Sciences et Technologies de l´Information et de la Communication pour l´Éducation et la Formation

Volume 14, 2007
Article de recherche

Une approche conduite par les modèles pour le traçage des activités des utilisateurs dans des EIAH hétérogènes

Julien BROISIN, Philippe VIDAL [IRIT, Toulouse]

■ RÉSUMÉ : Cet article propose une approche conduite par les modèles pour la gestion des traces d'activité des utilisateurs au sein de systèmes d'apprentissage hétérogènes instrumentés par les technologies Web. Un modèle UML générique de traces permet de structurer et d'ajouter une sémantique claire aux données observées, auquel est associée une architecture distribuée et décentralisée favorisant le partage et la réutilisation des traces collectées par différents outils et services. Nous appliquons ensuite cette approche au traçage de l'utilisation des objets pédagogiques par des utilisateurs exploitant des plates-formes d'apprentissage et viviers de connaissance. Enfin nous proposons une application pour la visualisation des traces collectées, ainsi qu'un service de recherche avancée d'objets pédagogiques qui offre l'opportunité de capitaliser les expériences d'une large communauté d'utilisateurs.

■ MOTS CLÉS : Traces d'activité, partage et réutilisation, modélisation, architecture distribuée, objet pédagogique.

■ ABSTRACT : This paper suggests a model driven approach dedicated to the management of tracking data resulting from users activities within heterogeneous learning systems based on internet technologies. A generic UML model describing traces allows to structure and to bring semantics to observed data, while a distributed and decentralized architecture makes it easier to share and reuse traces between various tools and services. We then apply this approach to track activities of users working with learning objects deployed into learning management systems and learning object repositories. Finally, we suggest an application to visualize tracking data, together with an advanced search service of learning objects that offers the opportunity to capitalize experiences of a wide community of users.

■ KEYWORDS : Usage tracking, share and reuse, modelling, distributed architecture, learning object.

1. Introduction

L'intérêt pour l'observation, l'instrumentation et l'évaluation des systèmes éducatifs en ligne est de plus en plus fort au sein de la communauté des environnements informatiques pour l'apprentissage humain (EIAH). Collecter des données sur les utilisateurs à partir des traces de leurs activités dans l'objectif de faciliter les processus de personnalisation, de réingénierie, ou d'aide constitue un enjeu majeur aujourd'hui, comme le montre le nombre d'appels à soumission ou de projets (ICALTS, 2004) (DPULS, 2005) relatifs à ce sujet. Le partage et la réutilisation des traces représentent alors un facteur important pour atteindre ces objectifs puisque les différents acteurs de l'e-formation utilisent souvent plusieurs outils pour effectuer des tâches spécifiques. Un enseignant-tuteur peut par exemple être amené à exploiter un vivier de connaissance pour trouver des objets pédagogiques existants lors de la création d'un cursus, une plate-forme d'apprentissage pour mettre à disposition des apprenants un scénario pédagogique, et un système de tutorat avancé lors de situations d'apprentissage collaboratives. Nous devons donc être en mesure de répondre à certaines questions tout en prenant en compte des caractéristiques dynamiques et de mise à l'échelle (Steinmetz, 2005) : (a) Quelles sont les informations utiles et nécessaires à la supervision des activités des utilisateurs impliqués dans un EIAH comprenant des outils hétérogènes, et comment les représenter ? (b) Comment collecter ces informations ? et (c) Où stocker ces informations et comment les exploiter ?

Nous proposons dans cet article un modèle UML générique de traces pour les activités ainsi qu'une architecture de gestion distribuée et décentralisée qui permettent d'apporter une réponse aux questions soulevées ci-dessus et ainsi de bénéficier de l'expérience d'utilisateurs de différents systèmes hétérogènes. Ce modèle, organisé autour d'un métamodèle existant issu du monde de la gestion de systèmes et de réseaux, ne s'intéresse qu'à la spécification des activités réalisables sur des systèmes et ressources d'apprentissage, mais démontre également comment il peut être étendu pour satisfaire d'autres objectifs d'observation. Certains composants de l'architecture de gestion nécessaires à l'instrumentation de l'observation permettent de collecter les traces produites par les actions explicites des utilisateurs et de les stocker dans un référentiel, alors que d'autres applications et services sont dédiés à la visualisation et à l'exploitation des traces.

La deuxième partie expose les approches existantes dédiées à la gestion des traces au sein d'un EIAH, et identifie les verrous qui doivent être franchis pour assurer une observation uniforme de systèmes hétérogènes à grande échelle. Nous proposons dans la troisième partie le modèle générique de traces d'activité et l'architecture distribuée associée qui assure la prise en compte des contraintes identifiées dans la deuxième section ; les fondements de notre approche sont également évoqués. La quatrième partie s'intéresse à une implantation de notre approche théorique à travers un cas d'usage qui s'attache à faciliter la réalisation des activités relatives aux objets pédagogiques, en particulier lorsqu'il s'agit de retrouver et de réutiliser des ressources pédagogiques existantes. Nous présentons (1) le modèle UML de traces pour la représentation d'activités spécifiques aux objets pédagogiques et (2) les différents composants inclus dans l'architecture d'observation responsables de l'acquisition, du stockage, de la visualisation et de l'exploitation des traces. Enfin, nous concluons et exposons nos travaux à court et moyen terme dans la dernière partie de ce document.

2. Les systèmes à base de traces dans les EIAH

Différentes approches dédiées à l'observation des activités des utilisateurs au sein d'un EIAH ont vu le jour au cours de ces dernières années. Cette partie situe tout d'abord le contexte de nos travaux. Les approches prédominantes mises en œuvre dans un environnement d'apprentissage en ligne sont ensuite exposées, et permettent de mettre en avant les lacunes et barrières qui doivent être franchies pour faciliter le partage de l'expérience d'une large communauté d'utilisateurs.

2.1. Couverture des travaux

P. Tchounikine définit un EIAH comme "un environnement informatique conçu dans le but de favoriser l'apprentissage humain, c'est-à-dire la construction de connaissances chez un apprenant" (Tchounikine, 2002). Nos travaux s'intéressent en particulier aux systèmes d'apprentissage à distance instrumentés par le *Web* tels que les plates-formes de téléformation (Learning Management System - LMS), viviers de connaissance (Learning Object Repository - LOR) ou systèmes de tutorat en ligne, mais ne prétendent pas couvrir l'ensemble du panel d'outils et de systèmes dédiés à l'apprentissage et à la formation qui sont englobés dans la précédente définition.

Nous nous attachons à collecter les traces produites par les actions explicites des utilisateurs au sein des systèmes d'apprentissage, en y associant une sémantique claire pour leur donner un sens ; ces traces sont donc qualifiées de traces brutes (Courtin et Talbot, 2007).

Leurs usages varient selon les objectifs à atteindre, et peuvent se focaliser sur la surveillance d'une machine ou d'un parc informatique, la collecte d'informations relatives aux utilisateurs, l'évolution et/ou l'amélioration des Interfaces Hommes Machines, etc. Dans le cadre des EIAH, elles sont souvent utilisées à des fins de réingénierie des scénarios d'apprentissage (Kepka et al., 2007), d'aide à la compréhension du comportement de l'utilisateur (Loghin, 2006), ou de support à la création de tutoriels d'aide intelligents (Héraud et al., 2004). Les traces recueillies dans notre contexte visent à faciliter la réalisation d'activités par les différents acteurs de l'e-formation, qu'il s'agisse d'actions effectuées par les administrateurs, les apprenants, les enseignants ou les tuteurs. Cet objectif implique nécessairement de bénéficier de l'expérience d'une large communauté d'utilisateurs, puisqu'un facteur primordial du succès de ce processus est la mise à disposition d'une masse importante de traces à des fins de partage et de réutilisation (Laflaquière et al., 2006).

2.2. Les approches existantes

2.2.1. Les sources de traces : les fichiers de log

L'apprentissage en ligne est souvent basé sur le paradigme Client/Serveur : le système d'apprentissage est hébergé sur un logiciel serveur d'une machine distante, et les enseignants/apprenants utilisent un logiciel client tel qu'un navigateur internet pour effectuer leurs formations. Aujourd'hui, la plupart des serveurs *Web* tels que le logiciel Apache propose un moyen de recueillir des informations concernant les pages, images ou fichiers qui ont fait l'objet d'une requête HTTP, les utilisateurs qui ont effectué ces requêtes, ou encore le nombre d'octets qui ont été transférés.

De nombreuses informations peuvent ainsi être déterminées, telles que le navigateur et le système d'exploitation utilisés, le moment auquel une certaine activité a été effectuée, la durée passée sur une certaine page, le nombre de fois qu'une page spécifique a été visitée, ou encore l'adresse IP correspondant à chacun des événements. Ces

données présentent un certain potentiel pour évaluer l'efficacité d'un cursus en ligne ; elles permettent de quantifier les interactions entre les utilisateurs et les pages du cours/cursus et ainsi de répondre à certaines questions comme : les pages *Web* du cours sont-elles adaptées au navigateur le plus utilisé par les apprenants ? Est-ce que les apprenants accèdent facilement aux pages essentielles du cours ? Une page spécifique est-elle réellement nécessaire à ce cours ? Le laps de temps passé par les étudiants sur une page particulière est-il adapté pour atteindre l'objectif visé ? etc.

Les travaux présentés dans (Iksal et Choquet, 2005a) (Iksal et Choquet, 2005b) s'appuient sur les fichiers de *log* pour l'acquisition des traces, et suggèrent le métalangage *Usage Tracking Language* (UTL) pour analyser le comportement des utilisateurs dans l'objectif d'adapter et d'améliorer les scénarios et ressources d'apprentissage. Cette analyse s'appuie sur une approche orientée modèles décrivant les scénarios pédagogiques, et permet d'aiguiller les apprenants vers les scénarios les mieux adaptés à leur situation.

Dans le même ordre d'idée, (Stermsek et al., 2007) cherche à exploiter les fichiers de *log* des apprenants et les métadonnées des pages HTML pour déduire leur profil et ainsi proposer des mécanismes de filtrage d'informations. L'extraction des données pertinentes au sein du fichier de *log* et leur analyse donnent des indicateurs sur le comportement de l'utilisateur ; celui-ci est interprété (à partir de données statistiques, temporelles, etc.) pour dériver le profil de l'utilisateur.

Cependant, les approches existantes de gestion de traces qui sont fondées sur les fichiers de *log* doivent faire face à certaines barrières relatives à l'accès et à la signification des informations :

- Elles sont renfermées dans des fichiers qui ne sont généralement pas disponibles à n'importe quel utilisateur ; les administrateurs Réseau et Système sont de plus en plus méfiants et n'autorisent que très rarement l'accès à ce type d'information.

