Des métadonnées pour la description des composants logiciels pédagogiques

Issam Rebaï, Jean-Marc Labat

Des métadonnées pour la description des composants logiciels pédagogiques

Issam REBAÏ[1, 2], Jean-Marc LABAT[1]
[1] Laboratoire CRIP5, Université Paris V, équipe AIDA
45, rue des Saints-pères 75270 Paris Cedex 06
[2] ESIEA Recherche
9, rue Vésale 75005 Paris
{Issam.Rebai, Jean-Marc.Labat}@math-info.univ-paris5.fr

Résumé

Les développements logiciels issus de la recherche sur les EIAH sont souvent sous-utilisés. Pour changer cet état de fait, il faut d'une part développer les logiciels sous forme de composants et d'autre part réaliser une plate-forme de mutualisation permettant de les stocker et de les assembler. Dans cet article, après avoir rappelé la définition des composants, nous nous intéressons à ceux qui sont spécifiques au domaine des EIAH. Afin de mutualiser leur utilisation, nous proposons de les classer en 4 catégories et pour la principale, un schéma de description est proposé. Ce schéma reprend certains éléments de la LOM (Learning Object Metadata) mais la description sur le plan pédagogique est plus précise puisque ces composants logiciels sont destinés à être assemblés pour construire différents EIAH.

Mots-clés : Standard, composant logiciel, métadonnées, réutilisabilité, description, indexation, partage.

Abstract

Software developments in the area of the e-learning research have been often underused. To change this it is necessary to develop them as software components and to design a common platform to handle their storage and assembling. In this paper we will remind first the definition of the components and will then focus on some specific component to the field of e-learning ones. In order to unify their usage, we propose to classify them in 4 categories. A description diagram of the main category is proposed. It takes advantage of some LOM (*Learning Object Metadata*) elements, but it extends and develops further the descriptions at a pedagogical level, since these components are intended to be used as software bricks for constructing different e-learning software.

Keywords: Standard, software component, metadata, reusability, description, indexation, sharing.

Introduction

Le besoin croissant de formation de la part des milieux académiques et professionnels a renforcé la demande en EIAH. Ainsi de nombreux projets de recherche ont vu le jour pour concevoir et développer des environnements d'apprentissage ou des outils permettant de mieux tirer profit de la puissance des ordinateurs et des réseaux afin d'offrir de nouvelles fonctionnalités.

Si les résultats de ces recherches sont intéressants, souvent le produit final reste inexploité. En effet, la majorité des projets de recherche n'ont pas pour objectif de construire un EIAH mais de se pencher sur un problème particulier et d'essayer de lui trouver une solution. Le résultat est souvent une application ou un prototype rendant un service particulier. Malheureusement, et faute d'intégration dans un EIAH opérationnel, le fruit de ces recherches est souvent voué aux oubliettes ou, dans le meilleur des cas, à une utilisation limitée à quelques entités d'enseignement. D'un autre coté, tester et faire fonctionner un outil obligent généralement les chercheurs à développer des programmes complémentaires souvent disponibles ailleurs mais non référencés. Pour mieux valoriser les travaux de recherche et éviter de re-développer plusieurs fois les mêmes fonctionnalités, la première condition est donc que les réalisations logicielles soient décrites, indexées et récupérables. Mais cette condition n'est pas suffisante. Une autre difficulté est liée au fait que les logiciels développés ne le sont pas sous forme de composants, ce qui rend la réutilisabilité difficile.

C'est dans ce contexte que notre travail intervient et tente de trouver une solution sur le premier point pour améliorer la coopération entre les laboratoires informatiques travaillant sur les environnements d'apprentissage. Nous espérons contribuer à la création d'une synergie pour le développement des EIAH et en faire bénéficier tous les acteurs désirant coopérer.

Nous proposons pour cela la construction de bibliothèques assurant la description et l'indexation des programmes produits par les laboratoires. Nous visons ainsi à mettre à la disposition de la communauté scientifique le moyen de faire connaître et de valoriser leurs produits.

Dans cet article, après avoir rappelé le concept de composant, nous en définissons quatre catégories pour les EIAH. Ensuite, nous présentons le schéma de métadonnées que nous proposons pour la description, l'indexation et l'assemblage des composants logiciels pédagogiques.

Le concept de composant

Le concept de composant logiciel [1] ne date pas d'aujourd'hui. Il a été utilisé pour la première fois en 1968 dans le Workshop NATO Software Engineering par Doug McIllroy lors de la présentation de son article intitulé "Mass produced Software Component". Ce visionnaire prévoyait que la production massive de

composants logiciels allait résoudre la crise du logiciel [2].

