Un système de workflows flexible pour la formation ouverte et à distance

Thomas Vantroys, Yvan Peter

Un système de workflows flexible pour la formation ouverte et à distance

Thomas Vantroys[*,†], Yvan Peter[*]

[*]Université des Sciences et Technologies de Lille
Laboratoire TRIGONE – Équipe NOCE
Cité scientifique, Bât B6
59655 Villeneuve d'Ascq - France
{thomas.vantroys,yvan.peter}@univ-lille1.fr

[†]Archimed SA
49 Boulevard de Strasbourg
59042 Lille - France
t.vantroys@archimed.fr

Abstract

Open and Distance Learning platforms require mechanisms for the enactment and coordination of pedagogical modules and learning activities. For that purpose, they can use workflow management systems. These systems formerly reserved for highly structured procedures can be used in dynamic and reactive environments such as virtual campuses platforms, thanks to a enhanced flexibility in the execution of models and in the management of exceptions. In this article we shall present COW our flexible workflow engine dedicated to open and distant learning. We shall see how it is possible to organize the pedagogical modules and the learning paths to answer the expectation within the framework of individual courses (lifelong learning orientation) or within the framework of group courses (closer to the traditional face to face learning).

Résumé

Les plates-formes de formation ouverte et à distance nécessitent des mécanismes d'animation et de coordination des modules et des activités pédagogiques. Pour cela, elles peuvent s'appuyer sur les systèmes de Workflows. Ces systèmes autrefois réservés aux procédures fortement structurées peuvent, grâce à une plus grande flexibilité dans l'exécution des modèles et à la prise en compte des exceptions, être utilisés dans des environnements hautement dynamiques et réactifs comme les plates-formes de campus virtuel. Dans cet article, nous présentons COW, notre moteur de workflows flexible adapté à l'éducation. Nous verrons comment il est possible d'organiser les modules d'enseignement et les parcours de formation pour répondre aux attentes dans le cadre de parcours individuels (orientation formation continue) et dans le cadre de parcours de groupe (plus proche du modèle classique de formation présentielle transposé vers une formation à distance).

1 Introduction

Fournir la bonne activité, avec les bons documents, à la bonne personne et au bon moment est la fonction fondamentale des systèmes informatique administratifs fortement structurés. Ce principe est également vrai dans les plates-formes d'enseignement à distance où l'un des objectifs est de procurer à l'étudiant un travail à effectuer avec des documents et/ou des outils en fonction de ses activités antérieures. Alors que des systèmes de workflows ont été utilisés pour gérer les travaux administratifs, ces mêmes systèmes se retrouvent de façon très rare dans les plates-formes d'éducation. Les lacunes originelles des systèmes de workflows comme le manque de flexibilité dans l'exécution des modèles ou la mauvaise gestion des exceptions disparaissent peu à peu, permettant ainsi leur utilisation dans des environnements hautement dynamiques et réactifs. Pour ces raisons, nous avons entamé des recherches pour la réalisation d'un moteur de workflow flexible adapté à la formation ouverte et à distance.

Nous allons présenter notre prototype et montrer les apports d'un système de workflows dans une plateforme de campus virtuel. Nous verrons comment il est possible d'organiser les modules d'enseignement et les parcours de formation pour répondre aux attentes dans le cadre de parcours individuel (orientation formation continue) et dans le cadre de parcours de groupe (plus proche du modèle classique de formation présentielle transposé vers une formation à

distance).

La suite de cet article se décompose comme suit. Nous allons dans un premier temps introduire les standards et les principes de base des systèmes de workflows, puis nous illustrerons leurs utilisations dans deux plates-formes d'enseignement. Dans la partie suivante, nous présenterons le système que nous avons conçu. Nous exposerons dans la section 4 la modélisation des parcours et des modules de formation ainsi que leurs mises en œuvre dans notre système. Enfin, la section 5 conclura cet article en résumant notre proposition et en présentant quelques pistes d'évolutions que nous allons parcourir dans le futur.

2 Systèmes de workflows flexibles

Dans cette section, nous présenterons les standards et les concepts des systèmes de workflows flexibles puis, nous verrons deux cas d'utilisation dans des systèmes d'éducation à distance.

2.1 Standards

Pour permettre l'émergence de standards dans le monde des systèmes de workflows, des éditeurs de logiciels, des laboratoires de recherches et des utilisateurs de ces systèmes ont créé le consortium *Workflow Management Coalition* (WfMC) [4]. L'objectif de cette association est la promotion et le développement des systèmes de workflows. Pour cela, ils ont réalisé un glossaire [6] afin d'unifier la terminologie et ils ont défini un modèle de référence (figure 1) [7] centré autour du moteur d'exécution. Ce modèle présente 5 interfaces de standardisations :

Interface 1 : Elle correspond à l'échange des modèles entre moteurs de workflows et les différents outils de modélisation de processus ;

Interface 2 : Elle permet à des applications clientes de communiquer avec le moteur de workflows ;

Interface 3 : Elle permet au système de workflows d'appeler des applications clientes ;

Interface 4 : Elle permet l'interopérabilité entre moteurs de workflows ;

Interface 5 : Elle correspond à l'interaction entre les applications d'administration et le moteur de workflows.