- Elles ne transcrivent pas nécessairement l'exploitation réelle du système par les utilisateurs : un grand nombre de moteurs de recherche tels que Google™ ou Altavista™ sont basés sur des robots qui analysent le contenu d'un logiciel serveur afin d'en extraire les informations pertinentes. Même si certains systèmes différencient les visites effectuées par les robots de celles effectuées par les utilisateurs, la majorité d'entre eux accumulent le nombre de visites, ce qui ne rend pas compte de l'usage véritable du système d'apprentissage.

- L'acquisition de ces données doit être effectuée selon un intervalle de temps déterminé par le concepteur de l'outil de supervision, elle n'est pas réalisée en temps réel.

- Les informations renfermées dans ces fichiers ne sont pas toujours très précises. Les données produites par les applications s'exécutant sur le poste client (telles que les applets Java™ ou les fichiers Macromedia Flash™ qui génèrent des contenus multimédias plus riches que des applications "ordinaires") ou celles issues d'applications produisant du contenu dynamique (comme PHP ou Perl qui est basé sur la technologie CGI) ne sont pas recueillies, empêchant ainsi de connaître les informations réellement consultées par les utilisateurs (Fansler et Riegel, 2004).

Pour pallier aux inconvénients présentés par les approches fondées sur l'analyse des fichiers de *log*, d'autres approches proposent des mécanismes d'observation spécifiques à un outil d'apprentissage particulier.

2.2.2. Les approches spécifiques

L'application eMediathèque de la classe virtuelle eLycée (eLycée, 2007) est un outil de travail collaboratif interactif qui vise à améliorer les activités réflexives lors d'un apprentissage collaboratif. Toutes les interactions de l'utilisateur avec cet outil sont observées et traitées à partir de deux modèles (Michel et al., 2005) : l'un définit les objets observables (les outils mis à disposition tels que le navigateur internet ou la messagerie instantanée, mais aussi les ressources telles que les textes, les images, etc.) et l'autre spécifie les actions réalisables par l'utilisateur (création, modification, suppression de contenus, etc.). Les traces sont ensuite créées en fonction des activités des utilisateurs, stockées dans un composant interne de l'application, et affichées en temps réel à l'utilisateur.

Dans le cas du Knowledge Pool System (KPS) de la fondation ARIADNE (Ariadne, 1996), chaque interaction d'un utilisateur avec l'outil SILO (l'interface entre les utilisateurs et le KPS) est enregistrée dans un fichier au format XML (Najjar et al., 2004). Chaque enregistrement donne des informations telles que le titre des ressources ayant été indexées, téléchargées ou supprimées, l'utilisateur à l'origine de l'enregistrement, la date de création de l'enregistrement. Cependant, des opérations complexes telles que le dépouillement du fichier, sa décomposition et la mise en relation des différentes informations doivent être effectuées pour calculer les statistiques liées à un objet pédagogique particulier. De plus, ce fichier est centralisé dans chacun des viviers institutionnels qui constituent le KPS global ; des efforts supplémentaires sont donc nécessaires pour obtenir une vision générale de l'utilisation des objets pédagogiques au sein du vivier global.

Si les approches spécifiques à un outil particulier présentent certains atouts qui comblent les lacunes des approches fondées sur les fichiers de *log* (acquisition et visualisation des traces en temps réel, traces recueillies en fonction des activités des utilisateurs, informations précises et détaillées), elles souffrent de leur cloisonnement et de leur aspect propriétaire. Les traces collectées par les systèmes mentionnés ci-dessus présentent un format qui leur est spécifique et qui empêche leur traitement par d'autres systèmes à base de traces (SBT). D'autre part, les traces sont renfermées au sein d'un système spécifique et ne peuvent pas être réutilisées.

2.3. Les verrous à lever et leurs enjeux

Cette étude nous conduit à proposer un cadre de gestion des traces qui intègre les avantages des approches spécifiques, et qui comble leurs manques. Notre approche doit donc être capable de :

- représenter de façon uniforme (standardisée) les entités à observer pour permettre la collecte de traces issues de systèmes hétérogènes ;

- définir ce qui est tracé afin de représenter les activités avec une sémantique de plus haut niveau que celle offerte par les fichiers de *log*, dans le but de rendre les traces plus facilement interprétables (Héraud et al., 2005) ;

- assurer le stockage des traces dans des systèmes dédiés et distincts des systèmes à observer afin de favoriser le partage des traces, tout en offrant la possibilité de facilement exploiter ces systèmes de stockage pour bénéficier de l'expérience de nombreux utilisateurs ;

- prendre en compte le caractère distribué des systèmes d'apprentissage en ligne.

D'un point de vue organisationnel, le franchissement de ces barrières permettrait de limiter le processus de recherche d'informations. Des études ont montré que des sujets à la recherche de connaissances préfèrent rechercher dans leur application de messagerie électronique ou dans leur disque dur local plutôt que dans les systèmes de gestion de documents (Swaak et al., 2004). Partant de ce constat, les nouvelles technologies de l'information et de la communication, associées à l'accès et à l'analyse de différents référentiels de traces responsables de diverses applications, offriraient l'opportunité de tirer partie de ce type d'outils en informant directement les utilisateurs de (nouvelles) ressources (humaines ou électroniques) disponibles. Dans le cas d'un apprenant, du contenu pédagogique en relation avec son cursus d'apprentissage pourrait lui être délivré en complément des ressources disponibles au sein de son EIAH. A plus long terme, la gestion de traces issues de différents systèmes, applications et utilisateurs laisse entrevoir la possibilité de construire dynamiquement des environnements d'apprentissage adaptés non seulement au contexte pédagogique, mais également au profil des utilisateurs visés.

Des outils en ligne tels que les moteurs de recherche proposés par Google™ ou Amazon™ suggèrent à l'utilisateur des ressources qui correspondent à leurs attraits favoris, sur la base d'activités antérieures réalisées par de nombreux utilisateurs. Ce mécanisme peut facilement être mis en œuvre dans ce type de systèmes, car les utilisateurs exploitent tous le même outil pour réaliser leurs activités. En revanche, cette approche ne peut être transposée au contexte de l'apprentissage en ligne, puisque les acteurs de l'e-formation utilisent les outils spécifiques à leur contexte d'apprentissage. Une externalisation des traces d'usage renfermées dans chacun de ces systèmes vers un module indépendant de ceux-ci offrirait la possibilité de bénéficier de l'expérience d'un nombre beaucoup plus important d'utilisateurs, et ainsi d'augmenter l'efficacité des processus évoqués dans la section 2.1.

D'autre part, le partage des traces permettrait d'élargir les communautés de pratiques qui, aujourd'hui, se limitent souvent aux membres d'une même organisation. Pourtant, les cursus pédagogiques sont pour la plupart reproduits dans plusieurs institutions, et les acteurs d'une structure pédagogique particulière devraient être capables de découvrir leurs homologues exerçant au sein d'un autre EIAH, et de mutualiser ainsi leurs expériences en termes de pédagogie et/ou de pratiques.

3. Une approche conduite par les modèles pour la gestion des traces

Pour satisfaire l'ensemble de ces contraintes, nous proposons de :

- faire le choix d'un métamodèle commun pour la modélisation universelle des entités à observer ;

- élaborer un modèle de traces qui permet de spécifier les activités à observer au sein des systèmes d'apprentissage et dont la représentation permet d'ajouter une sémantique claire qui donne un sens aux traces ;

- adopter une architecture de supervision distribuée et décentralisée qui offre la possibilité de consulter une partie ou l'ensemble des systèmes dédiés au stockage des traces selon différents niveaux d'abstraction ;

- exploiter des protocoles et modes de communication standards capables d'utiliser l'internet pour véhiculer les traces d'activité des utilisateurs.

Certains travaux menés au sein de notre équipe de recherche qui traitent de la gestion de réseaux et de systèmes ont mis en avant les atouts d'une approche orientée modèle, en particulier à partir de l'initiative proposée par le DMTF (Distributed Management Task Force), pour proposer des solutions à la supervision et à l'intégration d'entités hétérogènes. En effet, l'approche de modélisation CIM (Common Information Model) (DMTF, 1999) spécifie un métamodèle sur la base des concepts Objet qui répond à des besoins d'unification et d'extensibilité, alors que l'architecture WBEM (Web Based Enterprise Management) (WBEM, 1999) propose une organisation distribuée du cadre de supervision.

Le métamodèle CIM et son ensemble de diagrammes représentent un choix approprié pour atteindre nos objectifs de partage et de réutilisation de traces issues de systèmes hétérogènes car son approche par niveaux d'abstraction

offre plusieurs avantages incontestables.

- Un modèle de gestion unificateur : toutes les classes identifiant des éléments à observer héritent de la classe *CIM_ManagedElement*. En proposant CIM, le DMTF avait pour objectif d'apporter une solution d'intégration des approches de supervision, puis à plus long terme d'unification. Sa politique de partenariat avec les autres organismes de standardisation en témoigne.

- Une approche extensible par métier/domaine : CIM offre la possibilité de définir les modèles génériques d'un domaine (Réseau, Système, etc.) ou métier (Spatial, Finance, E-formation, etc.) particulier, ainsi que des modèles spécifiques à un projet ou environnement donné.

- Des choix conceptuels non spécifiques au monde de la gestion : concepts Objet, utilisation des diagrammes de classes UML avec quelques particularités propres, syntaxe MOF (Managed Object Format) décrite par une DTD XML.

- Le langage de requête CIM Query Language (DMTF, 2006) permet d'obtenir les classes et instances des différents éléments observés qui sont définis dans le modèle selon des niveaux d'abstraction plus ou moins élevés.

- Le métamodèle inclut la modélisation des utilisateurs à travers le modèle *CIM User* (DMTF, 2003d) qui représente les personnes et leurs caractéristiques associées.

- Des initiatives *open source* : depuis l'an 2000, des plates-formes *open source* ont vu le jour afin de promouvoir l'adoption de CIM (WBEM Services, 2003).

- Notre expérimentation d'intégration et de distribution avec CIM : nos travaux menés dans le cadre d'un stage de DEA à l'IRIT ont abouti à l'interopérabilité entre deux plates-formes de Gestion de Réseau.

D'autre part, l'architecture WBEM satisfait les contraintes mentionnées dans la section 2.3.

- Certaines entités appelées *Object Provider* sont dédiées à l'intégration d'environnements et de systèmes hétérogènes.

- D'autres entités nommées *Object Manager* qui renferment les traces d'observation, sont distribuées et peuvent être facilement consultées à travers leurs fonctions internes de gestion. Différents services et applications ont donc l'opportunité d'exploiter les informations stockées dans chacune de ces entités.

