Approche par transformation de modèles pour la conception d'EIAH

Pierre Laforcade

Approche par transformation de modèles pour la conception d'EIAH

Illustration entre les langages CPM et IMS-LD

Pierre Laforcade

LIUPPA
Université de Pau et des Pays de l'Adour
B.P. 1155
64013 Pau Cedex
Pierre.Laforcade@univ-pau.fr

RÉSUMÉ. *Ce papier concerne la conception de situations d'apprentissage dans le cadre de la formation à distance. Des travaux préalables ont été menés afin de proposer un langage, CPM (Cooperative Problem-based learning Metamodel), exploitant la richesse graphique proposé par UML (Unified Modeling Language). CPM est alors construit sous la forme d'un profil UML.Il se positionne au niveau des phases de conception en amont des EML (Educational Modeling Language). Nous présentons alors une technique de transformation de modèles, entre les langages CPM et IMS-LD, pour la spécification de scénarios pédagogiques. Cette technique est basée sur l'idée qu'un modèle UML outillé peut être interprété comme un système d'information ; la transformation est alors dirigée selon les éléments constituant le profil. Un exemple illustre la démarche.*

MOTS-CLÉS : *génie logiciel, ingénierie, conception des EIAH, formation ouverte ou à distance, scénario pédagogique, système auteur, normes et standards, UML*

1. Introduction

Ce papier décrit une étude centrée Génie Logiciel pour le domaine pluridisciplinaire de l'ingénierie des EIAH. Notre intérêt se porte sur le champ de la formation à distance. Dans un tel contexte industrialisé, nos sujets de recherche se positionnent au niveau de la phase de conception des formations. Nous nous intéressons alors pour ce contexte précis aux modèles de conception de type design pédagogique [PAQUETTE 04] dont les objectifs sont i) de faciliter la spécification *prescriptive* des formations (quelles sont les ressources ? les critères de succès à spécifier ? les activités à proposer ? les rôles à jouer ? etc.) et ii) d'agir comme supports de réflexion et de communication pour l'équipe multidisciplinaire chargée de la conception. Le scénario pédagogique se définit comme *le résultat manipulable de la modélisation d'une situation d'apprentissage* [DAELE & al. 03] ou plus précisément comme *la description du déroulement d'une situation d'apprentissage en termes de rôles, d'activités et d'environnement nécessaire à sa mise en œuvre, mais aussi en termes de connaissances manipulées* [PERNIN 03].

Depuis quelques années, les travaux de recherche ont montré que les plates-formes de formation à distance ne peuvent se contenter de seulement mettre à disposition des ressources d'apprentissage. Des langages de modélisation pédagogiques (EML) sont apparus dans le but de gérer les ressources mais surtout les activités pédagogiques les utilisant. Le travail autour des EML, initié avec EML-OUNL [KOPER 02] puis standardisé dans la spécification d'IMS-LD, permet de spécifier formellement des unités d'apprentissage correspondants à la description des ressources et du scénario les manipulant (séparation contenu/scénario sous la forme d'une méthode spécifiant les rôles, les activités réalisées pour chaque rôle et les ressources manipulées au travers de la réalisation). Ces EML concernent des ingénieurs pédagogiques experts de ces langages. Bien qu'IMS-LD ait été finalisée récemment, quelques initiatives ont abouti à des prototypes d'outils-auteurs compatibles avec IMS-LD : [EDUBOX 04], [RELOAD 04], le système LAMS [DALZIEN 03] et l'*Open Source LD engine* [VOGTEN & MARTENS 03]. Toutefois, aucun environnement auteur n'adresse l'équipe pluridisciplinaire chargée de la conception des unités d'apprentissage, c'est-à-dire les utilisateurs finaux non-experts. L'élaboration de modèles avec des langages comme IMS-LD concerne la phase de conception détaillée pour laquelle un pré-scénario a déjà été établi par les enseignants et autres concepteurs de l'équipe en charge de concevoir la formation. Des modèles basés sur le langage UML, et plus particulièrement les diagrammes d'activités, servent à illustrer les résultats d'analyse, les *narrations*, sur lesquels se basent les EML. La spécification d'IMS-LD encourage l'utilisation d'UML pour ces phases plus amont.