2.2 Terminologie

Avant d'aller plus loin, nous allons définir la terminologie workflow que nous utilisons. Elle se base sur le glossaire du WfMC [6]. Un *processus de workflow* est défini comme une suite ordonnée d'*activités* dont le but est la résolution d'un problème ou la réalisation d'un objectif. Chaque activité représente une étape logique dans la résolution. Elles sont ordonnées par des *transitions*. Un *workitem* est la représentation d'un travail à réaliser par une *ressource*. Une ressource représente aussi bien un humain qu'un logiciel. Un système de workflows gère les instances de processus. Il va les créer en fonctions des modèles existants, il les gère en ordonnançant les différentes activités et en assignant les tâches à effectuer aux ressources. Dans notre approche, nous classifions les ressources dans deux catégories ; les utilisateurs (humain ou rôle associé) et les documents électroniques (tout ce qui est manipulé par les utilisateurs).

2.3 Flexibilité

La flexibilité dans le cadre d'un système de workflows se trouve à plusieurs niveaux. Ceux qui nous intéressent sont l'adaptabilité des modèles et la gestion des exceptions.

L'adaptabilité représente la capacité de modifier le modèle du processus en cours d'exécution afin de l'adapter à des conditions changeantes. Le système doit offrir la capacité d'ajouter ou de supprimer des activités, de redéfinir l'ordonnancement entre les activités existantes et de modifier les différents attributs des processus, activités et workitems.

Dans le cadre de l'éducation à distance, le système doit par exemple permettre à l'enseignant d'ajouter des étapes supplémentaires pour un élève en difficulté. Il doit également permettre de réassigner les tâches d'un enseignant indisponible (vacances, maladie) à un autre.

Le deuxième élément important de la flexibilité est la gestion des exceptions. Le système doit réagir face aux situations imprévues. Une classification des différents types d'exception peut être trouvée dans [9].

Les cas les plus courants auxquels nous sommes confrontés concernent des étudiants qui envoient un mauvais type de document ou qui ne réalisent pas leur travail dans le temps alloué. Certaines solutions à ces problèmes peuvent être automatiques ou demander une intervention humaine. La solution la plus simple consiste à arrêter le processus et avertir le responsable du workflow ou du processus pour qu'il rétablisse le bon fonctionnement et relance le processus.

2.4 Systèmes de workflow et formation ouverte à distance

À notre connaissance il existe assez peu de plates-formes d'éducation à distance utilisant des moteurs de workflows. Pour illustrer, nous en verrons deux qui visent des publics différents. La première, issue de travaux de recherches du DSTC, est utilisée dans une université australienne. La deuxième plate-forme, qui est industrielle, sert de cadre à nos travaux.

2.4.1 Flex-eL

Le système Flex-eL [10] (Flexible e-learning) est conçu par le Distributed Systems Technology Center (DSTC) [11] et par l'université du Queensland en Australie. Flex-eL est une plate-forme d'éducation à distance basée sur un moteur de workflows flexible.

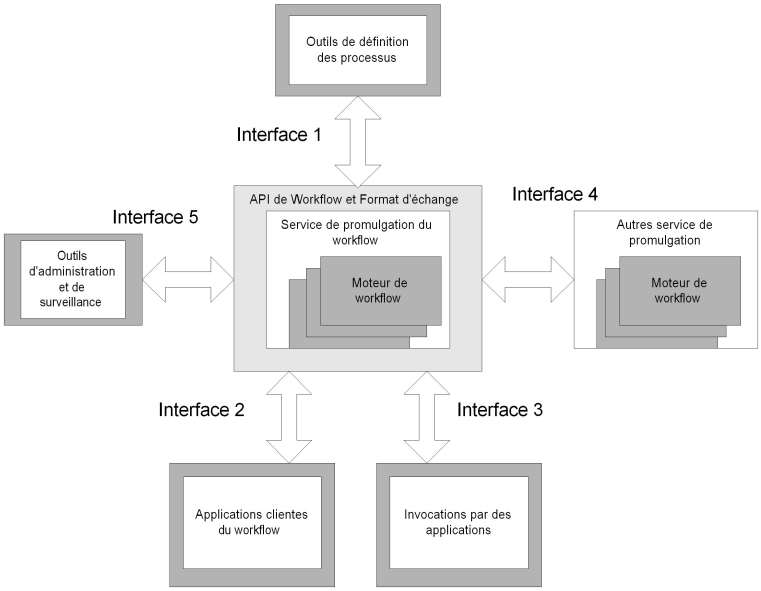

FIG. 1 – *Modèle de référence des systèmes de workflow ([7])*

Les créateurs sont partis du principe que les systèmes d'enseignement à distance existants se focalisaient plus sur les technologies que sur le processus d'apprentissage. L'objectif n'est pas de recréer ce que l'on trouve de manière traditionnelle dans l'enseignement mais d'inventer un nouvel environnement d'apprentissage.