Durant ces trois décennies, plusieurs définitions ont été attribuées au terme composant logiciel. Même si ces définitions sont différentes selon le contexte et les technologies dans lesquelles elles ont été annoncées, elles ont, au fond, le même point de convergence : la réutilisabilité [3 ; 4]. Par réutilisation, on entend la possibilité de construire une nouvelle application en récupérant le code et les bouts de programmes développés auparavant. Cette réutilisation peut être compliquée ou facilitée selon le type d'élément qu'on tente de récupérer. En effet, reprendre un extrait de programme, y apporter des modifications et l'intégrer dans un nouveau programme n'est pas aussi facile que configurer un élément générique et le connecter avec d'autres éléments. La question qui se pose alors est la suivante : « qu'est ce qu'un composant ? »

Selon Allen [5], « un composant est une unité exécutable ayant la forme d'une boîte noire encapsulant les services qu'elle fournit. Ces services ne sont accessibles que par les interfaces publiées correspondantes et ce à travers un standard d'interaction ».

Brown précise qu'un composant est un fragment utile d'un système logiciel pouvant s'assembler avec d'autres fragments pour former des pièces plus grandes d'applications complètes. Brown rajoute qu'un composant doit être organisé sous la forme d'un package distribuable et livrable [6].

Quant à nous, nous retenons la définition suivante : un composant logiciel est une entité pouvant être contrôlée dynamiquement et assemblée pour former des applications ou des composants. C'est une unité comportant des instructions de traitement nécessaire à l'application la contenant.

Remarquons que ces instructions de traitement doivent être à l'origine d'un processus ou d'une tâche. Ainsi, un code XML, une requête SQL ou des règles d'un moteur d'inférence ne sont pas des composants logiciels mais des données transmises à des éléments capables de les manipuler appelés composants.

Les composants en EIAH

Le secteur d'activité d'un composant est un des premiers critères de classification possible. Il permet de définir le domaine d'application et ses objectifs, de recenser les besoins fonctionnels en composants et de fournir un cadre conceptuel harmonisé. Nous introduisons ainsi, dans le cadre des EIAH, quatre types de composants logiciels détaillés dans les paragraphes suivants : les composants logiciels pédagogiques (CLP), les composants logiciels de services(CLS), les composants logiciels techniques (CLT) et les composants logiciels de fabrication (CLF).

Les composants logiciels pédagogiques

Un composant logiciel pédagogique (CLP) est un composant métier apportant une plus-value pédagogique. Il participe directement ou indirectement au processus d'apprentissage humain. C'est un composant métier réutilisable essentiellement dans les EIAH. Des exemples typiques sont les gestionnaires de scénarios pédagogiques, les simulateurs, les résolveurs de problèmes à vocation pédagogique ou les outils d'évaluation de l'apprenant. En génie logiciel, un CLP fait partie de ce qu'on appelle les composants verticaux.

Généralement, les auteurs parlent de « composants pédagogiques » pour désigner les ressources pédagogiques ou les objets pédagogiques [7 ; 8 ; 9]. Une ressource pédagogique est un élément numérique contenant une information destinée à l'enseignement et susceptible d'intervenir dans la constitution d'un cours et ce, indépendamment de sa taille physique et de son format numérique. Un objet pédagogique est un composant constitué d'un ensemble d'objets et/ou de ressources ayant un objectif pédagogique et une durée déterminée [10].

Comme le montre la figure 1, nous proposons d'ajouter à ces 2 catégories les CLP qui sont aussi, par définition, des composants qu'il est possible d'intégrer dans un cursus de formation ou un cours. L'inclusion est stricte car un composant pédagogique ne peut jouer le rôle d'un CLP que s'il comporte des instructions de traitements à l'origine d'un processus machine reflétant un comportement interne. De ce fait, un cours html en ligne n'est pas un CLP alors qu'une applet de simulation d'une expérience en physique l'est.

Pour désigner le sous-ensemble constitué des ressources pédagogiques et des objets pédagogiques qui ne sont pas des CLP, nous proposons d'appeler ce sous-ensemble « contenus pédagogiques » (voir fig 1)

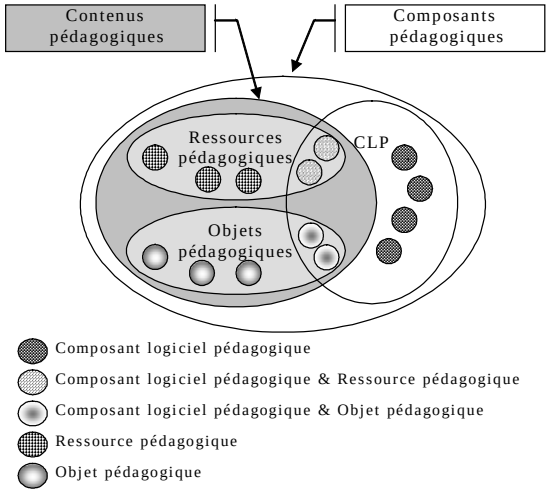

Fig 1 – CLP & Composants pédagogiques.