- Le protocole de transport mis en œuvre entre les entités *provider* et *manager* assure la mise à l'échelle : les informations CIM sont véhiculées via les protocoles XML/HTTP qui facilitent la traversée des pare-feux ou *proxy* présents au sein d'une architecture réseau.

La prochaine partie présente tout d'abord les principes fondamentaux du métamodèle CIM qui justifient notre choix, et propose un modèle UML générique de traces d'activité. Une architecture distribuée et décentralisée fondée sur l'approche WBEM qui assure la prise en compte du modèle défini est ensuite exposée, et démontre comment des traces d'activités issues d'EIAH variés peuvent être interprétées, partagées et exploitées par une large communauté.

3.1. CIM : un métamodèle commun de l'information

L'approche de modélisation CIM spécifie un métamodèle sur la base d'un modèle commun d'abstraction décrivant une connaissance sémantique du monde à observer. Les éléments de base suivants sont utilisés pour modéliser les entités à gérer (cf. figure 1).

- Objet géré/Classe : les objets gérés sont tous dérivés de la classe *CIM_ManagedElement*. Toute classe d'objet est spécifiée par un ensemble de propriétés et de méthodes.

- Opération/Action : une action correspond à une opération à invoquer sur un élément du modèle commun d'abstraction. Elle correspond à une méthode spécifique définie dans la description de la classe.

- Relation : deux types de relation sont considérés dans CIM :

• Une relation d'héritage : les classes peuvent dériver d'au plus une classe (héritage simple).

• Une association mettant en relation au moins deux classes d'objets. Parmi les associations, deux associations spécifiques ont été définies, à savoir la composition (*CIM_Component*) qui exprime une relation de composition, et la dépendance (*CIM_Dependency*) qui traduit une sémantique existentielle ou fonctionnelle.

- Evénement : un événement est une instance de la classe *CIM_Indication*. Un graphe d'héritage est défini dans le modèle commun d'abstraction ainsi qu'un modèle permettant la levée et l'abonnement à ces indications (DMTF, 2003e).

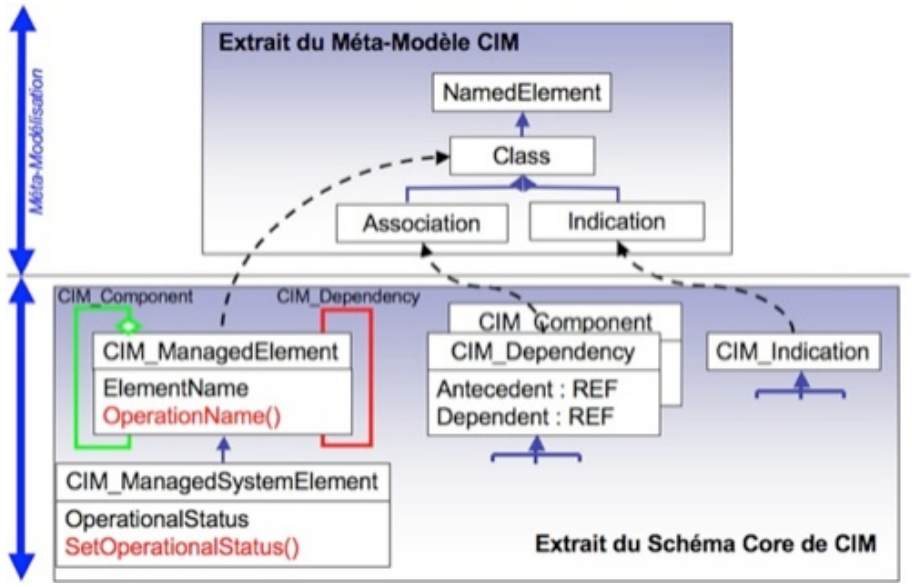

Figure 1 • Les éléments de base de la modélisation CIM

Afin de répondre aux besoins d'unification et d'extensibilité, CIM offre une uniformisation des informations de gestion en proposant un ensemble de schémas divisé en trois niveaux distincts.

- Le modèle *CIM Core* (DMTF, 2000) : il définit un ensemble de classes et d'associations génériques à tout domaine de gestion, c'est-à-dire une structure de base pour tous les schémas futurs qui concernent des domaines particuliers. Ainsi, le modèle *CIM Core* est stable et comporte un nombre assez limité de classes.

- Le modèle *CIM Common* : il définit des classes et des associations indépendantes de toute implantation, relatives à un domaine particulier de la gestion. Il se compose de différents sous-modèles qui concernent les applications (DMTF, 2003a), systèmes (DMTF, 2003b), réseaux (DMTF, 2003c), etc.

- Les modèles *CIM Extension* : les schémas *CIM Extension* permettent d'étendre les classes existantes dans le modèle *CIM Common*.

A partir des modèles CIM existants, la section suivante s'attache à proposer un modèle générique de traces capable de représenter les activités des utilisateurs au sein de systèmes d'apprentissage en ligne hétérogènes.

3.2. Un modèle UML générique de traces pour les activités

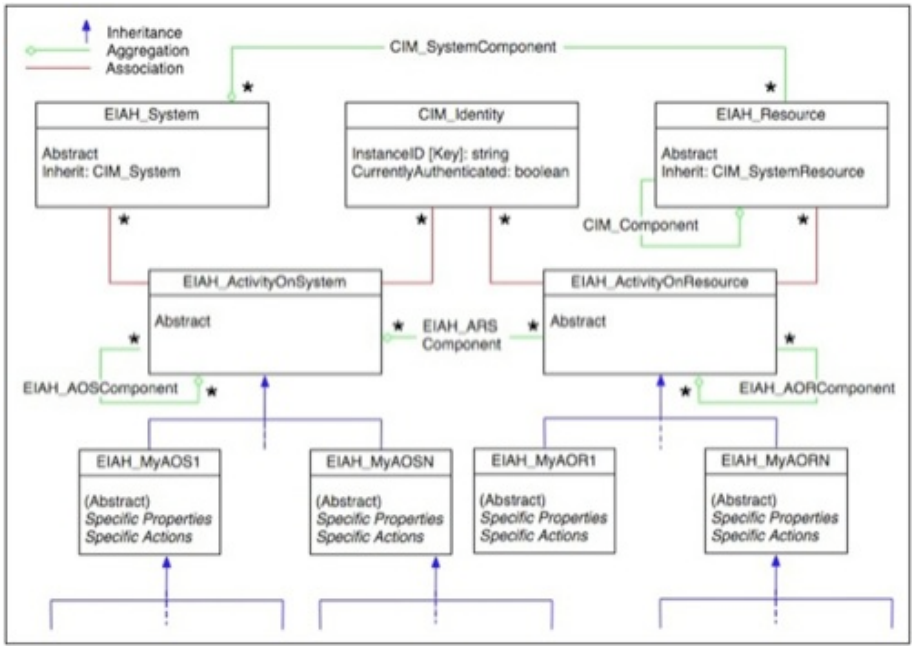

Figure 2 • Un modèle UML générique de traces pour les activités

Nous souhaitons élaborer un modèle dont un des objectifs concerne la généricité, et qui permet de spécifier les traces produites par les activités explicites des utilisateurs dans un système d'apprentissage instrumenté par le *Web*.

La modélisation UML illustrée par la figure 2 s'organise donc autour de quatre concepts principaux : les systèmes d'apprentissage en ligne, les ressources intégrées dans ces systèmes, les utilisateurs, et les activités que ces derniers peuvent réaliser sur les systèmes et ressources. Les trois premiers concepts ont été modélisés par des objets gérés, alors que les activités sont représentées par des relations d'association entre objets gérés.

La classe abstraite *EIAH_System* modélise les traces à collecter pour les systèmes d'apprentissage, alors que la classe abstraite *EIAH_Resource* décrit d'un point de vue observation toute ressource intégrée dans un système d'apprentissage, et qui est disponible pour ce système. Ces classes héritent toutes deux de classes CIM prédéfinies dans le métamodèle, et bénéficient naturellement d'une relation de composition *CIM_SystemComponent* qui permet de connaître les différentes ressources qui constituent un système d'apprentissage. La relation de composition *CIM_Component* entre des instances de la classe *EIAH_Resource* permet d'exprimer le fait qu'une ressource d'apprentissage peut être constituée d'une ou plusieurs autres ressources.

Pour des raisons de simplification, la figure 2 ne présente pas l'ensemble des classes définies dans CIM qui sont dédiées à la spécification des informations à recueillir pour un utilisateur. Nous avons simplement réutilisé les classes existantes du modèle *CIM User* (DMTF, 2003d).

Parmi l'ensemble des activités qui peuvent être proposées dans un EIAH, nous distinguons les activités réalisées sur le système d'apprentissage lui-même de celles qui sont opérées sur une ressource intégrée dans le système. Ces deux types d'activité sont respectivement modélisées par les relations abstraites d'association *EIAH_ActivityOnSystem* et *EIAH_ActivityOnResource.*

- La première associe un utilisateur à un système d'apprentissage, et s'intéresse en particulier aux activités fonctionnelles, de maintenance et de configuration qui sont généralement effectuées par les administrateurs des systèmes et applications. Les traces de certaines activités comme l'ajout ou la suppression d'une fonctionnalité, la mise à jour, ou le changement de l'interface homme/machine d'un système d'apprentissage sont donc spécifiées par la spécialisation de cette classe. La relation de composition *EIAH_AOSComponent* dénote le fait qu'une telle activité peut être constituée d'une ou plusieurs sous-activités.

- La seconde référence un utilisateur et une ressource intégrée dans un système, et se focalise sur les activités d'apprentissage réalisées par les apprenants et enseignants-tuteurs sur les ressources. Les traces d'activité à collecter lors de la création d'un nouveau cursus, de l'envoi d'un message dans un forum de discussion, du dépôt d'un devoir, ou de la participation à une séance de messagerie instantanée sont modélisées à partir de cette classe en définissant de nouvelles relations qui caractérisent ces activités. La relation de composition *EIAH_AORComponent* permet de déterminer les sous-activités qui constituent une activité de plus haut niveau.

Enfin, la relation de composition *EIAH_ARSComponent* associe les deux types d'activité identifiés, et signifie qu'une action exécutée sur une ressource peut constituer une partie d'une activité réalisée sur un système d'apprentissage.

Nous avons proposé ici un modèle UML dédié à la spécification des traces d'activités à collecter lors de l'usage d'un système d'apprentissage par un utilisateur. La section suivante s'intéresse quant à elle à l'architecture informatique distribuée et décentralisée que nous avons adoptée pour assurer la gestion, le partage et la réutilisation de ces traces.