Ce projet, débuté en mars 2000, est actuellement testé à l'université du Queensland dans le cadre d'un cours du Master of Information Technology. Les premiers résultats semblent positifs notamment du point de vue du taux de désaffection.

De part sa philosphie, Flex-eL s'adresse plus particulièrement au domaine de la formation tout au long de la vie, sans véritables contraintes temporelles. L'étudiant construit son parcours au fur et à mesure.

Cette plate-forme répond bien au contexte de la formation continue. Néanmoins, elle ne répond pas complètement à notre problématique aussi bien sur le contexte d'utilisation que sur l'approche technologique. Nous voulons un système utilisable à la fois dans un contexte de formation continue et dans un contexte de formation traditionnelle. Nous souhaitons spécifier de véritables tâches collaboratives où les étudiants travaillent ensemble.

2.4.2 Campus Virtuel

La plate-forme " Campus Virtuel "^{TM} [1] de la société Archimed propose également l'utilisation d'un système de workflows pour modéliser les différentes étapes d'un module d'enseignement (figure 2).

Un enseignant peut ainsi décomposer son cours en différentes étapes, en indiquant à chaque fois les

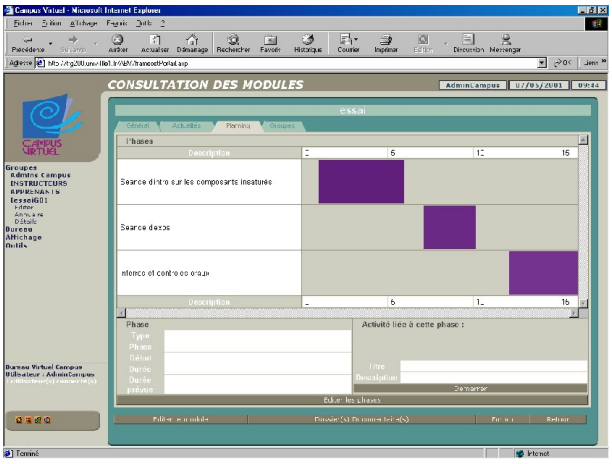

FIG. 2 – *Exemple de modélisation dans le " Campus Virtuel*^{TM} *"*

documents utilisés. Le système, qui s'appuie sur un système de gestion de documents multimédias, assure le routage des documents entre les étapes. L'inconvénient du système actuel est qu'il ne permet pas un véritable travail individuel. Lorsque l'enseignant décrit une activité, il indique le travail que les étudiants doivent réaliser. Pour passer à l'étape suivante, tous les étudiants doivent avoir terminé leurs tâches, ce qui empêche un étudiant d'avancer à son rythme.

Notre travail de recherche, en partenariat avec la société Archimed, a pour objectif de réaliser un système de workflow flexible adapté au contexte de la formation ouverte à distance. Ce système sera en-

suite intégré dans la plate-forme " Campus Virtuel^{TM} " pour la gestion des activités pédagogiques.

Nous allons maintenant présenter notre plate-forme.

3 La plate-forme COW

L'objectif de la plate-forme COW, pour *Co-operative Open Workflow*, est d'offrir un support d'exécution pour des workflows flexibles adaptés à l'éducation à distance. Nous voulons gérer un groupe d'étudiants effectuant un module de formation tout en permettant une possible personnalisation du cheminement d'un étudiant au sein de ce module. L'autre aspect important pour nous est la collaboration entre étudiants et avec l'enseignant.

3.1 Modélisation et implémentation ouverte

Pour exprimer les modèles de processus et d'activités utilisés par le moteur de workflow, nous avons défini notre langage nommé *COWL*, pour COW Language. Ce langage est dérivé du XPDL [5], *XML Process Definition Language*, le langage standard de description de processus proposé par le WfMC dans le cadre de l'interface 1.

Notre langage permet d'exprimer les modèles en fonction de notre méta-modèle. Celui-ci (figure 3) est basé sur celui du WfMC mais nous avons redéfini le terme de workitem. Pour nous il est une représentation d'un travail atomique et nous pouvons l'exprimer de manière explicite dans le modèle, alors que cette notion n'apparaît que dans les instances du modèle présenté par le WfMC.