Pour décrire les contenus pédagogiques, nous disposons de la LOM et de plusieurs standards propriétaires comme le Dublin Core [11], SCORM [12], etc. Pour les composants logiciels pédagogiques, ces standards ne sont pas adaptés car ils ne comportent pas suffisamment d'informations pour favoriser leur réutilisation. Pour cela, nous proposons dans cet article un ensemble de métadonnées spécialisées pour les décrire.

Les trois autres types de composants, CLT, CLS, CLF sont présentés ci-dessous rapidement mais ils ne sont pas concernés par le schéma de description que nous présentons dans cet article.

Les composants logiciels de service

Un composant logiciel de service (CLS) est un composant métier apportant une plus-value fonctionnelle pouvant rendre services aux CLP ou aux utilisateurs de l'EIAH. C'est un composant utile mais pas nécessaire, pour garantir la vocation pédagogique de l'EIAH. Il offre des services annexes et assure une meilleure ergonomie et maniabilité pour les utilisateurs. C'est le cas, par exemple, des synthétiseurs vocaux, des adaptateurs d'interfaces ou des outils de communication (messagerie, chat, forum, etc.). En génie logiciel, un composant logiciel de service fait partie des composants transversaux. Il est utilisable dans les EIAH mais aussi dans d'autres domaines d'applications.

Les composants logiciels techniques

Un composant logiciel technique (CLT) est un composant apportant des fonctionnalités non pédagogiques aux EIAH. Il fournit les mécanismes de base assurant le bon fonctionnement des CLP et CLS. C'est un composant susceptible d'être utilisé dans tous les domaines. En génie logiciel, il fait partie de ce qu'on appelle les composants horizontaux.

Ces composants seront principalement utilisés pour la construction de l'infrastructure des plates-formes EIAH ou comme intermédiaires entre des CLP et des CLS peu interopérables. C'est le cas, par exemple, des composants de persistance (sauvegarde des données), d'impression, de formatage de texte (transformation d'un texte XML en pdf), d'adaptateur de structures de données, etc.

Le constructeur d'EIAH disposera alors d'une large palette de petits composants utiles lui facilitant sa tâche. La majorité de ceux-ci proviendront des environnements de développement ou des librairies des langages de programmation.

Les composants logiciels de fabrication

Un composant logiciel de fabrication (CLF) est un composant métier servant à construire et modifier des composants pédagogiques. Ce type de composant n'intervient pas lors de l'exploitation de l'EIAH mais lors de la construction des cursus de formation.

Selon l'architecture de l'environnement d'apprentissage, ce type de composant peut y être intégré ou mis à l'extérieur dans des outils auteurs. Dans le premier cas, il devient possible au formateur d'agir sur les constituants du cursus de formation et de les adapter immédiatement à son contexte d'utilisation. Dans le deuxième cas, il n'est possible d'agir sur ces constituants qu'en amont de la phase d'exploitation du cursus.

En génie logiciel, un CLF fait partie de ce qu'on appelle les composants verticaux.

LSCM, une proposition de standard pour décrire les CLP

La LSCM (Learning Software Component Metadata) est le schéma de métadonnées que nous proposons pour décrire les composants logiciels pédagogiques. Pour l'élaborer, nous avons étudié standards, normes et modèles utilisés pour la description des logiciels, des composants, des documents ou des ressources pédagogiques. Nous avons aussi observé quelques sites marchands ou logithèques de logiciels et de composants COTS [13].

Lors de cette étude, nous avons analysé, comparé et groupé les métadonnées utilisées dans la LOM, le DublinCore, le standard BIDM [14] (Basic Interoperability Data Model) du IEEE, les notes du W3C sur l'OSD [15] (Open Software Description) et les données utilisées dans les sites décrivant des logiciels.

En se basant sur l'analyse des besoins de la communauté EIAH et de ses recommandations, nous avons procédé à plusieurs itérations de filtrage et de classification des métadonnées recueillies. Des métadonnées que nous avons, par ailleurs, complétées par des champs spécifiques aux EIAH et à l'attente des utilisateurs de composants. Ce travail a été fait de manière pluridisciplinaire associant des chercheurs en informatique et des chercheurs en didactique et en sciences de l'éducation.

La LSCM s'inspire largement de la LOM, dans la mesure où ces descriptions sont en partie similaires. En fait, à terme, il faudrait définir une structure de description générique, utilisable pour tous les composants pédagogiques. La LSCM ou la LOM en seraient alors des spécialisations. Cependant, actuellement, si leur structure se ressemble et si certains champs sont communs, la LSCM et la LOM diffèrent néanmoins sur beaucoup de points. La LSCM est divisée en deux sections : Une section commune avec les autres catégories (CLS, CLT, CLF) et une section spécifique. Chaque section comporte un ensemble de métadonnées structurées hiérarchiquement sous forme d'un arbre. Avant de lister les métadonnées, nous décrivons, dans le paragraphe suivant, comment celles-ci sont elles-mêmes décrites.