3.3. Une architecture distribuée et décentralisée pour le suivi des activités au sein d'EIAH

3.3.1. Les fondements : l'architecture WBEM

Le DMTF propose, au travers de l'architecture WBEM associée au modèle CIM, une distribution des composants de supervision en introduisant les notions d'*Object Manager* et d'*Object Provider* qui présentent des interfaces génériques basées sur les concepts CIM de classe, propriété, méthode et instance. Ces deux composants sont respectivement désignés par CIM OM et CIM OP dans la suite du document.

Un CIM OM est une application qui a pour charge d'assurer le stockage et l'intégrité des informations relatives à un ou plusieurs domaines d'observation. Pour cela, il dispose d'un référentiel contenant la description de sa connaissance de gestion sous forme de classes CIM, ainsi que cette connaissance représentée par des instances de classes CIM. Le référentiel est exploité par des fonctions de gestion intégrées au sein du CIM OM.

Un CIM OP est une entité de gestion responsable de l'intégration d'un environnement à gérer comme l'environnement Unix ou un environnement de base de données Oracle™ ; pour chaque type d'environnement à observer, un CIM OP est nécessaire. Ils permettent d'offrir une vue homogène des environnements hétérogènes à intégrer.

Le monde hétérogène à observer est instrumenté par l'intermédiaire de composants logiciels, accessibles via des protocoles standards ou spécifiques.

Enfin, des applications clientes restituent sous la forme d'interfaces graphiques orientées utilisateur les informations de gestion distribuées auprès des CIM OM, et offrent aux gestionnaires humains la possibilité de réaliser des actions (mise à jour, invocation de méthode, etc).

L'architecture WBEM permet donc de collecter des informations de supervision à travers les composants logiciels intégrés dans les systèmes à observer et le CIM OP, de stocker ces informations dans un référentiel CIM par le biais du CIM OM, et enfin d'exploiter ces informations en consultant les classes et instances contenues dans le référentiel. Sur la base de cette architecture, nous proposons dans la section suivante une architecture distribuée et décentralisée pour le suivi des activités des utilisateurs au sein d'un EIAH.

3.3.2. L'organisation distribuée et décentralisée

3.3.2.1. Processus de génération et de stockage des traces

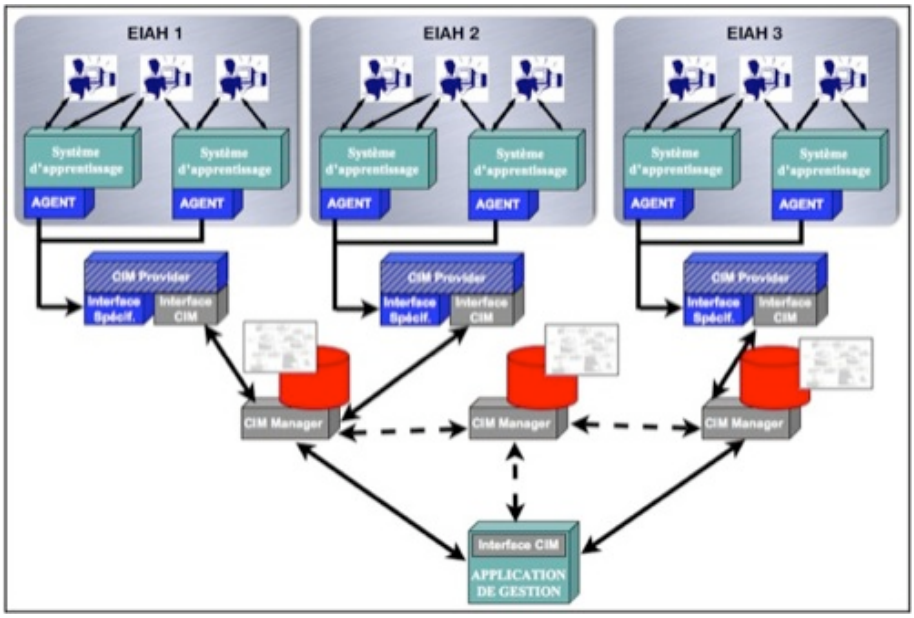

Figure 3 • Une organisation distribuée et décentralisée pour la gestion des traces

Parmi les différentes propositions de distribution des entités de gestion énoncées dans (Martin Flatin, 2003), nous pourrions qualifier notre architecture de fortement distribuée puisque tous les CIM OM sont des composants distribués et qu'ils ont, de base, les mêmes fonctionnalités. Par ailleurs, leur répartition est fonction des structures pédagogiques souhaitant partager leurs traces d'activités, alors que la distribution des CIM OP est structurée selon les environnements d'apprentissage des institutions qui composent une structure pédagogique. Sur la figure 3, les EIAH 1 et 2 constituent les environnements d'apprentissage de deux institutions différentes comprises dans une même structure de pédagogie, alors que l'EIAH 3 correspond à une autre structure. Ces choix assurent :

- le partage de l'ensemble des traces issues des systèmes hétérogènes d'une même structure de pédagogie, mais également la mise à disposition des traces renfermées dans les différents CIM OM à n'importe quel outil fondé sur les technologies *Web* ;
- la prise en compte du caractère distribué des environnements d'apprentissage ;
- la répartition des charges de traitement des traces sur différents composants logiciels ;
- l'extensibilité et l'ouverture de l'architecture de gestion (cf. section 3.4).

Les composants logiciels intégrés dans les systèmes à observer ont pour fonction de capturer les traces brutes, c'est-à-dire chacune des informations correspondant aux propriétés définies dans les classes du modèle UML de traces qui traduisent la réalisation d'une activité par un utilisateur (l'activation du processus doit être effectuée par l'administrateur lors de la configuration du système d'apprentissage). Ces composants logiciels transmettent ensuite l'ensemble des données capturées ainsi que la nature de l'activité réalisée à un CIM OP qui joue le rôle d'intermédiaire entre l'environnement d'apprentissage et l'environnement de gestion des traces.

Un CIM OP est constitué de deux interfaces : l'une est dédiée à la réception des traces brutes transmises par les composants logiciels, alors que l'autre est responsable de la création ou de la modification des traces d'activité à travers l'invocation de méthodes implantées dans le CIM OM qui permettent de créer de nouvelles instances ou de modifier des instances existantes de classes définies dans le modèle UML de traces.

En effet, un CIM OM est capable de réaliser des opérations relatives aux traces d'activités dans la mesure où le modèle UML de la figure 2 est décrit par le langage MOF et injecté dans son référentiel (des extraits de fichiers MOF sont donnés dans la partie 4). A partir d'une DTD XML décrivant le langage MOF, un CIM OM stocke les classes et instances de classes CIM sous la forme de documents XML valides ; la création, la modification ou la suppression d'une instance d'une classe correspond donc à la modification du document XML correspondant, la manière de gérer ces documents étant spécifique d'une implémentation de l'architecture WBEM à une autre.

3.3.2.2. Consultation des référentiels de traces

Le DMTF a introduit le langage de requête CIM Query Language (CQL) pour permettre à des applications clientes d'envoyer des requêtes vers un CIM OM afin de retrouver les classes et instances contenues dans son référentiel. Les applications clientes exploitent ce langage pour spécifier la nature et le nombre d'instances qui doivent être sélectionnées, ainsi que les propriétés à retourner pour chacune des instances.

CQL est composé d'un ensemble de fonctionnalités de base auxquelles s'ajoutent des fonctionnalités optionnelles qui déterminent la complexité de la syntaxe et de la sémantique des requêtes. Les principes les plus importants de CQL sont : requête simple, jointure simple ou complexe, liste de sélection étendue, agrégation, et mécanisme d'expression régulière. L'ensemble des fonctionnalités proposées par CQL est à la disposition du lecteur dans ce document (DMTF, 2006).

Une vue homogène des traces d'activités renfermées dans différents CIM OM peut alors être obtenue selon deux approches (cf. figure 3):

- un CIM OM additionnel a pour rôle d'assurer la fédération de plusieurs référentiels de traces ; ce CIM OM pourra être interrogé par une application cliente pour retrouver l'ensemble des traces ;

- la jointure des traces est laissée à la charge de l'application orientée utilisateur qui restitue de manière interprétable les traces d'activité en envoyant des requêtes CQL aux différents CIM OM.

3.3.2.3. Protocoles et modes de communication

Le protocole de communication proposé par le DMTF s'appuie sur deux standards du *Web* : XML pour l'encodage et HTTP pour le transport. Le DMTF a opté pour l'utilisation du protocole HTTP étendu (Nielsen et al., 2000) qui consiste à encapsuler la description XML des informations CIM dans le corps d'une unité de données HTTP. Ces spécifications CIM/HTTP ont été publiées en 2002 et plusieurs plates-formes adoptent ces spécifications.

Les communications mettant en jeu un CIM OM sont donc fondées sur le protocole XML/HTTP. La transmission des traces brutes capturées par les composants logiciels au sein des systèmes à observer vers les CIM OP doit quant à elle être assurée par n'importe quel protocole de communication supportant les technologies *Web*.

Dans le monde des systèmes distribués, les modes *Pull* et *Push* désignent deux approches pour échanger des données entre entités distantes. Dans le cadre de la gestion de réseaux et de systèmes, le mode *Pull* est basé sur le paradigme Client/Serveur : le client envoie une demande au serveur, puis le serveur répond de manière synchrone ou asynchrone. Le mode *Push*, à l'inverse, est basé sur le patron conceptuel "Publieur/Souscripteur" ou encore "Publieur/Consommateur". Le client peut s'abonner pour recevoir des événements, chaque serveur prenant individuellement l'initiative d'envoyer les événements aux souscripteurs.

Dans notre architecture, le mode *Push* est le mode par lequel les composants intégrés dans les systèmes d'apprentissage notifient en temps réel les CIM OP d'une activité réalisée par un utilisateur. Ce mode de communication est utilisé entre CIM OP et CIM OM pour propager toute notification issue des composants intégrés dans les EIAH à superviser, mais également dans le cas d'échanges entre CIM OM, suite à une requête cliente effectuée par exemple par un service souhaitant exploiter les traces. Le mode *Pull* intervient seulement dans ce type d'interaction, puisque les applications clientes cherchent à retrouver des informations existantes qui sont renfermées dans les référentiels CIM ; elles n'ont pas pour objet de notifier les activités des utilisateurs.

3.4. Capacité d'adaptation du cadre de gestion des traces

Les environnements d'apprentissage sont loin d'être des environnements figés et sont sujets à de nombreuses évolutions. Nous étudions dans cette section comment notre approche peut s'adapter à différents changements en termes d'organisation d'un EIAH ou d'activités additionnelles à observer.