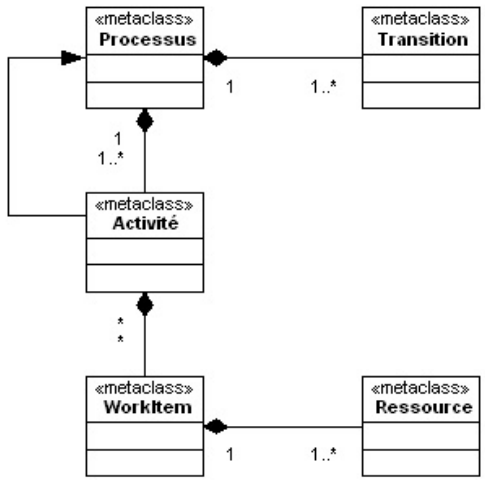

FIG. 3 – *Méta-modèle de COW*

Notre modèle de workflow est constitué de trois niveaux. Le `processus` représente un ensemble d'`activités` coordonnées par des `transitions`. Une activité peut référencer un autre processus (notion de sous-processus) ou comporter des `worktitems` qui représentent l'unité atomique de travail à réaliser. Les transitions peuvent être de deux types. Le premier représente le *flot de contrôle*, i.e., l'ordonnancement de l'exécution des activités. Le deuxième type de transition représente le *flot de données*, i.e. la transmission des données entre les activités.

Pour obtenir un système flexible, nous avons implémenté notre méta-modèle en y intégrant des mécanismes de réflexivité. Cela offre aux utilisateurs la possibilité de connaître le modèle du processus en cours (introspection) et de le modifier (intercession) [8]. Pour une plus grande souplesse, nous instancions les composants au fur et à mesure de l'avancement du processus (liaison tardive).

3.2 Implémentation

Pour réaliser notre implémentation, nous souhaitions respecter les différents standards industriels existant. Nous nous sommes donc basés sur le modèle de référence du WfMC [7] et sur la spécification *Workflow Management Facility* (WMF) [2] de l'Object Managment Group (OMG) qui propose une représentation objet d'un moteur de workflow.

Pour réaliser nos développements, nous nous sommes tournés vers une architecture à composants logiciels qui nous offre des facilités de mise en œuvre.

3.2.1 Le modèle de composant EJB

Les *Enterprise JavaBeans* (EJB) [3] de SUN Microsystems sont des composants logiciels serveurs basés sur le langage de programmation Java. La spécification de ces composants sépare la logique métier (ce que doit faire le composant) des propriétés fonctionnelles (sécurité, transaction, ...). Une représentation schématique d'un serveur EJB est visible à la figure 4. Un composant EJB est constitué de deux interfaces et d'une implémentation : l'interface *Home* est utilisée par les clients pour gérer le cycle de vie du composant (e.g. création, destruction, ...) ; l'interface *Remote* représente l'interface métier du composant ; ces deux interfaces sont implémentées par le Bean.