Descripteurs des métadonnées

Toutes les métadonnées définies sont elles-mêmes décrites par un ensemble de descripteurs. À chaque métadonnée sont associés un nom, un identifiant, une description, un nombre d'occurrences minimal et maximal et un type de données.

- Le nom spécifie l'information qui sera apportée par la métadonnée.
- L'*identifiant* associe un code unique à chaque métadonnée. Il servira pour réaliser des traitements automatiques sur la description et pourra remplacer le chemin absolu d'une métadonnée dans l'arborescence.
- La *description brève* précise la signification du nom si ce dernier est ambigu. Il permet aux utilisateurs du schéma de description d'avoir une idée sur le rôle de la métadonnée sans consulter la documentation fournie avec le schéma de description.
- Le *type de donnée* détermine si la métadonnée est une structure, donc un nœud de l'arbre ou si c'est une feuille de l'arbre, donc un contenu. Pour ce dernier cas, cette information précise le type d'informations qu'il est possible de mettre : un texte, une chaîne de caractères, un entier, une date, une url, etc. Le type peut être spécifié par des contraintes permettant de préciser la taille minimale et maximale du texte, les

plages de valeurs possibles pour une donnée numérique, le format, etc.

- Le *nombre d'occurrences minimal* détermine combien de fois, au minimum, peut exister une instance de la métadonnée dans un document de description d'un composant. Si ce nombre est égal à 0, cela signifie que la métadonnée est optionnelle. S'il est supérieur ou égal à 1, cela signifie qu'elle est obligatoire.
- Le *nombre d'occurrences maximal* détermine combien de fois, au maximum, peut exister une instance de la métadonnée dans une description d'un composant. Si ce nombre est égal à 0, cela signifie que la métadonnée n'existe plus et a été retirée du schéma. S'il est égal à un, cela signifie qu'il existe une seule instance de la métadonnée dans la description. S'il est égal à une valeur entière n supérieure à 1, cela signifie qu'il peut exister au maximum n instances. Enfin, s'il comporte le mot-clé « unbounded » c'est qu'il peut exister une infinité d'instances. Des exemples sont fournis ci-dessous en même temps que la liste des métadonnées.

Dans la LSCM, après réflexion, nous n'avons pas déclaré les descripteurs des métadonnées ayant trait à leur remplissage. Nous visons ainsi à séparer les propriétés stables de la description de celles qui peuvent évoluer ou changer pour s'adapter à un contexte particulier. Ces informations complémentaires devront être renseignées lors de la mise en œuvre du système gérant ces métadonnées. Il s'agit de :

- *Contrôle source de donnée* : propriété précisant si le contenu de la métadonnée est libre (peut être remplie par n'importe quelle information), restreint à une liste fermée de termes ou restreint à une liste ouverte de termes, c'est-à-dire extensible par des termes inexistants dans le vocabulaire.
- *Vocabulaire* : ou liste de valeurs définissant l'espace de valeurs d'une métadonnée. Ce vocabulaire peut se présenter sous la forme d'une liste ou d'une taxonomie de termes. À titre informel et pour concrétiser la signification d'une métadonnée, nous mettons des exemples de valeurs pouvant faire partie de son vocabulaire.
- *Obligation* : propriété précisant si le renseignement de la métadonnée est obligatoire, facultatif, recommandé ou verrouillé. Si la métadonnée est recommandée, cela signifie que l'information est jugée pertinente pour le système. Enfin, si elle est verrouillée, l'utilisateur ne doit pas renseigner ce champ car il sera rempli automatiquement ou par une autre personne.
- *Stabilité* : propriété permettant de renseigner sur la fréquence de modification de contenu d'une métadonnée. Trois valeurs sont possibles : figé, faible évolutivité ou forte évolutivité.
- La *présentation de la métadonnée* : ensemble de propriétés servant à présenter l'information aux utilisateurs au niveau de l'IHM. Il s'agit du libellé de la métadonnée, de sa définition, de son explication ou de ses exemples. L'objectif étant de pouvoir utiliser le même concept, même s'il est nommé différemment (par exemple dans différentes langues).