Nous avons étudié les apports potentiels du formalisme UML ainsi que ceux de méta-modélisation UML afin de pouvoir le spécialiser [LAFORCADE 04]. Ces travaux de recherche ont abouti à la proposition du langage CPM. Ce langage a été outillé sur la base d'un Atelier de Génie Logiciel existant [OBJECTEERING 04]. Nous avons également initié un travail de transformation de modèle afin d'expérimenter et de garantir de nouveaux usages qui permettront de ne pas restreindre les modèles élaborés avec le langage CPM à un usage *documentaire*.

L'objectif de ce papier consiste à présenter la démarche technique que nous avons expérimentée afin de montrer la valeur ajoutée de disposer de modèles CPM (c'est-à-dire élaborés avec le langage CPM) outillés. Nous avons choisi d'exprimer cette valeur ajoutée sur un usage particulier de l'exploitation des modèles UML outillés : la transformation de modèles. Dans notre contexte, il s'agit de transformer des modèles CPM en modèles XML conformes au standard IMS-LD. De manière similaire au rapprochement entre la méthode d'ingénierie pédagogique MISA[1] (et sa notation MOT) et les EML [PAQUETTE 04], notre objectif consiste également à présenter les contributions et adaptations réciproques entre CPM et IMS-LD.

Dans la prochaine section nous établissons une comparaison de ces deux langages. Ensuite, nous présentons la technique de transformation basée sur l'exploitation des informations contenues dans les modèles CPM grâce au mécanisme des profils et à l'outil. Cette technique est ensuite illustrée sur l'un de nos exemples d'expérimentation. Finalement, nous discutons de la valeur ajoutée de ce type de transformation, puis concluons par la présentation de perspectives exploitant cette technique.

2. Comparaison des langages de modélisation CPM et IMS-LD

Le langage IMS-LD a été présenté brièvement dans la section précédente. Nous ne détaillons pas davantage ce langage formel standard présenté et étudié déjà dans de nombreux articles. Nous préférons inviter le lecteur à consulter les spécifications officielles [IMSa 03, IMSb 03]. En revanche, nous présentons dans cette section le langage semi-formel CPM, basé sur une spécialisation d'UML *via* la notion de profil UML. Enfin, nous terminons cette section en résumant les différences entre les langages CPM et IMS-LD.

2.1. Le la` gage CPM

Nous avons proposé un langage facilitant l'élaboration de modèles pour la conception de situations-problèmes (PBL) sur des plates-formes de formation à distance. L'étude des différents langages correspondant en partie à cet objectif a permis de mettre en évidence i) le manque actuel de modèle et de langage pour l'étape d'expression initiale des besoins, ainsi que ii) la nécessité d'utiliser UML pour décrire des modèles abstraits (étape d'analyse) servant de base à la spécification formelle réalisée par les EML comme IMS-LD (étape de conception détaillée). Ce constat nous a alors amené à positionner notre proposition comme un langage de modélisation graphique spécialisant UML pour la conception de PBL en amont des langages de type EML : il couvre alors la phase de conception pour les étapes d'expression initiale des besoins, d'analyse et de conception.

---

[1] Il s'agit d'une méthodologie de design pédagogique basée sur une approche ingénierie des connaissances.

Le langage CPM proposé permet alors de décrire les différents modèles de la PBL pour les phases amont de conception sur deux niveaux :

- horizontalement : les modèles sont différents (en termes d'objectifs, d'abstraction, de contenus et d'usages) selon les étapes dans la phase de conception (expressions initiales des besoins, analyse, conception) ;
- verticalement : les modèles capturent différents points de vue ou perspectives sur la même situation d'apprentissage (pédagogique, didactique, social ou encore médiatique).

Ces modèles aident l'équipe pluridisciplinaire de conception à décrire, spécifier et documenter la PBL qu'ils conçoivent. De plus, le caractère visuel des modèles produits permet d'abstraire la complexité de la PBL mais constitue également une représentation adéquate pour favoriser le dialogue, la compréhension et l'implication des différents intervenants.

Le langage CPM propose une syntaxe (terminologie et notation) et une sémantique adaptées afin de satisfaire l'ensemble des différents modèles possibles pour la phase de conception. La syntaxe abstraite est capturée dans le méta-modèle CPM [NODENOT et al. 03], tandis que la syntaxe concrète est décrite au travers du profil CPM [LAFORCADE et al. 03]. La sémantique des différents concepts et relations proposés est donnée *via* des contraintes OCL (*Object Constraint Language)* et des explications en langage naturel.