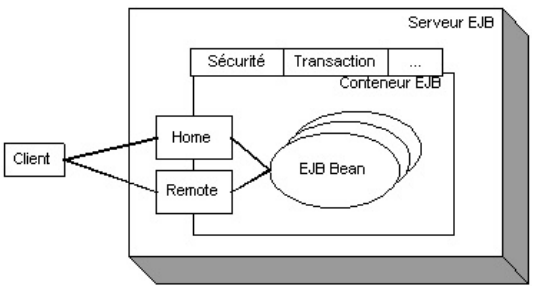

FIG. 4 – *Représentation schématique d'un serveur EJB*

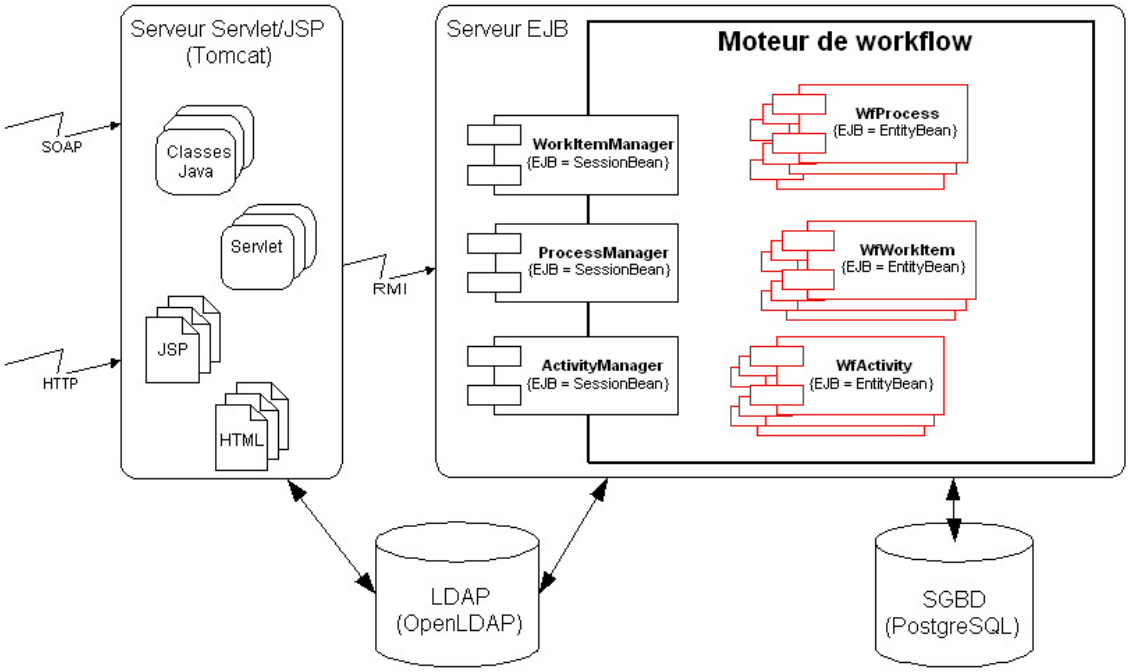

FIG. 5 – *Architecture du moteur de workflows*

Les EJBs sont déployés dans un conteneur qui offre aux composants un ensemble de fonctions systèmes (transaction, ...) libérant ainsi le programmeur qui peut se concentrer sur le code métier. Le conteneur est un espace d'exécution au sein d'un serveur EJB. La dernière spécification définit trois types de composants : *entity bean*, *session bean* et *message driven beans* :

Entity Beans : Ils correspondent à des données persistantes dans le système d'information. La persistance peut être gérée par le conteneur ou par le bean lui-même.

Session Beans : Ils sont dédiés à la gestion des interactions avec les clients. Ils ne correspondent pas à des données du système d'information mais peuvent contenir des données de session.

Message Driven Beans : Ils ont été introduits dans la dernière spécification pour permettre la communication asynchrone et anonyme entre composants basés sur le service d'événement *Java Messaging Service* (JMS).

3.2.2 Réalisation

La figure 5 représente de manière schématique l'architecture globale de notre système.

Pour notre réalisation, nous avons choisi une démarche reposant sur une architecture à micronoyau [15]. Notre noyau, composé essentiellement d'EJB de type entité, offre les fonctionnalités de bas niveau d'un moteur de workflows (ordonnancement, gestion des activités, ...). Le niveau supérieur offre des services de plus haut niveau pour les outils externes (gestionnaire de modèles, ...). Cette couche cache une partie de la complexité du noyau et permet d'ajouter facilement de nouveaux services au moteur (gestion de l'historique, ...). La couche de niveau supérieur permet de présenter notre système en tant que service web reposant sur les technologies SOAP [13] et WSDL [14]. Elle permet l'intégration de notre moteur dans des plates-formes hétérogènes et plus particulièrement avec le Campus Virtuel d'Archimed. La figure 6 représente les différents niveaux logiques de COW.

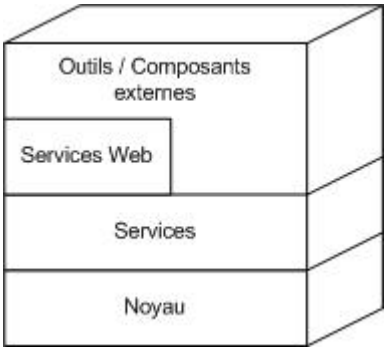

FIG. 6 – *Niveaux d'abstraction de COW*

Pour gérer les instances de processus et pour les modifier dynamiquement, nous disposons d'un outil

graphique (figure 7). Il nous permet notamment de modifier graphiquement les modèles des instances de processus.

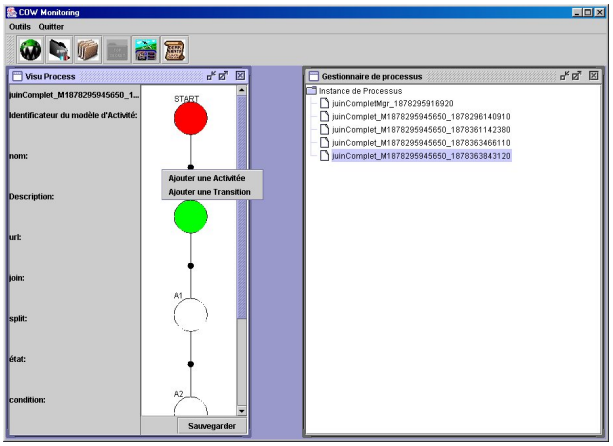

FIG. 7 – *Outil de gestion des instances de processus.*

4 Modélisation des parcours et modules de formation

4.1 Module d'enseignement

La fonction de base de notre système de workflow est l'ordonnancement des différentes étapes d'un module d'enseignement. Ce module étant effectué par un groupe d'étudiants (de 1 à n).