Présentation succincte de la section commune

Cette section commune comporte une partie des rubriques habituelles que nous retrouvons dans la LOM. Plus généralement, cette section peut décrire n'importe quelle catégorie de composants logiciels car elle ne contient que des données sur les caractéristiques du composant, indépendamment de son domaine et de sa spécialité. Nous donnons à titre indicatif les douze grandes rubriques de métadonnées formant cette section ainsi que leur rôle :

1. Rubrique « Générale » : regroupant des informations d'identification et de description circonscrit du composant.
2. Rubrique « Meta-MetaData » : comportant des renseignements sur les métadonnées décrivant le composant, comme par exemple la version du schéma de métadonnées utilisée.
3. Rubrique « Cycle de vie » : décrivant l'état du composant, sa version et l'historique de son évolution.
4. Rubrique « Contact » : permettant de trouver les coordonnées du fournisseur du composant et des services qu'il met à disposition pour les utilisateurs.
5. Rubrique « Information Technique » : spécifiant les contraintes techniques aussi bien matérielles que logicielles nécessaires pour exploiter le composant.
6. Rubrique « Droit » : permettant de déclarer les droits de propriété intellectuelle et les conditions commerciales d'utilisation du composant.
7. Rubrique « Relation » : énumérant les composants qui doivent être connectés avec le composant lors de l'assemblage afin que ce dernier puisse fonctionner.
8. Rubrique « Caractéristique composant » : listant les propriétés et les services offerts par le composant de point de vue informatique.
9. Rubrique « Documentation » : comportant des informations sur les documents ou les manuels joints au composant.
10. Rubrique « Annotation » : regroupant des notes (de critique, de recommandation ou de suggestion) déposées par les personnes ayant manipulé le composant.
11. Rubrique « Extension » offrant le moyen de définir, de classifier le composant ou d'apporter des métadonnées supplémentaires.
12. Rubrique « Évaluation » permettant de noter le composant selon des critères de test comme la performance, l'ergonomie, etc.

Cette section est donc un dénominateur commun de métadonnées pour toutes les catégories de composants logiciels. Elle comporte suffisamment d'informations pour pouvoir assembler le composant, le configurer et le déployer sur une architecture matérielle et logicielle. En revanche, pour l'utiliser, il est nécessaire de compléter ces informations avec celles de la section spécifique.

Section spécifique aux CLP

Nous allons maintenant détailler les métadonnées spécifiques aux CLP qui sont celles sur lesquelles nous avons plus particulièrement travaillé. Pour chaque

métadonnée, nous présentons entre parenthèses ses caractéristiques : type, nombre minimal d'occurrences et nombre maximal d'occurrences. Dans la suite, les titres des sous paragraphes sont les noms des métadonnées. L'identifiant, ne présentant pas d'intérêt particulier, sera omis. Cette section comporte six rubriques :

Catégorie

Propriétés : (Chaîne de caractères, 1, unbounded).

La catégorie précise la « classe d'environnements informatiques » dédiés à l'apprentissage (tuteur intelligent, micro-monde, environnement support de pédagogie de projet, etc.) [16] dans laquelle le composant peut être rangé. Cette classification ne se fait pas par rapport aux disciplines mais par rapport à la stratégie pédagogique sous-jacente que le composant est capable de mettre en œuvre. Si un composant appartient à plusieurs classes d'environnements, il aura dans sa description plusieurs instances de cette métadonnée. Pour faciliter la description d'un composant, une liste plate de catégories peut être liée à cette métadonnée.

Utilisation

Propriétés : (Structure, 1,1).

Cette rubrique précise les conditions d'exploitation prévues a priori dans lesquelles le composant peut être utilisé. Elle comporte les métadonnées suivantes :

- EspaceTemps (Chaîne de caractères, 1, unbounded) : permet de savoir dans quels cadres d'enseignement il est possible d'utiliser le composant. Un cadre est défini par l'axe spatial (Environnement d'apprentissage à distance ou en présence) et par l'axe temporel (Environnement d'apprentissage synchrone ou asynchrone). Une liste hiérarchisée comportant les termes : Formation en ligne, Formation à distance, formation par alternance… peut être associée à cette métadonnée.
- Relation_Inter_Apprenants (Chaîne de caractères, 1, unbounded) : permet de savoir dans quelles situations d'apprentissage il est possible d'utiliser le composant. Plus précisément, cette métadonnée permet de répondre à la question suivante : Quels types de rapports peuvent avoir les apprenants entre eux en utilisant le composant ? Une liste hiérarchisée comportant les termes : usage individuel, en groupe, en compétition ou en coopération… peut être associée à cette métadonnée.
- Commentaire : (Chaîne de caractères, 0, 1) est une métadonnée facultative permettant d'apporter des renseignements supplémentaires sur les contextes ou les situations d'apprentissages dans lesquels le composant est utilisable.

Pédagogie

Propriétés : (Structure, 1, unbounded)

Cette rubrique présente le composant selon l'axe pédagogique. Il s'agit de donner des indications sur la pédagogie implémentée par le composant et le contexte pédagogique dans lequel il est prévu de l'utiliser.

