrit en HTML ou en XML.

DocBook : DocBook fournit un système pour écrire des documents structurés en SGML ou en XML. Il est particulièrement utilisé pour écrire des livres et des articles traitant d'informatique mais peut être utilisé pour tout type de document.

DTML : Document Template Markup Language. Langage permettant de générer des informations textuelles dynamiques au travers d'un document modèle (template) par appel à des applications Zope. Remplacé progressivement par ZPT.

LOM : Learning Object Metadata. IEEE Standard 1484.12.1-2002. La représentation en XML du LOM est elle-même un standard IEEE (Standard 1484.12.3).

MathML : Mathematical Markup Language. Recommandation du W3C. MathML permet d'encoder les notations mathématiques dans un document XML, afin de simplifier les échanges dans le domaine des mathématiques.

SGML : Standard Generalized Markup Language. Norme ISO 8879 :1986. Ce méta-langage permet de définir, indépendamment de tout système, des méthodes de représentation de textes sous forme électronique.

SMIL : Synchronized Multimedia Integration Language. Recommandation du W3C. L'objectif de SMIL est de permettre l'intégration de contenus multimédias diversifiés (images, sons, textes, vidéos, animations, flux de texte) en les synchronisant afin de permettre la création de présentations multimédias.

XML : eXtensible Markup Language. Recommandation du W3C. Ce méta-langage, dérivé de SGML, sert de base pour créer des langages balisés spécialisés.

ZPT : Zope Page Template. Langage succédant à DTML pour la génération de pages web dynamiques dans Zope. La syntaxe ZPT est conforme à XML.

Références

[1] https://formation.ens-lyon.fr

[2] http://culturesciencesphysique.ens-lyon.fr

[3] http://www.ens-lyon.fr/Planet-Terre

[4] http://www.u-escales.org

[5] http://dicos.ens-lyon.fr

[6] http://www.w3.org/TR/smil20

[7] http://wam.inrialpes.fr/software/limsee2

[8] http://www.educnet.education.fr/equip/sdet.htm

[9] http://www.oasisopen.org

[10] http://www.docbook.org

[11] http://dyomedea.com

[12] Van der Vlist, E. 2004a, Lightweight derivation and translation of xml vocabularies. In Proceedings of Extreme Markup2004. Montreal, Quebec, Canada.

[13] http://www.w3.org/Math

[14] http://www.altova.com/products_doc.html

[15] http://www.xmlmind.com/xmleditor

[16] http://www.oxygenxml.com.

[17] http://www.jedit.org

[18] http://plone.org.

[19] http://exist.sourceforge.net

[20] http://www.pentila.com/References

[21] http://www.zope.org.

[22] http://www.python.org/doc/Comparisons.html.

[23] http://www.pentila.com