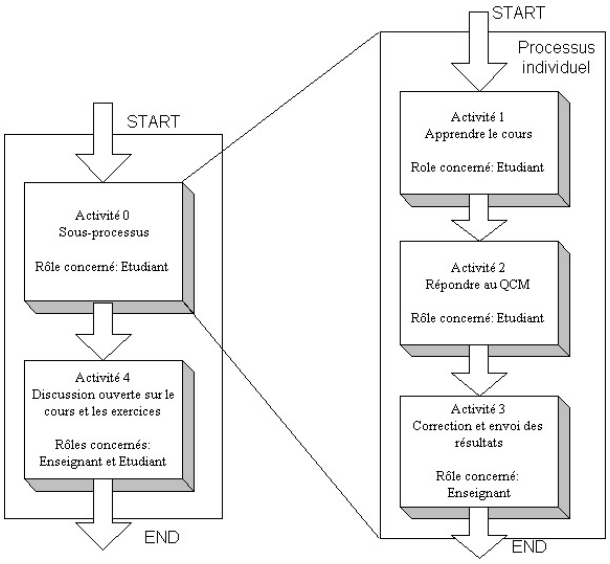

FIG. 8 – *Modélisation d'un module d'enseignement*

4.1.1 Description de l'exemple

Pour illustrer les principes de modélisation, nous allons prendre l'exemple d'un cours de physique. Ce module d'enseignement est décomposé en quatre étapes successives qui sont :

- l'activité *apprentissage du cours* associée au rôle *Étudiant* ;
- l'activité *réalisation d'exercices* associée au rôle *Étudiant* ;
- l'activité *correction* associée au rôle *Enseignant* ;
- l'activité *discussion sur le module* associée aux rôles *Étudiant* et *Enseignant*.

4.1.2 Modélisation

Un des paramètres à prendre en compte lors de la réalisation du modèle est l'individualisation du processus global d'apprentissage, i.e. la possibilité pour chaque étudiant d'avancer à son rythme dans certaines parties du processus.

L'enseignant doit décider de la manière dont les étudiants du groupe vont réaliser ces différentes activités. Nous pouvons distinguer deux grands modes de fonctionnement :

- Le premier mode consiste à dire qu'une activité est terminée uniquement lorsque tous les étudiants ont terminé. Cela signifie qu'un étudiant ne peut commencer l'activité *réalisation d'exercice* que lorsque tous les autres étudiants ont terminé l'apprentissage du cours. L'avancement est réalisé de manière synchrone pour tous les étudiants. Cette politique est très rigide. Elle est une copie presque parfaite des cours en présentiel.
- Le deuxième mode de fonctionnement consiste à déterminer des enchaînements d'activités qui peuvent être réalisés de manière autonome par chaque étudiant. Dans notre exemple, l'enseignant peut souhaiter que les étudiants apprennent le cours puis réalisent les exercices à leurs rythmes (et dans le respect des contraintes temporelles). De plus, il souhaite corriger les exercices au fur et à mesure que les étudiants envoient leurs résultats. Cela signifie qu'un étudiant qui a terminé l'apprentissage du cours peut réaliser les exercices sans attendre que les autres étudiants aient terminé la première activité. Cette politique correspond plus à la philosophie d'enseignement flexible associé à l'éducation à distance. L'étudiant évolue à son rythme à l'intérieur d'un groupe.

Pour supporter ces deux modes, nous utilisons des sous-processus. Dans notre exemple, l'enseignant décide que les trois premières étapes peuvent être réalisées de manière indépendante par chaque étudiant. L'enchaînement de ces activités réalisées de manière individuelle (cours, exercice, correction) est

donc modélisé à l'intérieur d'un processus. Le processus global du module d'enseignement est alors composé de deux activités séquentielles, la première faisant référence au processus individuel, la deuxième étant l'activité de discussion qui est réalisée de manière synchrone entre tous les participants.