3.4.1. Ajout/modification d'un système dans un EIAH

Lorsqu'un nouveau système est introduit dans un environnement d'apprentissage, deux cas de figure doivent être distingués : soit le composant logiciel spécifique au système à observer a déjà été implémenté, soit il n'existe pas. Dans la première situation, l'unique tâche à accomplir est d'introduire le composant dans le système d'apprentissage. Des approches fondées sur la programmation orientée aspect sont envisageables pour réaliser cette opération puisqu'elles ont pour fonction principale la modification du code source d'une application en y intégrant de nouvelles fonctionnalités (Tarby et al., 2007) (Leclet et al., 2007). Si le composant logiciel n'existe pas, une phase de création nécessitant l'étude du système à observer doit précéder le processus décrit ci-dessus. Cela ne représente pas un travail trop lourd puisque la tâche du développeur ne consiste qu'à identifier les informations à capturer au sein du nouveau système avant d'exploiter le mécanisme de transmission des informations au CIM OP.

Les outils informatiques ne cessent d'évoluer, comme en témoignent les nombreuses mises à jour régulièrement proposées aux utilisateurs. Dans le cas d'une nouvelle version d'un système d'apprentissage qui apporte des changements significatifs à la structure de l'application, la seule entité à modifier dans notre architecture est le composant logiciel qui doit satisfaire les nouvelles spécifications du système d'apprentissage.

3.4.2. Observation d'activités additionnelles

L'observation d'une activité additionnelle implique la modification de plusieurs entités de l'architecture de gestion des traces.

- Le modèle UML de traces d'activité : les traces brutes à collecter qui sont relatives à la nouvelle activité à observer doivent être décrites dans le modèle UML de traces de la figure 2 afin d'injecter les classes MOF correspondantes dans le ou les CIM OM responsables de l'observation de ces activités.

- Les composants logiciels : sans une mise à jour, ils ne seront pas capables de capturer les informations relatives à la nouvelle activité à observer qui sont définies dans le modèle UML de traces.

- Les CIM OP : à partir des traces brutes collectées par les composants logiciels, ils doivent être en mesure de créer ou modifier les instances CIM qui traduisent la réalisation de la nouvelle activité par un utilisateur.

Même si trois entités sont modifiées lors de l'observation d'une activité supplémentaire, seule la tâche d'instrumentation des CIM OP requiert un investissement non négligeable en terme de temps de travail. Mais une application cliente pourra alors disposer d'une vue uniforme et cohérente de différentes activités effectuées au sein de multiples outils d'apprentissage déployés dans différentes infrastructures de formation en ligne.

3.5. Positionnement de notre approche

L'approche orientée modèle permet de spécifier les traces à acquérir pour atteindre les objectifs pédagogiques visés par l'observation des activités des utilisateurs. Si ces travaux (Héraud et al., 2005) mettent en avant les avantages d'une représentation des objets observés avec un haut niveau d'abstraction et exprimant les activités avec une sémantique claire, ils dénotent également que cette approche empêche de considérer les activités qui n'ont pas été prédites dans le modèle et qui sont pourtant utiles en terme d'analyse des activités d'un utilisateur. Une proposition intégrant différentes sources d'informations (celles contenues dans les fichiers de *log*, celles issues des activités explicites des utilisateurs, etc.) ainsi que des mécanismes capables de générer des traces combinées et interprétables sont alors suggérés. Le métamodèle CIM élaboré à ce jour propose de nombreux modèles, et les entités de gestion spécifiques à une source de données prédominante sont d'ores et déjà implémentées ; les composants nécessaires à la supervision de certaines activités des systèmes d'exploitation Windows et Unix sont notamment disponibles. Les traces d'usage du système d'exploitation peuvent alors être intégrées dans le même CIM OM que celui renfermant les traces des activités explicites des utilisateurs sur un système d'apprentissage. L'ensemble des traces étant représentées de façon uniforme, aucun mécanisme de transformation n'est nécessaire.

Plusieurs travaux s'intéressant à l'exploitation de traces en vue d'une réingénierie des scénarios pédagogiques ou des interfaces graphiques sont fondés sur des systèmes multi-agents (Carron et al., 2006) (Michel et al., 2005) (Laflaquière et Prié, 2003). Le système de gestion de traces est généralement composé d'un agent responsable de la collecte des traces brutes, d'un agent capable de structurer les traces brutes en traces interprétables, et d'un agent de visualisation qui restitue à travers une interface les traces interprétables à l'utilisateur. (Carron et al., 2006) a toutefois soulevé deux difficultés à surmonter dans ces approches :

- un système multi-agents est plus difficile à mettre en œuvre et à superviser qu'un système centralisé car les agents sont fortement dépendants entre eux ;

- les agents qui sont implantés dans les stations des utilisateurs entraînent une montée en charge significative des processeurs.

L'architecture distribuée et décentralisée tente de combler ces manques : les entités de l'architecture de gestion des traces sont fortement indépendantes les unes des autres (cf. section précédente), et les traitements à exécuter sur les stations des utilisateurs sont réduits au minimum (seul le composant logiciel responsable de la capture des données est intégré dans le système à superviser). Notons également que même si la masse globale des traces stockées dans des référentiels décentralisés est importante, la durée nécessaire au traitement de l'ensemble des traces par un logiciel client n'est pas proportionnelle à cette masse globale puisqu'il bénéficie des traitements simultanés de plusieurs CIM OM.

Les travaux présentés dans (Iksal et Choquet, 2005a) et évoqués en section 2.1 ont défini le langage UTL afin d'ajouter de la sémantique aux traces collectées à partir de fichiers de *log*, et d'extraire des informations statistiques ou de comparer des scénarios d'apprentissage observés à ceux prédits par le concepteur de cursus. Dans notre approche, la sémantique des activités observées est exprimée au sein même du modèle UML de traces, alors que le langage CQL permet de restituer ces traces selon différents niveaux d'abstraction. Ce processus est facilité par la description UML des informations disponibles d'une part, et par le caractère normalisé de CQL d'autre part. Celui-ci est fondé sur les langages SQL de l'ISO et XML-Query du W3C, ce qui en facilite sa prise en main et l'élaboration de requêtes complexes.

4. Cas de l'exploitation des objets pédagogiques

Nous avons présenté dans (Broisin et Vidal, 2005) une architecture pour la virtualisation des objets pédagogiques (cf. partie gauche de la figure 5) fournissant à la fois une vue unifiée d'un ensemble de ressources stockées dans

différents viviers de connaissance, et un accès facilité à ces ressources à travers les plates-formes d'apprentissage. Cette architecture offre une interopérabilité entre LOR et LMS et permet aux utilisateurs, à partir des plates-formes d'apprentissage, d'exploiter de façon transparente les objets pédagogiques renfermés dans différents viviers de connaissance. Cette architecture a été implémentée avec deux plates-formes *open source*, respectivement INES et MOODLE, et quatre viviers de connaissance : le KPS de la fondation ARIADNE, MERLOT, EDNA et le LRC.

Nous proposons dans cette partie une application de l'approche présentée dans la partie précédente. Le cadre de travail présenté assure la gestion des traces qui traduisent les activités réalisées sur des objets pédagogiques par les utilisateurs impliqués dans l'architecture de virtualisation ; aucune activité effectuée sur les systèmes d'apprentissage n'est proposée. L'outil SILO est également intégré dans notre proposition, validant ainsi le caractère flexible de notre solution.

4.1. Un modèle UML de traces d'activités pour les objets pédagogiques

A partir du modèle UML générique de traces pour les activités illustré par la figure 2, nous avons élaboré un modèle qui s'attache, en spécialisant les classes abstraites modélisant les systèmes, ressources et activités d'un EIAH, à représenter les traces des activités qui peuvent être réalisées sur des objets pédagogiques. Le modèle résultant est exposé par la figure 4 ; nous ne décrivons ici que les principales classes du modèle, le lecteur peut consulter (Broisin, 2006) pour de plus amples détails.

Des plates-formes pédagogiques et viviers de connaissance constituent notre environnement d'apprentissage. La classe *EIAH_LearningManagementSystem* et la classe *EIAH_ContentManagementSystem* héritent de la classe abstraite *EIAH_System*, et permettent de représenter les traces brutes à capturer lors d'activités réalisées sur ces systèmes. Quatre attributs sont définis pour ces deux classes, et indiquent leurs nom, rôle, localisation et description. Pour les plates-formes de téléformation, la version du système est également disponible, alors que les viviers de connaissance sont décrits à l'aide d'un attribut additionnel qui spécifie le standard de méta-données utilisé par le système de stockage.

Deux types de ressources sont identifiés dans notre contexte : les objets pédagogiques et les cursus d'apprentissage. Les traces à collecter pour les objets pédagogiques (classe *EIAH_LearningObject*) concernent des données statistiques (le nombre de consultations des méta-données, le nombre de téléchargements, le nombre d'intégrations de chaque objet au sein d'un cursus d'apprentissage), ainsi que des repères temporels (date de création, dates de consultation, de téléchargement et d'intégration de l'objet pédagogique). Un ensemble de méthodes est associé aux attributs précités afin de modifier leur valeur. La classe *EIAH_Courseware* décrit un cursus pédagogique délivré par un LMS ; en plus des propriétés définies par sa classe parente *EIAH_Resource*, elle spécifie la discipline concernée par cet enseignement.

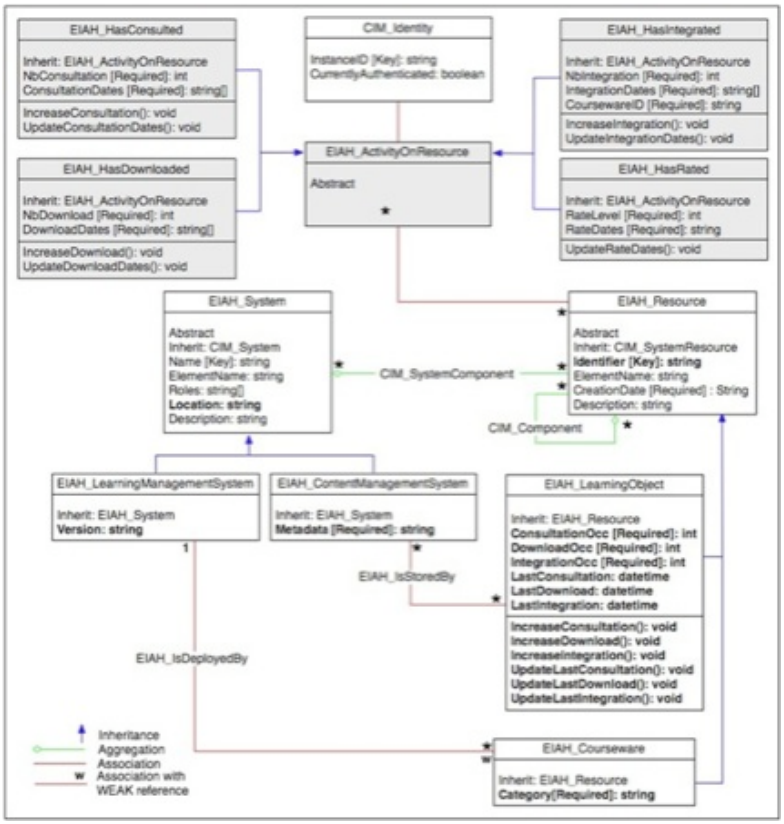

Figure 4 • Un modèle UML de traces d'activités pour les objets pédagogiques

Afin d'exprimer les relations existant entre systèmes et ressources d'apprentissage et ainsi d'apporter une

sémantique aux traces représentées par les classes précitées, nous avons défini deux relations d'association : *EIAH_IsStoredBy* indique le(s) vivier(s) de connaissance qui renferme(nt) un objet pédagogique spécifique (et réciproquement, l'ensemble des objets stockés dans un vivier particulier), alors qu'*EIAH_IsDeployedBy* renseigne la plate-forme d'apprentissage déployant un cursus pédagogique donné.