Si le composant peut être configuré pour adopter des comportements pédagogiques différents, plusieurs instances de « Pédagogie » doivent être spécifiées. Nous avons sous cette rubrique les métadonnées suivantes :

- Modèle (Chaîne de caractères, 1, 1) : C'est le modèle pédagogique auquel les auteurs du composant se sont référés pour construire les services offerts par le composant. Cette métadonnée oriente l'organisation et l'action pédagogique. Comme modèles, on peut avoir : le modèle constructiviste, institutionnel, transmissif, appropriatif, etc. [17]
- Stratégie (Chaîne de caractères, 1, unbounded) : Pour un modèle pédagogique donné, la stratégie pédagogique spécifie la méthode (ensembles de démarches formalisées et appliquées) adoptée pour atteindre le but du composant. Comme stratégie, on peut avoir une stratégie de découverte guidée, de découverte avec médiation, de résolution de problèmes, d'exposition simple, d'exposition avec dialogue, etc. Bien sûr, la stratégie n'est pas complètement indépendante du modèle indiqué dans la rubrique précédente mais à un modèle donné peuvent, malgré tout, correspondre plusieurs stratégies.
- Commentaire : (Chaîne de caractères, 0, 1) est une métadonnée facultative permettant d'apporter des consignes et/ou des explications sur le choix pédagogique.

Didactique

Propriétés : (Structure, 1, 1)

Cette rubrique présente le composant selon l'axe didactique. Il s'agit de renseigner les disciplines, les savoirs et les ressources pédagogiques manipulables par le composant.

- Domaine_Enseignement (Chaîne de caractères, 1, unbounded) permet d'énumérer les disciplines ou les sous disciplines manipulées par le composant. Une arborescence des disciplines doit être associée à cette métadonnée. La spécification d'une sous discipline précise (feuille de l'arbre) est privilégiée afin d'affiner la description. Par contre, la spécification d'une discipline générale (nœud de l'arbre) signifie implicitement que le composant gère toutes les sous disciplines de ce nœud. Si le composant manipule des disciplines spécifiques n'appartenant pas à la même discipline générale, il faut lui associer plusieurs instances de cette métadonnée. Comme domaines d'enseignement au premier niveau de l'arbre, on peut par exemple trouver : Mathématique, Physique ou Anglais, et comme sous-domaine : Algèbre, Géométrie, Géométrie 3D, Sphère, Verbe irrégulier en anglais.
- But_Didactique (Chaîne de caractères, 1, unbounded) définit les effets attendus ou visés par les actions de formation du composant ou par son utilisation. Les buts sont déclarés avec des phrases comportant un verbe d'action de type « être capable de ». Par exemple, on peut indiquer : « Maîtriser la résolution des équations de second degré ».
- Savoir (Structure, 1, 1) est une structure énumérant l'ensemble des connaissances théoriques, pratiques

ou comportementales auxquelles le composant peut faire appel. Il s'agit des savoirs manipulés par le composant ou par ses utilisateurs. Cette structure est constituée des métadonnées suivantes :

- Pré-requis (Chaîne de caractères, 0, unbounded) : ce sont les savoirs, savoir-faire et savoir-être exigés pour pouvoir manipuler efficacement le composant ou profiter de ses services.
  - Savoir_Enseigné (Chaîne de caractères, 0, unbounded) : ce sont les connaissances théoriques manipulées, enseignées ou transmises par le composant.
  - Savoir-Faire_Enseigné (Chaîne de caractères, 0, unbounded) : ce sont les connaissances pratiques manipulées, enseignées ou transmises par le composant. Ces connaissances sont relatives à la mise en œuvre d'un savoir théorique et d'une habilité pratique maîtrisée dans une réalisation spécifique.
  - Savoir-Être (Chaîne de caractères, 0, unbounded) : ce sont les connaissances de type savoir-faire relationnel entreprises, enseignées ou transmises par le composant. Ces connaissances concernent les comportements et les attitudes à entreprendre dans une situation donnée.
  - Savoir-Implicite (Chaîne de caractères, 0, unbounded) : ce sont les connaissances théoriques, manipulées ou non par le composant, que l'apprenant doit de lui-même découvrir car elles ne lui sont pas explicitement transmises.
  - Savoir-Faire_Implicite (Chaîne de caractères, 0, unbounded) : ce sont les connaissances pratiques, manipulées ou non par le composant, que l'apprenant doit de lui-même découvrir car elles ne lui sont pas explicitement montrées.
  - Savoir-Être_Implicite (Chaîne de caractères, 0, unbounded) : ce sont les connaissances de type savoir-faire relationnel, adoptées ou non par le composant, que l'apprenant doit de lui-même avoir car elles ne lui sont pas explicitement indiquées.
- Ressource (Structure, 0, unbounded) est une structure permettant de renseigner les types de ressources pédagogiques que le composant manipule. Elle comporte les métadonnées suivantes :
  - Description (Chaîne de caractères, 1, 1) permet de renseigner de façon synthétique le type des ressources pédagogiques (format ou contenu). « Propriétés en XML d'une pièce mécanique à dessiner en trois dimensions » ou « QCM à soumettre aux apprenants » en sont des exemples.
  - Lien_Informations (Structure, 0, 1) est une structure optionnelle spécifiant une adresse url pour avoir accès à des informations complémentaires sur le type des ressources pédagogiques décrit dans « description ». Cette structure comporte une métadonnée « Adresse » (URL, 1, unbounded) et une métadonnée « SOS » (Contact, 1, 1) permettant de contacter un individu.
  - Exemple (Structure, 0, unbounded) est une structure donnant des références d'exemples concrets de ressources pédagogiques manipulable par le composant. Cette structure, que nous ne détaillons pas, comporte des informations pour accéder aux métadonnées (LOM pas exemple) des exemples des ressources pédagogiques.
- Recommandation (Chaîne de caractères, 0, unbounded) est une métadonnée facultative permettant de proposer des recommandations didactiques aux responsables de l'exploitation du composant.