Un prototypage d'environnement-auteur pour le langage CPM a été proposé sur la base d'une implantation du profil CPM dans l'outil *Objecteering Profile Builder* et d'une personnalisation/adaptation de l'interface de l'AGL pour faciliter la création et le suivi de modèles CPM. Ce type d'environnement concerne plus particulièrement l'ingénieur pédagogique pour lequel un niveau de connaissance suffisant d'UML est nécessaire. Concrètement, le langage CPM est actuellement disponible sous la forme d'un module intégrable à l'AGL UML *Objecteering Modeler*.

2.2. Tableau comparatif

Le tableau suivant résume différents éléments permettant de comparer de manière très synthétique les deux langages IMS-LD et CPM. Ces éléments d'informations sont tirés des deux sections précédentes. Les éléments de comparaison sont regroupés en trois catégories : Langage, Modèles/instances et Outils.

	CPM	**IMS-LD**
Langage		
Type	Semi-formel (graphique)	Formel
Définition de la Terminologie	Méta-modèle CPM	Modèle d'information IMS-LD
Définition de la notation	Profil UML CPM	
Public cible	Ingénieur pédagogique spécialiste d'UML	Ingénieur pédagogique formé à IMS-LD

Modèles/instances		
Phases concernées	Expression initiale des besoins, analyse et conception	Conception détaillée
Type	Modèles UML	Modèles XML
Public cible	Equipe pluridisciplinaire chargée de la conception	Machine
Outillage		
	Oui (Objecteering)	Voir l'introduction (section 1)

Tableau 1. *Comparaison entre les langages CPM et IMS-LD*

3. Technique d'exploitation des modèles CPM

Notre objectif technique consiste ici à transformer des modèles CPM vers des modèles conformes à la spécification IMS-LD (l'inverse serait également envisageable dans une approche rétro-ingénierie visant à la ré-utilisation de scénarios pédagogiques). Etant donné que i) CPM est une spécialisation d'UML et que les modèles élaborés avec CPM sont des modèles UML (ensemble de diagrammes) manipulant des stéréotypes et des valeurs marquées, et que ii) les modèles élaborés selon la spécification IMS-LD sont des modèles XML, alors cette transformation se réduit conceptuellement à une transformation de modèles UML vers des modèles XML conformes à une certaine DTD.

Une première technique intuitive consiste alors à exporter les modèles CPM en modèles XML via les règles décrites dans la spécification 1.1 du *XML Metadata Interchange* (XMI) [OMGb 03]. Ensuite, il s'agit de transformer un modèle XML vers un autre, en établissant les relations entre les deux DTD correspondantes.

Nous avons choisi, pour notre part, d'expérimenter une seconde technique permise par l'outillage actuel de notre langage CPM. C'est cette technique que nous présentons succinctement dans cette section.

3.1. Modèle UML comme système d'information

Il existe de nombreux outils supportant UML, allant de simples outils de dessin proposant la notation UML jusqu'aux ateliers de génie logiciel (AGL) pour UML (une liste exhaustive des AGL UML les plus aboutis et les plus utilisés est disponible sur [UMLTOOLS 05]). Ces AGL fournissent de nombreuses fonctionnalités pour le support du développement et d'exploitation de modèles UML (voir dans [LAFORCADE 04] section 5.1.5).

La fonctionnalité qui nous intéresse est celle d'agir comme un entrepôt : les informations contenues dans les modèles doivent être conservées et synchronisées (par exemple, si le nom d'une classe est modifié alors le changement doit être répercuté dans tous les autres diagrammes dans lesquels la classe est utilisée). Le modèle UML élaboré avec l'outil est alors perçu comme une source d'information interprétable par la machine : le modèle UML est un système d'information.

Certaines fonctionnalités sont facilitées grâce à cet entrepôt : vérifier la cohérence des modèles, critiquer les modèles (l'outil peut indiquer les erreurs comme les oublis ou pointer les solutions inappropriées en appliquant des heuristiques sur les modèles), générer des documentations, permettre la réutilisation d'éléments ou de diagrammes (de manière à ce que tout ou partie de solutions dans un projet puisse être facilement réutilisé dans d'autres).

L'AGL *Objecteering*, que nous avons utilisé pour outiller notre langage CPM, propose lui aussi ces fonctionnalités codées en interne grâce à leur langage propriétaire. De plus, ce langage est également à disposition pour l'utilisateur final.