Aucun effort de modélisation n'a dû être entrepris pour représenter les utilisateurs des systèmes d'apprentissage, les classes nécessaires étant définies dans le modèle existant *CIM User* (DMTF, 2003d).

Nous avons recensé six activités pouvant être réalisées sur un objet pédagogique par un utilisateur à partir des plates-formes d'apprentissage :

- la recherche d'objets pédagogiques renfermés dans des viviers de connaissance et qui correspondent à des mots clés particuliers renseigne sur les concepts recherchés par un utilisateur, spécifiés dans le modèle *CIM User* ;

- la consultation des métadonnées de ces objets d'apprentissage est modélisée par la relation d'association *EIAH_HasConsulted* ;

- le téléchargement des documents correspondants sur leur poste local apparaît à travers la relation *EIAH_HasDownloaded* ;

- l'importation des ressources au sein d'un cursus de la plate-forme d'apprentissage est représentée par l'association *EIAH_HasIntegrated* ;

- l'indexation de nouveaux objets au sein de la plate-forme et du vivier de connaissance cible correspond, à une variante près, à l'activité d'importation ; nous n'avons pas spécifié de classe spécifique ;

- l'évaluation d'un objet pédagogique est exprimée à l'aide de la classe *EIAH_HasRated*.

Les classes exprimant les relations d'association mentionnées ci-dessus héritent toutes de la classe abstraite *EIAH_ActivityOnResource*. Elles référencent donc l'utilisateur et l'objet pédagogique concernés par l'activité réalisée au sein du système d'apprentissage. Un attribut permet de connaître les dates auxquelles l'utilisateur a réalisé l'activité.

Nous avons établi dans cette section un modèle UML de traces pour le suivi des activités relatives aux objets pédagogiques conforme au métamodèle CIM. Le modèle représente les différentes traces que nous souhaitons collecter lors de la réalisation d'une activité ; la section suivante s'intéresse à l'intégration de l'organisation distribuée et décentralisée au sein de l'EIAH existant.

4.2. L'architecture pour la gestion des traces

L'architecture globale de l'EIAH illustrée par la figure 5 prend en compte l'architecture de virtualisation, à laquelle nous avons intégré les composants spécifiés par l'initiative WBEM et présentés précédemment.

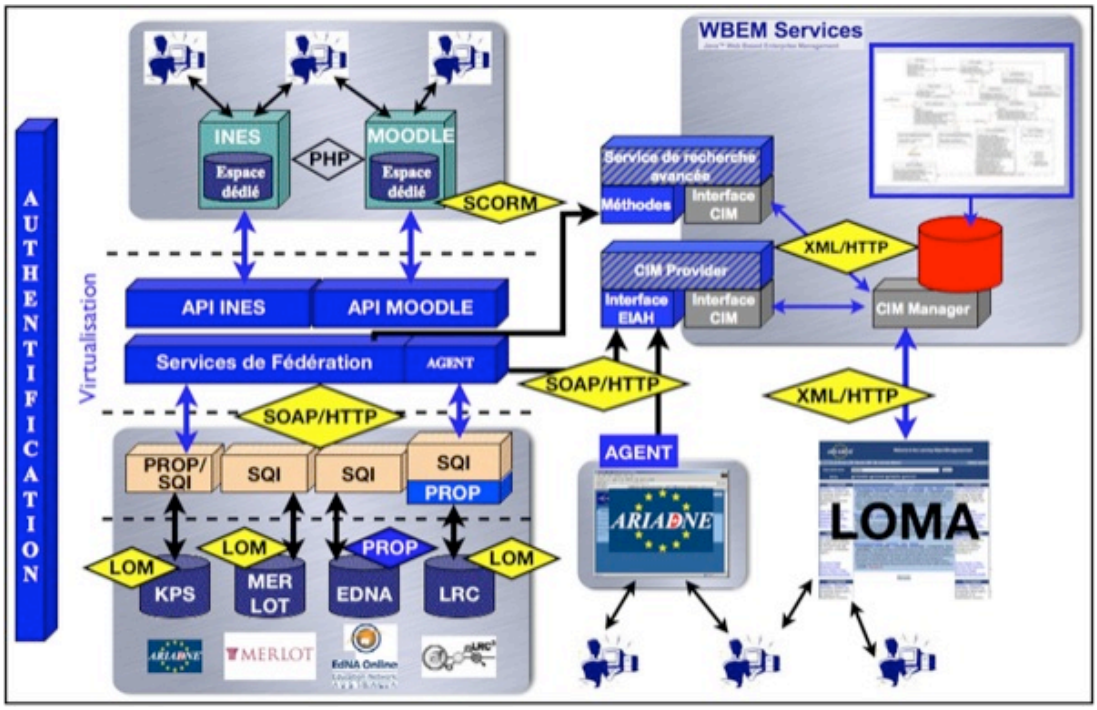

Figure 5 • Architecture de gestion des traces

La première étape a consisté à adopter un outil *open source* conforme à l'architecture WBEM et fournissant un CIM OM. Parmi les initiatives existantes, le projet WBEM Services (WBEM Services, 2003) développé par Sun

Microsystems™ consiste en un ensemble d'API et d'applications Client/Serveur Java™. Il fournit un compilateur qui permet d'introduire un fichier MOF au sein du référentiel associé au CIM OM et figurant en rouge sur la figure 5. Afin d'y intégrer notre modèle de traces d'activité, nous avons décrit chaque classe de notre modèle UML selon le format MOF avant de compiler l'ensemble de ces fichiers dans le référentiel. La figure 6 donne un extrait de la description MOF de la classe *EIAH_HasConsulted*.

Nous avons ensuite développé un CIM OP responsable de l'intégration des traces d'activité dans le CIM OM. Une interface dédiée à la réception des traces brutes capturées au sein d'un système d'apprentissage a été implémentée sous la forme d'un service *Web*. L'autre interface, celle qui communique avec le CIM OM via XML/HTTP, est constituée de six méthodes qui correspondent aux six activités recensées. Pour chacune d'entre elles, le CIM OP assure la création et/ou la modification des instances du modèle de traces impliquées dans l'activité réalisée par l'utilisateur.

```
// ==========================================================================
// EIAH_HasConsulted
// ==========================================================================
[Association,  Description (
"EIAH_HasConsulted associates an Identity with a resource and allows"
"to know the Resources consulted by a user.")]
class EIAH_HasConsulted : EIAH_ActivityOnResource {
   [Key, Description ("The Identity having consulted the Resource")]
   EIAH_Identity REF TargetIdentity;
   [Key, Description ("The Resource having been consulted")]
   EIAH_Resource REF TargetResource;
   [Required, Description("The number of consultation of the resource"
   "by the user.")]
   uint32 NbConsultation;
   [Required, MaxLen(256), Description("The date of the last"
   "consultation of the resource by the user.")]
   string[] ConsultationDates;
};
```

Figure 6 • Extrait de la représentation MOF de la classe *EIAH_HasConsulted*

Enfin, pour permettre l'acquisition de chacune des traces brutes résultant des activités des utilisateurs exploitant la couche de virtualisation, nous y avons intégré un composant logiciel. Celui-ci a également été adapté et introduit dans la version officielle de l'outil SILO, offrant ainsi la possibilité de recueillir des traces issues de ce système. Ce composant communique avec l'interface *Web* du CIM OP à travers les protocoles SOAP/HTTP. Les traces brutes capturées étant transmises lors de l'activation d'une activité spécifique chez le client, les valeurs des données produites par des langages de programmation dynamiques sont disponibles.

4.3. Visualisation des traces d'activités

WBEM Services fournit une application pour naviguer dans le référentiel CIM et consulter une à une les instances qui y sont renfermées. Cependant, cette application n'offre pas l'opportunité de consulter un ensemble d'instances qui donne un sens aux traces d'activité. Nous avons alors développé l'application baptisée LOMA (Learning Object MAnagement tool) qui offre sous la forme d'une interface graphique fondée sur les technologies *Web* une vue homogène des traces d'activité qui sont renfermées dans le référentiel CIM. Cette application émet différentes requêtes CQL au CIM OM (via les protocoles XML/HTTP) afin d'extraire du référentiel CIM les instances de classes demandées avant de les retourner à l'application LOMA qui, finalement, les met en forme avant de les présenter à l'utilisateur. L'interface graphique propose quatre sections différentes pour consulter les traces stockées dans le référentiel CIM, dont deux s'intéressent en particulier à la visualisation des traces d'activités sur les objets pédagogiques.

Pour un utilisateur spécifique, l'application permet de visualiser les traces d'usage suivantes (cf. figure 7) :

- pour chacune des activités, les objets pédagogiques qu'il a manipulés ;
- des données statistiques sur l'usage des objets pédagogiques ainsi que les dates associées à ces données ;
- Des informations sur le déploiement des objets pédagogiques en activant l'hyperlien placé sur le titre de la ressource (le(s) vivier(s) de connaissance de stockage, le(s) cursus d'apprentissage intégrant cet objet, la ou les plate(s)-forme(s) d'apprentissage déployant ces différents cursus).