Exploitation

Propriétés : (Structure, 0, 1)

Si le composant propose des services à des acteurs humains, cette rubrique permet de les énumérer. L'objectif n'est pas d'expliciter comment les exploiter mais de préciser les interactions que peut avoir un acteur avec le composant. La structure « Exploitation » comporte les éléments suivants :

- Service (Structure, 1, unbounded) permet d'énumérer l'ensemble des services offerts par le composant. Seuls les services destinés aux acteurs humains sont concernés par cette rubrique. Les informations mentionnées dans cette structure sont :
  - Nom (Chaîne de caractère, 1, 1) décrit sémantiquement le service offert par le composant. Ce nom peut être une phrase telle que « Proposer des questions pour assister l'apprenant dans la démonstration de la perpendicularité de deux droites » ou « Dessin de figures géométriques simples ».
  - Caractéristique (Structure, 0, unbounded) est une structure décrivant certaines caractéristiques du service. Il est ainsi possible de dire, pour le service précédent, que la « Dimension de dessin » est « 2D ». Chaque caractéristique est décrite par :
    - Nom (Chaîne de caractère, 1, 1) de la propriété à spécifier (« Dimension de dessin » pour notre exemple).
    - Valeur (Chaîne de caractère, 1, 1) assignée à la propriété spécifiée dans Nom (« 2D » pour notre exemple).
    - Commentaire (Chaîne de caractère, 0, 1) permet d'apporter des renseignements supplémentaires sur la propriété en question.
- Acteur (Structure, 1, unbounded) permet d'énumérer l'ensemble des acteurs humains concernés par les services du composant. Les informations mentionnées dans cette structure sont identiques aux informations de Service :
  - Nom (Chaîne de caractère, 1, 1) de l'acteur concerné par au moins un service du composant. Il est souhaitable d'associer à cette métadonnée une liste hiérarchisée comportant les termes : apprenant, co-apprenant, enseignant, tuteur, formateur, etc.
  - Caractéristique (Structure, 0, unbounded) : Ce champ est du même type que celui pour Service. Il permet de décrire un acteur en lui associant des caractéristiques. Il est ainsi possible par exemple

de dire que l'âge minimal d'un apprenant est de 8 ans.

- Relation (Structure, 0, unbounded) permet d'associer services et acteurs (comme les associations n-n dans les bases de données). Elle comporte les métadonnées suivantes :
  - Nom (Chaîne de caractère, 1, 1) permet d'identifier une relation entre un service et les acteurs concernés.
  - Service (Chaîne de caractère, 1, 1) précise le nom du service concerné par la relation. Le nom doit être mentionné dans une instance de la métadonnée Service.
  - Acteur (Chaîne de caractère, 1, unbounded) précise le nom des acteurs concernés par le service. Les noms doivent être mentionnés dans les instances de la métadonnée Acteur.
  - Caractéristique (Structure, 0, unbounded) : Ce champ est de même nature que celui défini pour Service. Il permet d'attribuer des caractéristiques à la relation service acteurs.

Durée

Propriétés : (Structure, 0, 1)

Cette rubrique permet d'énumérer les actions du composant dont il est important de préciser une estimation de leur durée d'exécution ou d'exploitation. Elle comporte la seule métadonnée suivante :

- Action (Structure, 1, unbounded) permet de renseigner la durée minimale et recommandée d'une action. Cette action est décrite par les métadonnées suivantes :
  - Nom (Chaîne de caractère, 1, 1) identifie l'action ou le service dont la durée a un impact sur la formation.
  - Duree_Minimale (Temps, 1, 1) donne la durée d'utilisation minimale nécessaire pour atteindre l'objectif visé du service.
  - Duree_Recommandee (Temps, 1, 1) donne la durée d'utilisation recommandée.
  - Unite_Mesure (Chaîne de caractère, 1, 1) : c'est l'unité de temps utilisée pour mesurer la durée minimale et la durée recommandée. Une liste plate avec les valeurs : seconde, minute, heure, jour, etc. doit être associée à cette métadonnée.
  - Commentaire : (Chaîne de caractères, 0, 1) est une métadonnée facultative permettant d'apporter des précisions sur l'action ou sur ses durées.