3.2. Langage J et démarche technique de transformation

Le langage J intégré est un langage objet, dynamique et interprété, dédié à la navigation et à la construction de requêtes sur le méta-modèle UML. Ce langage nous permet de naviguer dans les diagrammes UML comme s'il s'agissait d'un système d'information. Toutefois, la navigation est régie par des règles correspondant au parcours du méta-modèle d'UML tel qu'il est implanté dans l'outil.

Alors que la définition théorique du langage CPM est composée du méta-modèle CPM et du profil CPM, l'outil manipule le langage CPM sous sa forme pratique : comme un profil UML particulier implanté. Ce profil « outillé » se décompose en divers éléments dont les principaux sont les stéréotypes et les définitions de valeurs marquées. Les éléments de terminologie du langage CPM sont alors traités dans les modèles comme des instances de stéréotypes et de valeurs marquées. Ainsi, conformément à la spécification d'UML (voir les mécanismes d'extension section 2.6 dans [OMGa 03]), il est possible avec le langage J d'interroger tout élément de modélisation pour connaître le stéréotype et les valeurs marquées qui lui sont appliqués.

La démarche technique suivie, pour extraire les informations nécessaires au modèle cible conforme à IMS-LD, consiste à déterminer pour chaque concept d'IMS-LD quels sont :

- la méta-classe UML et le stéréotype utilisés comme équivalents dans la spécification du même concept avec le langage CPM ;
- optionnellement, quelle définition de valeur marquée apporte une information supplémentaire significative pour l'identification du concept.

Le modèle cible conforme à IMS-LD est alors construit dans l'ordre, chaque composante étant déduite d'informations extraites du modèle CPM. Toutefois, certains éléments sont plus difficilement déductibles comme, par exemple, le découpage d'un *play* en *acts* ; nous abordons ce problème dans la section d'expérimentation.

4. Expérimentation : CPM vers IMS-LD

La technique présentée précédemment a été mise à l'essai sur différents exemples (la plupart tirés de [IMSa 03]) dont nous présentons maintenant la mise en œuvre expérimentale. Mais tout d'abord nous présentons les choix préliminaires pour la mise en œuvre de l'expérimentation.

4.1. Contexte d'expérimentation

Nous avons choisi pour la mise en place de notre cadre d'expérimentation de restreindre, dans un premier temps, les modèles CPM aux seuls diagrammes d'activités comme source d'information sur nos modèles (puisqu'ils se prêtent conceptuellement davantage à décrire graphiquement les scénarios des modèles IMS-LD). Du point de vue du langage-cible IMS-LD, nous nous sommes limités à la génération du niveau A ; toutefois, les niveaux B et C sont envisageables étant donné les correspondances syntaxiques entre CPM et IMS-LD (voir [LAFORCADE 04] section 7.4.2).

4.2. Application de la technique précédente

Nous donnons dans la Tableau 2 ci-après les correspondances précises entre les concepts IMS-LD apparaissant dans les diagrammes d'activités UML utilisés dans le document [IMS 03] et les concepts CPM.

La première colonne fait référence aux seuls éléments d'IMS-LD apparaissant conceptuellement dans les exemples de diagrammes d'activités proposés dans [IMSa 03]. Nous donnons ensuite le concept CPM correspondant sous la forme d'un stéréotype, de la méta-classe de référence UML qu'il étend ainsi que la définition de valeur marquée utilisée si elle existe. Lorsque cette dernière prend un paramètre dont la valeur permet de mieux identifier le concept IMS-LD correspondant alors nous précisons également cette valeur attendue.

Concept IMS-LD	**Concept CPM**			
	Méta-classe UML	**Stéréotype**	**Définition de valeur marquée**	**Valeur du paramètre**
learner	Partition	Role	roleKind	learner
staff	Partition	Role	roleKind	staff
support-activity	ActionState	Activity ou CollaborativeActivity		
learning-activity	ActionState	Activity ou CollaborativeActivity		
activity-structure	SubactivityState	ActivityStructure	structureKind	

Tableau 2. *Correspondances IMS-LD ↔ CPM*

Les deux concepts d'activité d'IMS-LD (*support-activity* et *learning-activity*) sont déduits selon le type du rôle attribué à la partition dans laquelle se trouve l'activité concernée exprimée avec CPM : une activité positionnée dans la partition des activités d'un rôle d'apprenant est une *learning-activity*.