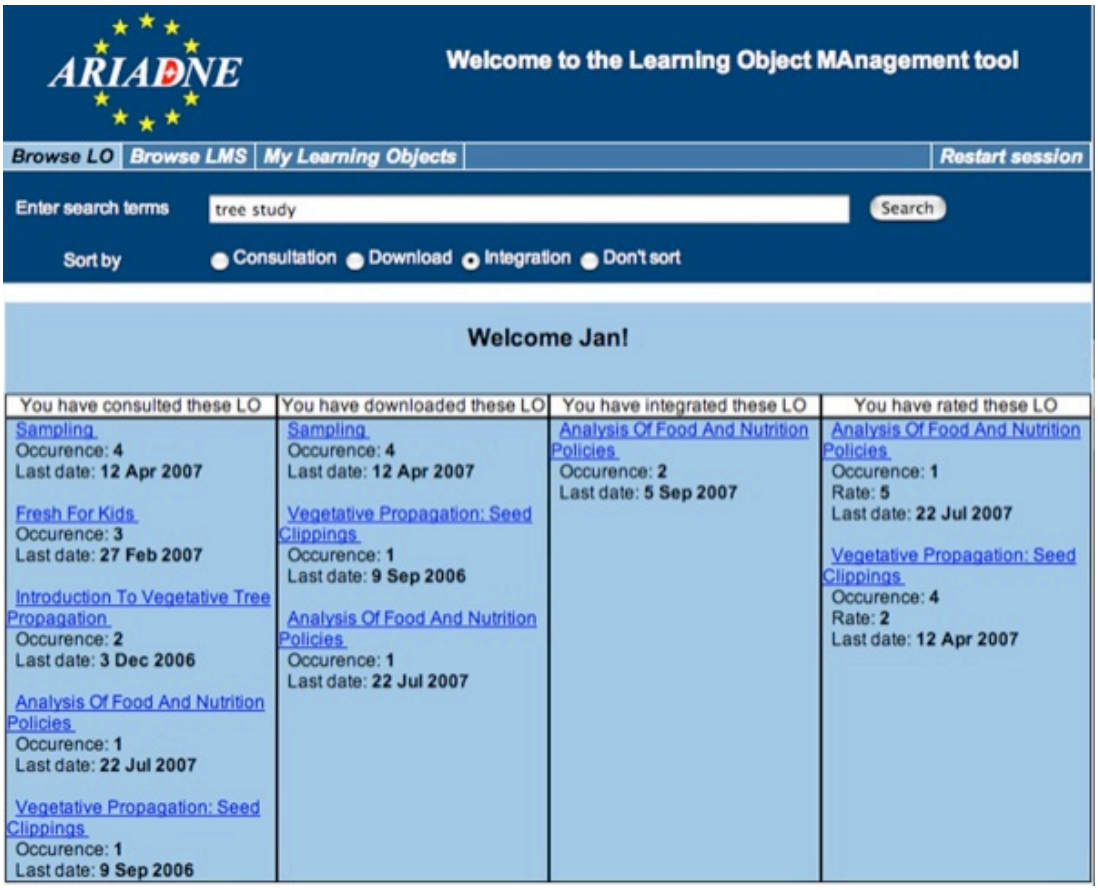

Figure 7 • L'interface de l'application LOMA

La visualisation par objets pédagogiques affiche les informations décrites ci-dessus, mais prend en compte les traces d'activité réalisées par l'ensemble des utilisateurs et non par un utilisateur spécifique.

4.4. Scénario d'activités

Nous illustrons ici un scénario simple de deux activités réalisées sur deux systèmes d'apprentissage différents. La première activité consiste en la consultation des méta-données d'un objet pédagogique traitant du langage de programmation PHP à partir de l'outil SILO. La deuxième activité réalisée sur la plate-forme INES est décomposée en deux sous-activités : l'attribution d'un niveau d'appréciation élevé à un objet pédagogique relatif lui aussi à PHP, et l'importation de cet objet dans un cursus d'apprentissage.

Le résultat de la première activité se traduit par les opérations successives suivantes (ne sont décrites ici que les opérations liées aux activités) :

- création d'une instance de la classe *CIM_Identity* pour représenter l'utilisateur à l'origine de l'activité ;
- création d'une instance *EIAH_LearningObject* qui correspond à l'objet pédagogique consulté ;
- création d'une instance de la relation d'association *EIAH_HasConsulted* qui référence les identifiants des deux instances ci-dessus.

La deuxième activité engendre, en plus des deux premières actions mentionnées ci-dessus, la création :

- d'une instance de la relation *EIAH_HasRated* qui référence les instances de l'utilisateur et de l'objet pédagogique correspondants ;
- d'une instance de la relation d'association *EIAH_HasIntegrated* qui référence également les mêmes utilisateur et objet pédagogique.

La partie suivante montre comment les traces d'usage produites par ces deux activités peuvent être exploitées pour faciliter le processus de recherche d'objets pédagogiques lors de la phase de création d'un cursus par un enseignant.

4.5. Exploitation des traces : un service de recherche avancée d'objets pédagogiques

Lors de la création d'un cursus, un enseignant recherche souvent du matériel pédagogique existant afin de l'adapter ou de le réutiliser dans son contexte d'apprentissage. Cependant, de nombreuses ressources sont aujourd'hui

disponibles dans différents viviers de connaissance, et un utilisateur doit parcourir un grand nombre d'objets d'apprentissage avant de trouver une ressource correspondant à ses objectifs. Pour lever ce verrou, nous avons proposé un service de recherche avancée d'objets pédagogiques qui exploite les traces d'activité renfermées dans le référentiel CIM (Broisin et Vidal, 2007). Ce service permet de présélectionner les ressources les plus pertinentes en fonction de certaines traces d'usage : les informations statistiques d'utilisation, le cursus pédagogique visé, et les évaluations associées aux objets pédagogiques. Le service de recherche avancée présente, de la même manière que le CIM OP, deux interfaces distinctes : l'une capable de recevoir des requêtes issues de différents systèmes, l'autre interagissant avec le CIM OM pour consulter les traces d'usage.

Continuons le scénario initié dans la section précédente, et imaginons qu'un enseignant recherche, à partir de MOODLE, un objet pédagogique pour un cursus d'apprentissage traitant du langage de programmation PHP. Le scénario mis en œuvre lors de l'activation du service de recherche de la couche de virtualisation se décompose en 5 étapes.

- Le service de recherche de la couche de virtualisation envoie une requête vers le service de recherche avancée en transmettant les mots clés mentionnés par l'utilisateur.

- Le service de recherche avancée traite la requête en recherchant, dans le CIM OM, les instances d'objets pédagogiques correspondant aux mots clés et dont les traces d'usage sont les plus pertinentes. Dans notre scénario, la ressource pédagogique ayant déjà fait l'objet d'une évaluation positive et d'intégration dans un cursus sera classée avant celle n'ayant été que consultée.

- Les résultats retrouvés et triés par le service de recherche avancée sont ensuite retournés au service de recherche de la couche de virtualisation.

- Ce dernier établit la correspondance entre les objets pédagogiques retrouvés dans le vivier de connaissance et ceux retournés par le service de recherche avancée.

- Les résultats sont formatés puis présentés à l'enseignant, les objets pédagogiques retrouvés par le service de recherche avancée figurant en tête de la liste des résultats.

L'enseignant a donc pu bénéficier de traces d'activités qui ont été produites par d'autres utilisateurs exploitant d'autres systèmes d'apprentissage que celui utilisé dans son contexte, ce qui a facilité l'activité de recherche d'objets pédagogiques. En effet l'enseignant n'est pas contraint de parcourir de nombreuses ressources avant de trouver celle(s) correspondant à ses besoins.

5. Conclusions et perspectives de recherche

Nous avons proposé une approche conduite par les modèles capable d'assurer la gestion de traces issues d'environnements d'apprentissage hétérogènes et qui facilite leur partage en vue d'une réutilisation par différents SBT. Le modèle UML générique que nous avons établi se focalise sur les traces produites par les activités explicites des utilisateurs, mais le caractère générique et extensible du métamodèle CIM sur lequel nous nous sommes appuyés montre que d'autres sources de traces peuvent facilement être prises en compte dans notre approche. Les protocoles et modes de transport retenus dans l'architecture distribuée et décentralisée mise en œuvre pour la gestion des traces s'appuient sur les technologies des services *Web*, et assurent ainsi l'interopérabilité et la communication entre systèmes à observer et composants d'observation sur une grande échelle. Cette architecture facilite la collecte et le partage de traces issues de diverses communautés d'utilisateurs, et participe ainsi à l'augmentation de la masse de traces disponibles pour atteindre les objectifs visés par les SBT.

L'approche proposée a été appliquée avec succès dans le cadre de l'observation des activités relatives aux objets pédagogiques intégrés dans des plates-formes de formation et viviers de connaissance, et facilite le processus de recherche de ressources pédagogiques en enrichissant les outils traditionnels associés aux viviers de connaissance. Pour des raisons de simplification, l'architecture déployée dans ce cas d'usage n'est constituée que d'un seul référentiel responsable du stockage de l'ensemble des traces issues des systèmes précités. Toutefois, dans le cadre de la nouvelle architecture orientée services actuellement élaborée au sein de la fondation ARIADNE, un référentiel de traces sera associé à chaque vivier local. Ainsi, les différentes institutions pourront choisir de bénéficier des traces produites par leur structure pédagogique, ou par l'ensemble de la communauté ARIADNE. Dans le premier cas, le référentiel ne retournera que ses propres données, alors qu'il enverra des requêtes vers tous les autres référentiels dans le deuxième cas.

Une perspective de ces travaux consiste à mettre en oeuvre d'autres fonctionnalités qui exploitent les concepts du métamodèle CIM. Ce dernier définit des associations entre objets gérés qui expriment des relations de dépendance reliant au moins deux classes d'objets : l'une identifiée comme *Antécédent*, l'autre comme *Dépendant*. Par exemple, une relation de dépendance est intrinsèquement définie entre un système (l'*Antécédent*) et un service (le *Dépendant*) hébergé par ce système. De plus, des travaux (Sibilla et al., 2004) ont proposé la possibilité d'associer des actions à effectuer sur l'objet *Dépendant* en fonction d'événements survenant sur l'objet *Antécédent*. En spécifiant des relations de dépendance entre certaines classes de notre modèle, nous serons en mesure d'automatiser des actions à exécuter en fonction d'événements se produisant sur des instances du modèle. Lorsqu'une nouvelle ressource pédagogique correspondant aux concepts préférés de certains utilisateurs est créée au sein du référentiel de traces, un courrier électronique pourra par exemple les avertir de la disponibilité d'une ressource supplémentaire. La

sémantique forte du métamodèle CIM et les travaux qui visent à ajouter de la dynamique permettraient d'aller plus loin dans le processus d'exploitation et de réutilisation des traces d'usage.

BIBLIOGRAPHIE

Fondation ARIADNE (1996). http://www.ariadne-eu.org, (consulté le 6 mai 2007).

BROISIN J., VIDAL P. (2007). Finding Learning Objects: an Advanced Search Service based on Tracking Data. *E-Learn 2007*, Québec ville, Canada.