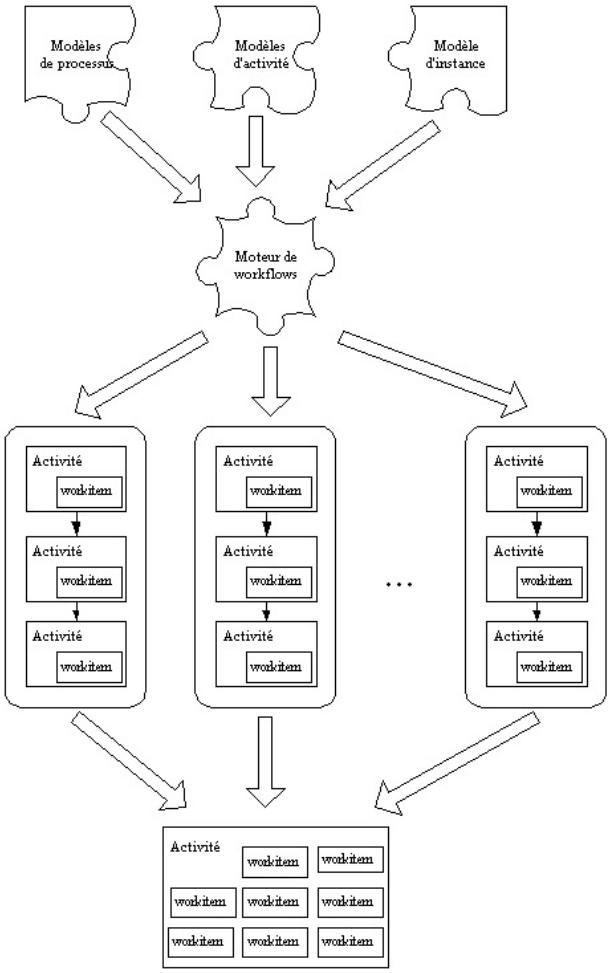

FIG. 9 – *Instanciation des modèles*

4.1.3 Aspect temporel

La gestion du temps est un élément fondamental dans les plates-formes d'éducation à distance. Deux cas de figures sont possibles, l'enseignement de type formation continue et l'enseignement de type formation de groupe. Pour une formation de groupe, il ne peut pas y avoir de trop grand décalage entre les différents étudiants. L'enseignant, comme dans une formation traditionnelle, doit spécifier des dates butoirs pour la réalisation des activités. Lors de la création des modèles il est possible de spécifier deux éléments temporels. Le premier, appelé *limit*, correspond au temps total alloué pour réaliser la tâche. Le deuxième, appelé *deadline*, correspond à la date maximale de fin de l'activité. Par exemple si l'on souhaite que l'activité `exercice` dure au maximum 2 heures et qu'elle soit terminée au plus tard 7 jours après sa création, on déclarera une limite de `2h` et une deadline de `7jours`. Lorsqu'une *limit* est atteinte, le système sauvegarde l'état courant de l'activité et la termine. Lorsque qu'une *deadline* est atteinte, l'activité est suspendue. L'enseignant responsable et l'étudiant concerné sont avertis. L'enseignant décide alors de la suite à donner (fin de l'activité, allocation d'un délai supplémentaire, ...)

Ces contraintes peuvent être relâchées dans le cadre d'un enseignement de type formation continue, où l'apprenant, qui travaille généralement seul, peut avancer complètement à son rythme.

4.1.4 Exécution des modèles

Lors de la création du processus, nous passons en paramètre un modèle d'instance représentant les liens entre le modèle et les ressources (participants et documents). Par exemple, dans le modèle nous avons défini 2 rôles (enseignant et étudiant). Dans le modèle d'instance nous indiquons les personnes ayant le rôle enseignant et celles ayant le rôle étudiant. L'assignation peut être globale pour le modèle, ou n'être valable que pour une activité particulière. L'utilisation de modèles d'instances, nous permet une réutilisation des différents modèles existants, aussi bien pour les processus que pour les activités.

Le fonctionnement global est illustré figure 9. À partir des modèles de processus et d'activités ainsi que du modèle d'instance, le moteur de workflows va créer un sous-processus pour chaque étudiant. Ce sous-processus comporte les trois étapes Cours-Exercices-Correction. Chacune de ces activités ne contient qu'un seul workitem. Lorsque tous les sous-processus sont terminés, le moteur crée l'activité de discussion. Cette activité est composée de n workitems, un pour chaque étudiant et un pour l'enseignant.

Dans le paragraphe précédent nous avons vu que nous pouvions lancer un module de formation pour un unique élève. Nous allons maintenant voir la méthode utilisée pour la création de parcours individuel dynamique.

4.2 Parcours de formation dynamique

Dans le cadre de la formation continue, la difficulté principale réside dans l'impossibilité de connaître le processus global que suivra un apprenant. Nous nous sommes inspirés des travaux de la plate-forme FlexeL.

Pour résoudre ce problème, nous disposons d'un modèle de processus comprenant les étapes initiales de tout apprenant (Inscription, paiement, ...) suivi d'une activité de choix dans laquelle l'apprenant

décidera des modules d'enseignement qu'il souhaite suivre.