Les concepts de *method* et *play* ne sont pas traités car un diagramme d'activités ne permet de spécifier que les actes pour un *play* donné. Le découpage en actes nécessite pour sa part la mise en œuvre d'un algorithme approprié. Celui-ci est construit sur la base de remarques déduites de l'observation et de l'analyse des différences entre les diagrammes d'activités et des modèles XML correspondants proposés dans [IMSa 03] :

- l'utilisation des barres de synchronisation (*fork* et *join*) permet de modéliser le parallélisme des activités implicitement mentionné dans la narration.
- les flèches traversant les couloirs expriment un besoin de précédence entre les deux activités qu'elles relient ;
- les cadres entourant des activités successives réalisées par le même rôle font référence aux *activity-structure* spécifiées dans le document XML correspondant.

4.3. Illustration

Nous présentons comme illustration de l'application de la transformation de modèles, l'exemple tiré de [IMSa 03] section 4.5, intitulé « *Problem Based Learning* ». Par souci de place, nous ne reportons pas la narration accompagnant le diagramme d'activité UML proposé ni le modèle XML complet produit conformément à la spécification IMS-LD (appelé le *XML Instance Document*). De même, nous reportons uniquement en Figure 1 un extrait de ce diagramme (Nous avons caché les éléments de modélisation se trouvant avant la barre de synchronisation de type *fork* comme ceux situés après la barre de type *join*).

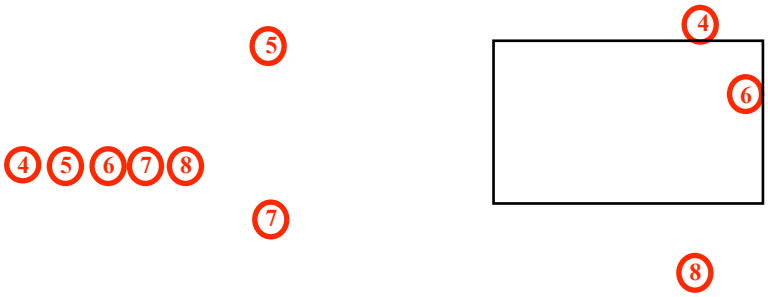

Figure 1. *Extrait du diagramme d'activité proposé dans [IMSa 03] section 4.5.2*

L'extrait choisi présente toutefois les deux caractéristiques (de la section 4.2 ci-dessus) exprimant la difficulté d'interprétation *automatisé* du scénario :

- *Provide assistance* est une activité qui doit être réalisée en parallèle de toutes les autres activités situées entre les deux barres de synchronisation ;
- le découpage en acte sur cet extrait est indiqué par les numéros ajoutés sur la figure 1 ;
- *Brainstorm explanations* et *Cluster explanations* seront regroupés dans un même *activity-structure* de type *sequence*.

La Figure 2 est un extrait du document XML IMS-LD spécifiant l'acte 4 comme le regroupement de deux *role-part* dans lesquels on retrouve effectivement les activités *Provide assistance* et *Clarify Problem* (annotés du chiffre « 4 » dans la figure 1).

```
- <imsld:act>
  - <imsld:role-part>
      <imsld:role-ref ref="R-student"/>
      <imsld:learning-activity-ref ref="LA-Clarify-Problem"/>
    </imsld:role-part>
  - <imsld:role-part>
      <imsld:role-ref ref="R-facilitator"/>
      <imsld:learning-activity-ref ref="SA-Provide-Assistance"/>
    </imsld:role-part>
  - <imsld:complete-act>
      <imsld:when-role-part-completed ref="R-student"/>
    </imsld:complete-act>
</imsld:act>
```

Figure 2. Extrait du *XML Instance Document* correspondant

Nous avons ensuite modélisé ce même exemple de situation d'apprentissage sous la forme d'un autre diagramme d'activité avec le langage CPM. Nous avons extrait la même sous-partie que précédemment (Figure 3).