BROISIN J. (2006). *Un Environnement Informatique pour l'Apprentissage Humain au service de la Virtualisation et de la Gestion des Objets Pédagogiques*. Thèse de doctorat de l'Université Paul Sabatier Toulouse III, 2006, 206 p. Disponible sur internet : http://www.irit.fr/~Julien.Broisin/

BROISIN J., VIDAL P. (2005). Un Environnement Informatique pour l'Apprentissage Humain au service de la Virtualisation des Objets Pédagogiques. *Revue Sciences et Techniques de l'Information et de la Communication pour l'Éducation et la Formation*, 2005, vol. 12. Disponible sur internet : http://sticef.univ-lemans.fr/num/vol2005/broisin-08/sticef_2005_broisin_08p.pdf

CARRON T., MARTY J. C., HERAUD J. M., FRANCE L. (2006). Helping the teacher to re-organize tasks in a collaborative learning activity: an agent-based approach. *The 6th IEEE International Conference on Advanced Learning Technologies*, Kerkrade, Pays-Bas, p 552-554.

COURTIN T., TALBOT S. (2007). Une station d'observation pour des Situations d'Apprentissage Collaboratif Instrumenté. *Environnements Informatique pour l'Apprentissage Humain*, Lausanne, Suisse, p. 371-376.

Distributed Management Task Force (1999). *Common Information Model (CIM) Specification*. Rapport technique, DSP0004, 1999, 96 p. Disponible sur internet : http://www.dmtf.org/standards/documents/CIM/DSP0004.pdf (consulté le 30 avril 2007).

Distributed Management Task Force (2000). *Common Information Model (CIM) Core Model*. Rapport technique, DSP0111, 2000, 54 p. Disponible sur internet : http://www.dmtf.org/standards/documents/CIM/DSP0111.pdf (consulté le 30 avril 2007).

Distributed Management Task Force (2003). *Understanding the Application Management Model*. Rapport technique, DSP0140, 2003, 50 p. Disponible sur internet : http://www.dmtf.org/standards/documents/CIM/DSP0140.pdf (consulté le 30 avril 2007).

Distributed Management Task Force (2003). *CIM System Model White Paper*. Rapport technique, DSP0150, 2003, 22 p. Disponible sur internet : http://www.dmtf.org/standards/documents/CIM/DSP0150.pdf (consulté le 30 avril 2007).

Distributed Management Task Force (2003). *CIM Network Model White Paper*. Rapport technique, DSP0152, 2003, 19 p. Disponible sur internet : http://www.dmtf.org/standards/published_documents/DSP0152.pdf (consulté le 30 avril 2007).

Distributed Management Task Force (2003). *CIM User and Security Model White Paper*. Rapport technique, DSP0139, 2003, 11 p. Disponible sur internet : http://www.dmtf.org/standards/documents/CIM/DSP0139.pdf (consulté le 30 avril 2007).

Distributed Management Task Force (2003). *CIM Event Model White Paper*. Rapport technique, DSP0107, 2003, 28 p. Disponible sur internet : http://www.dmtf.org/standards/documents/CIM/DSP0107.pdf (consulté le 9novembre 2007).

Distributed Management Task Force (2006). *CIM Query Language Specification*. Rapport technique, DSP0202, 2006, 68 p. Disponible sur internet : http://www.dmtf.org/standards/published_documents/DSP0202.pdf (consulté le 9 novembre 2007).

Design Patterns for recording and analysing Usage of Learning Systems (2005). Réseau d'Excellence Kaleidoscope, http://www.noe-kaleidoscope.org/pub/researcher/activities/wp_desc/32.html (consulté le 19 avril 2007).

eLycée. http://www.elycee.com, (consulté le 25 avril 2007).

FANSLER K., RIEGEL R. (2004). A Model Of Online Instructional Design Analytics. *20th Annual Conference on Distance Teaching and Learning,* Madison, Etats-Unis, Disponible sur internet : http://www.uwex.edu/disted/conference/Resource_library/proceedings/04_1069.pdf.

HERAUD J.M., FRANCE L., MILLE A. (2004). Pixed: An ITS that guides students with the help of learners' interaction logs. *Intelligent Tutoring System,* Maceio, Brésil, Disponible sur internet : http://www.andrew.cmu.edu/user/jb8n/its2004/heraud.pdf

HERAUD J.M., MARTY J. C., FRANCE L., CARRON T. (2005). Helping the interpretation of web logs : Application to Learning Scenario Improvement. *Workshop Usage Analysis in Learning Systems, 12th International Conference on Artificial Intelligence in Education,* Amsterdam, Pays-Bas.

Interaction & Collaboration Analysis' supporting Teachers & students' Self-Regualtion (2004). Réseau d'Excellence Kaleidoscope, http://www.rhodes.aegean.gr/ltee/kaleidoscope-icalts/ (consulté le 19 avril 2007).

IKSAL S., CHOQUET C. (2005). Usage Analysis Driven by Models in a Pedagogical Context. *Workshop on Usage Analysis in Learning Systems , 12th International Conference on Artificial Intelligence in Education*, Amsterdam, Pays-Bas, p. 49-56.

IKSAL S., CHOQUET C. (2005). An open architecture for usage analysis in a e-learning context. *The 5th IEEE International Conference on Advanced Learning Technologies*, Kaohsiung, Taïwan, p. 177-181.

KEPKA L., HERAUD J. M., FRANCE L., MARTY J. C., CARRON T. (2007). Activity Visualization and Regulation in a Virtual Classroom. *The 10th IASTED International Conference on Computers and Advanced Technology in Education*, Beijing, Chine.

LAFLAQUIERE J., SETTOUTI L. S., PRIE Y., MILLE A. (2006). Trace-Based Framework for Experience Management and Engineering. *10th International Conference on Knowledge-Based Intelligent Information and Engineering Systems*, Bournemouth, UK, Lecture Notes in Computer Science 4251 Springer 2006.

LAFLAQUIERE J., PRIE Y. (2003). Modélisation d'utilisation de système pour une assistance à base de trace : une application de MUSETTE à la tâche de veille documentaire. *Traces, interactions, co-constructions collectives et relations à la cognition, AS CoMETE,* Paris, France.

LECLET D., LEPRETRE E., PETER Y., QUENU-JOIRON C., TALON B., VANTROYS T. (2007). Améliorer un dispositif pédagogique par l'intégration de nouveaux canaux de communication. *Environnements Informatique pour l'Apprentissage Humain*, Lausanne, Suisse, p. 347-358.

LOGHIN J. C. (2006). Aide à la compréhension du comportement de l'utilisateur par la transformation des traces collectées. *1res Rencontres Jeunes Chercheurs sur les Environnements Informatiques pour l'Apprentissage Humain*, Evry, France, p. 115-122.

MARTIN FLATIN, J.P. (2003). *Web-based management of IP networks and systems*. Edition Wiley series in Communications Networking & Distributed Systems, 2003.

MICHEL C., PRIE Y., LE GRAET L. (2005). Construction d'une base de connaissance pour l'évaluation de l'usage d'un environnement STIC. *17eme Conférence Internationale Francophone sur l'Interaction Homme-Machine*, Toulouse, France, p. 199-202.

NAJJAR J., TERNIER S., DUVAL E. (2004). User Behavior in Learning Object Repositories: An Empirical Analysis. *Educational Multimedia, Hypermedia \& Telecommunications - EDMEDIA04*, Lugano, Suisse, p. 4373-4379.

NIELSEN H., LEACH P., LAWRENCE S. (2000). *An HTTP Extension Framework.* Internet Engineering Task Force, RFC 2774, 2000, 18 p. Disponible sur internet : http://www.ietf.org/rfc/rfc2774.txt (consulté le 30 avril 2007).

SIBILLA M., BARROS DE SALES A., BROISIN J., VIDAL P., JOCTEUR-MONROZIER J.F. (2004). Behaviour modelling : a contribution to CIM. *DMTF Academic Alliance Paper Competition*, 7p. Disponible sur internet : http://www.dmtf.org/education/academicalliance/sibilla_2004.pdf (consulté le 13 avril 2007).

STEINMETZ R. (2005). Learning Objects, Learning Object Repositories and Beyong. *Intelligent, Interactive, Learning Object Repositories 2005*, Vancouver, Canada.

STERMSEK G., STREMBECK M., NEUMANN G. (2007). A User Profile Derivation Approach based on Log-File Analysis. I*nternational Conference on Information and Knowledge Engineering*, Las Vegas, Etas-Unis.

SWAAK J., EFIMOVA L., KEMPSEN M., GRANER M. (2004). Finding in-house knowledge: patterns and implications. *The fourth International Conference on Knowledge Management, I-KNOW 04*, Gras, Autriche p. 27-36.

TARBY J. C., EZZEDINE H., ROUILLARD J., TRAN C. D., LAPORTE P., KOLSKI C. (2007). Traces Using Aspect Oriented Programming and Interactive Agent-Based Architecture for Early Usability Evaluation: Basic Principles and Comparison. *Human Computer Interaction, Part I, HCII 2007, LNCS 4550*, Beijing, Chine p. 632-641.

TCHOUNIKINE P. (2002). Pour une ingénierie des Environnements Informatiques pour l'Apprentissage Humain. *Revue I3 information - interaction – intelligence*, Vol. 2 n° 1, p. 59-95.

Web Based Enterprise Management. (1999). http://www.dmtf.org/standards/wbem/, (consulté le 30 avril 2007).

WBEM Services. (2003). http://wbemservices.sourceforge.net/, (consulté le 30 avril 2007).

A propos des auteurs

Julien Broisin est Maître de Conférences à l'Université Paul Sabatier de Toulouse. Ses travaux de recherche se préoccupent des problématiques liées à la capitalisation des expériences tracées en vue d'une conception dynamique d'environnements d'apprentissage personnalisés et adaptés à un contexte particulier.

Adresse : Institut de Recherche en Informatique de Toulouse, 118 route de Narbonne Bâtiment 1R1 31062 Toulouse Cedex 4

Courriel : broisin@irit.fr

Toile : http://www.irit.fr/~Julien.Broisin

Philippe Vidal est Professeur à l'Université Paul Sabatier de Toulouse. Il effectue son activité de recherche au sein de l'équipe SIERA du laboratoire IRIT. Sa thématique de recherche concerne la distribution et la supervision des objets pédagogiques à grande échelle dans l'objectif de partage et réutilisation, et la construction d'environnements d'aide à la

conception et à l'exploitation d'outils d'enseignement à distance.

Adresse : Institut de Recherche en Informatique de Toulouse, 118 route de Narbonne Bâtiment 1R1 31062 Toulouse Cedex 4

Courriel : vidal@irit.fr

Toile : http://www.irit.fr/~Philippe.Vidal

Référence de l´article :

Julien BROISIN, Philippe VIDAL, Une approche conduite par les modèles pour le traçage des activités des utilisateurs dans des EIAH hétérogènes, *Revue STICEF*, Volume 14, 2007, ISSN : 1764-7223, mis en ligne le 13/03/2008, http://sticef.org