Ainsi, l'étudiant se crée un parcours personnalisé. Nous stockons les parcours dans l'objectif d'analyser le comportement et le cheminement de l'étudiant afin de comprendre ses choix, et ainsi cristalliser l'expérience pour proposer un parcours semblable à des utilisateurs ayant le même profil. Cette partie est en cours de réalisation.

5 Conclusions et perspectives

Dans cet article, nous avons vu que les systèmes de workflows, grâce à leur flexibilité, peuvent être utilisés pour la coordination dans un système d'éducation à distance. Nous avons présenté les méthodes que nous utilisons pour représenter les modèles de formations et leurs instanciations dans COW, notre moteur de workflows.

Actuellement, nous intégrons notre moteur dans la plate-forme de campus virtuel de la société Archimed. Cela nous permettra de réaliser des expérimentations grandeur nature en utilisant les ressources gérées par le Campus Virtuel.

La modélisation des processus d'enseignement reste un problème non réellement résolu. Nous avons montré qu'il était possible de modéliser un cours, mais cela demande certaines compétences en informatique et notre modèle n'exprime pas de manière suffisamment compréhensible les différents éléments d'un enseignement. Pour répondre à ce problème, nous débutons un travail portant sur le langage EML (Educational Modelling Language) [16] qui a pour objectif la réalisation d'un modeleur graphique d'enseignement générant des modèles au format EML et une traduction des concepts vers notre langage de description de workflows.

Un autre type de flexibilité que nous voulons traiter au sein du laboratoire porte sur la possibilité d'accéder à une plate-forme d'enseignement de manière personnalisée via différents types de périphériques. Pour cela, nous avons initié un travail avec les spécialistes IHM de notre laboratoire pour fournir des interfaces utilisateurs personnalisables et adaptables au matériel (PC, PDA, GSM, ...). L'architecture que nous avons mise en place est décrite dans [17].

Remerciements

Une partie du travail présenté dans cet article est financée par la société Archimed (`http://www.archimed.fr`).

Bibliographie

1. Claude Viéville – Learning Activities in a Virtual Campus, Chapitre 15 dans *The Digital University - Building a learning Community*, Reza Hazemi et Stephen Hailes (Eds), Springer, pages 215-227, 2002,
2. Object Management Group (OMG) – Workflow Management Facility Specification, Version 1.2, 2000. `http://www.omg.org/technology/documents/formal/workflow_management.htm`.
3. Vlada Matena et Mark Hapner – Enterprise JavaBeans^{TM} Specification Version 1.1. Sun Microsystems, mai 1999.
4. Workflow Management Coalition. `http://www.wfmc.org`.
5. Workflow Management Coalition – Workflow Process Definition Interface - XML Process Definition Language, 22 may 2001. `http://www.wfmc.org/standards/docs/xpdl_010522..pdf`.
6. Workflow Management Coalition – Terminology & Glossary. WfMC-TC-1011, Version 3.0, février 1999 .`http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf`.
7. Workflow Management Coalition – The Workflow Reference Model. WfMC-TC-1003, Version 1.1, 19 janvier 1995.
8. Gregor Kiczales, Jim des Rivières et Daniel G. Bobrow – The Art of the Metaobject Protocol. MIT Press, septembre 1991. ISBN : 0-26-261074-4.
9. Johann Eder et Walter Liebhart – The workflow activity model WAMO. In *Proceedings of the third international conference on cooperative information systems (Coopis'95)*, Wien, Autriche, mai 1995.
10. Olivera Marjanovic et Maria E. Orlowska – Making Flexible Learning More Flexible. IEEE International Workshop on Advanced Learning Technologies IWALT'2000, décembre 2000. `http://flex-el.com`.
11. Distributed Systems Technology Centre `http://www.dstc.edu.au`.
12. Archimed SA. `http://www.Archimed.fr`.
13. Spécification XML Protocol. `http://www.w3.org/2000/xp/`.
14. Web Services Description Language (WSDL) 1.1. `http://www.w3.org/TR/wsdl`.
15. Andrew Tanenbaum – Systèmes d'exploitation: Systèmes centralisés, Systèmes distribués. 1994, InterEditions.
16. Rob Koper – Modeling units of study from a pedagogical perspective, the pedagogical meta-model behind EML. `http://eml.ou.nl/introduction/docs/ped-metamodel.pdf`
17. Thomas Vantroys et José Rouillard – Workflow and Mobile Devices in Open Distance Learning. IEEE International Conference on Advanced Learning Technologies ICALT 2002, Kazan, Tatarstan, 9-12 septembre 2002. `http://lttf.ieee.org/icalt2002/`.