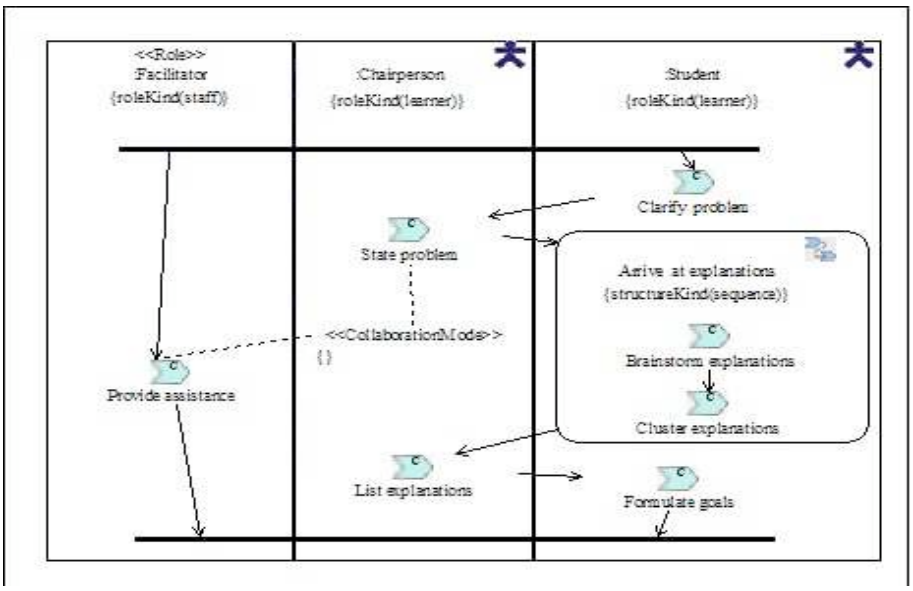

Figure 3. *Extrait du diagramme d'activité spécifié à l'aide du langage CPM*

La plupart des stéréotypes sont visibles sous leur forme d'*icône* : *Role*, *CollaborativeActivity*, et *ActivityStructure* ; seul le premier rôle de *Facilitator* montre le stéréotype sous sa forme labelisée. Les valeurs marquées sont affichées sous la forme **{définition-valeur-marquée(paramètre-valeur-marquée)}**. Une seule contrainte stéréotypée *CollaborationMode* est montrée dans la figure sur les cinq : les autres sont pourtant spécifiées entre *Provide assistance* et les autres activités mais ne sont pas affichées afin de ne pas alourdir la lisibilité du diagramme. C'est grâce à ce concept CPM que nous allons pouvoir spécifier les liens entre les activités collaboratives. La fonctionnalité de transformation CPM vers IMS-LD que nous avons développé utilise cette indication pour déduire les différents *role-part* composant les actes.

La génération du fichier XML conforme au standard IMS-LD est appelé *via* une commande accessible au niveau du modèle CPM dans le menu contextuel dédié à CPM.

5. Conclusions, discussion et travaux futurs

Dans ce papier, nous nous sommes intéressés aux modèles de conception de situations d'apprentissage pour la formation à distance. Nous avons ainsi présenté l'application d'une technique exploitant la possibilité d'interpréter par la machine des modèles UML conçus au sein d'un Atelier de Génie Logiciel UML. Cette technique permet de transformer un modèle UML vers un autre type de modèle grâce aux stéréotypes et définitions de valeurs marquées définis dans les profils. Ces éléments informent et dirigent la collecte d'informations contenues dans le modèle

UML. Nous avons appliqué cette technique dans le cadre de la spécification de scénarios d'apprentissage afin de transformer des diagrammes d'activités modélisant des scénarios à l'aide du langage CPM (défini sous la forme d'un profil UML) vers des modèles XML conformes au standard de la spécification actuelle d'IMS-LD.

L'utilisation d'autres diagrammes, comme par exemple des diagrammes de classes décrivant les objectifs et les pré-requis de la situation d'apprentissage, pourront être également utilisés par la suite. Ainsi, d'autres éléments terminologiques de la spécification IMS-LD pourront être interprétés et manipulés dans des modèles d'analyse/conception amont (nous travaillons actuellement sur les niveaux B et C d'IMS-LD). Ce travail actuel d'expérimentation, indépendant de l'ingénieur pédagogique qui est le véritable public-cible du langage CPM, laisse envisager la possibilité d'élaborer un profil UML spécifique à IMS-LD mais également la possibilité d'outiller ce profil afin qu'il facilite l'élaboration de modèles XML conformes au standard IMS-LD : un nouvel environnement-auteur basé UML pour IMS-LD est envisageable. En revanche, ces travaux d'expérimentation montrent également que de nombreuses informations capturées dans les modèles CPM n'ont pas de correspondance avec IMS-LD ; cette perte d'information sémantique du langage CPM est due principalement à son positionnement plus amont dans le processus de conception et à son centrage sur un type particulier de situation d'apprentissage : les PBL.