Aspects techniques

La LSCM est décrite par un schéma au sens XML du terme. Ce schéma XSD permettra de créer des documents XML valides décrivant les CLP. L'idée est de permettre aux concepteurs de composants de les décrire, les indexer et de les entreposer avec leurs descriptions sur une plate-forme de mutualisation que nous avons développée avec l'aide d'un stagiaire ingénieur. Une recherche sur ces métadonnées pour retrouver les composants sera possible grâce à un outil de recherche réutilisé et paramétré dans notre laboratoire. Cette plate-forme est basée sur les produits développés par la communauté cocoon d'Apache.

Conclusion

Dans cet article, nous avons rappelé le concept de composant et nous en avons distingué quatre catégories : Les CLP, CLS, CLT et CLF. Nous avons, ensuite, présenté un schéma de description des composants logiciels nommé LSCM. Comme la LOM pour les ressources pédagogiques, ce schéma comporte un ensemble de métadonnées hiérarchisées. Il servira pour indexer les composants logiciels, les prototypes ou les applications. Il est important de noter que, si nous avons essayé d'être exhaustifs dans la description de ces CLP, il y a seulement 7 champs obligatoires, a priori assez faciles à remplir. Il s'agit de : catégorie, EspaceTemps, relationInterApprenants, Modèle, Stratégie, DomaineEnseignement, et ButDidactique. La minimisation du nombre de métadonnées obligatoires est important car l'effort demandé pour les remplir doit absolument être limité. Nos futurs objectifs consistent à tester auprès des utilisateurs la validité de ce modèle en essayant d'indexer quelques produits sur la plate-forme de mutualisation qui sera disponible prochainement.

Remerciements

Nous tensons à remercier tous les membres de l'équipe AIDA ayant contribué à l'amélioration et la construction du schéma de description. Plus particulièrement, nous remercions Brigitte de La Passardière et Michèle Artigue pour le temps qu'elles ont consacré à l'étude détaillée des métadonnées.

Références

[1] BARBIER F. et al, (2002). « *Composant dans l'ingénierie des systèmes d'informations : concepts clés et techniques de réutilisation* ». Actes des 2e Assises nationales du GDR I^3 décembre 2002. Nancy : Cépaduès-Éditions. p. 95-117.

[2] SZYPERSKI C. (2002), « *Component software beyond object-oriented programming* ». Addison Wesley, États-Unis.

[3] EZRAN M., MORSIO M., (1999). « *Réutilisation logicielle* ». Paris, Eyrolles.

[4] SOMMERVILLE I., (1992). « Le génie logiciel ». Addison Wesley, France.

[5] ALLEN, P. & FROST, S. (Mars 1998) « *Tackling the Key Issues in CBD* ». Select Software Tools.

[6] A.W. BROWN and K.C. WALLNAU, « *The Current State of CBSE*", IEEE Software September/October 1998

[7] BOURDA Y., (2001). « Objets pédagogiques, vous avez dit objets pédagogiques ? ». *Cahier GUTenberg n° 39-40*, p. 71-79.

[8] GRANDBASTIEN. M. « *Quelques questions à propos de l'indexation et de la recherche de ressources pédagogiques sur le Web* ». In Baron G.-L. et Bruillard E. Eds. Les technologies en éducation : Perspectives de recherche et questions vives. Paris : INRP - MSH - IUFM de Basse Normandie. p. 211-220.

[9] PERNIN, J.P. « *Objets pédagogiques : unités d'apprentissage, activités ou ressources ?* ». Revue "Sciences et Techniques Educatives", Hors série 2003.

[10] «*Wisconsin Online Ressource Center*». http://www.wisc-online.com/ (consulté en 2003).

[11] http://dublincore.org/

[12] http://www.adlnet.org/

[13] IRIBARNE, L., VALLECILO, A., ALVES, C., CASTRO, C., (November 2001) « *A Non-functional Approach for COTS Components Trading* ». In IV Workshop on Requirements Engineering, Buenos Aires, Argentina.

[14] http://standards.ieee.org/reading/ieee/std_public/description/se/

[15] http://www.w3.org/TR/NOTE-OSD

[16] TCHOUNIKINE P., (2002), « *Conception des environnements informatiques d'apprentissage : mieux articuler informatique et sciences humaines et sociales* ». In : Les technologies en éducation : Perspectives de recherche et questions vives, Baron G.L., Bruillard E. (ed.), p. 203-210, Edité par Paris : INRP - MSH - IUFM de Basse Normandie.

[17] HOUSSAYE J, (1988). « Le triangle pédagogique ». Berne, Peter Lang.