Nous envisageons d'autres applications de transformation de modèles pour le langage CPM. Sur la base conceptuelle qu'un modèle UML outillé est interprétable par la machine et que les profils permettent de guider les transformations alors nous envisageons d'automatiser la transformation des modèles CPM d'expression initiale de besoins vers des modèles CPM d'analyse, puis, à leur tour, vers des modèles CPM de conception. Nous aurons ainsi des modèles qui guident et dirigent la conception, c'est-à-dire un contexte d'ingénierie des modèles propice à la mise en place d'une méthode d'ingénierie pédagogique adaptée à notre langage. Une perspective supplémentaire vise également à étendre les usages des modèles CPM au niveau de la phase d'exécution de l'apprentissage : ces modèles seraient alors une source d'informations pour la régulation (suivi des activités, évaluation ou encore aide au tutorat) de l'apprentissage *effectif*. Un système adapté manipulant de tels modèles interprétables pourrait alors capter des actions des différents acteurs afin de modifier le modèle-machine en conséquence (rétro-action avec mise à jour du modèle et adaptation possible de l'environnement de formation).

6. Bibliographie

[DAELE & al. 03] Daele A., Brassard C., Esnault L., O'Donoghue M., Uyttebrouck E., Zeileger R., « Wp2. conception, mise en oeuvre, analyse et évaluation de scénarios pédagogiques recourant à l'usage des technologies de l'information et de la communication ». Rapport recre@sup, 2003. http//tecfa.unige.ch/proj/recreasup/rapport/WP2.pdf

[DALZIEL 03] Dalziel J., Discussion Paper for Learning Activities and Meta-data, Technical report, 2003, Heerlen: Open University of the Netherlands.

[IMS 03a] IMS, IMS Learning Design Best Practice and Implementation Guide, Technical report, janvier 2003.

[IMS03b] IMS, IMS Learning Design Version 1.0 Final Specification, Technical report, février 2003.

[KOPER 02] Koper R. Educational modelling language: adding instructional design to existing specifications, Technical report, Educational Expertise Technology Centre, Open University of the Netherlands, 2002.

[LAFORCADE & al. 03] Laforcade P., Barbier F., Nodenot T., Sallaberry C., «Profiling Co-operative Problem-Based Learning Situations», *Proceedings of the 2nd IEEE International Conference on Cognitive Informatics (ICCI'2003)*, London, UK, 18-20 août 2003, IEEE Computer Society Press.

[LAFORCADE 04] Laforcade P., Modélisation et méta-modélisation UML pour la conception et la mise en œuvre de situations-problèmes coopératives, Thèse de doctorat, Université de Pau et des Pays de l'Adour, 2004, 350p.

[NODENOT & al. 03] Nodenot T., Laforcade P., Marquesuzaà C., Sallaberry C., «Knowledge Modelling of Co-operative Learning Situations: Towards a UML profile», *Proceedings of the 11th International Conference on Artificial Intelligence in Education (AIED'2003)*, Sydney, Australia, 20-24 juillet 2003, International AI-ED Society.

[OMGa 03] OMG, Unified Modeling Language v1.5, Report formal/03-03-01, March 2003.

[OMGb 03] OMG, XML Metadata Interchange, v2.0. Report formal/03-05-02, May 2003.

[PAQUETTE 04] Paquette G., Educational Modeling Language: from an Instructional Engineering Perspective, in an article to be published in a forthcoming handbook, 2004, http://www.licef.teluq.uquebec.ca/gp/fr/publications/documents/ArticleEML-MISA.doc

[PERNIN 03] Pernin J.-P., Critères pour une typologie des langages de modélisation pédagogique, présentation au GDRi3 EIAH, Paris, novembre 2003.

[VOGTEN et MARTENS 03] Vogten H., Martens H., Open Source Learning Design Engine, Technical report, 2003, http://mdlet.jtc1sc36.org/doc/SC36_WG4_N0043.pdf

7. Références sur le WEB

[EDUBOX 05] http://www.ou.nl/info-alg-edubox/

[OBJECTEERING 04] http://www.objecteering.com/products.php

[RELOAD 05] http://www.jisc.ac.uk/index.cfm?name=project_reload

[UMLTOOLS 05] http://www.objectsbydesign.com/tools/umltools_byCompany.html