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

Volume 17, 2010
Article de recherche

iFrimousse : Portails web éducatifs augmentés de terminaux mobiles

Florent Carlier, Valérie Renault (CREN - INEDUM, Le Mans)

■ RÉSUMÉ : Dans le cadre de dispositifs éducatifs, et en particulier de portails web, l'apprenant et le tuteur peuvent être déroutés devant la densité d'informations présentes, l'évolution des pratiques pédagogiques vers des supports mobiles et des difficultés d'interactions asynchrones. Notre problématique est d'augmenter les portails éducatifs Web grâce à des agents animés et à des supports mobiles en vue d'y délocaliser des informations synthétiques adaptées à un acteur (apprenant ou tuteur) dans un certain contexte. Nous présentons une architecture épiphyte venant s'intégrer dans un portail éducatif déjà existant. Nous illustrons ce travail à travers deux scénarios complémentaires, l'un dédié à l'expérimentation de la mobilité de l'agent sur une plateforme Moodle et le second à l'aide au tuteur via un système mobile, tel qu'un iPhone.

■ MOTS CLÉS : Aide à l'utilisateur, supports mobiles, architecture épiphyte, portails Web augmentés, système multi-agents.

■ ABSTRACT : In the context of educational devices, and in particular of Web portals, the learner and the tutor can be confused in front of the density of information, the evolution of the educational practices towards mobile supports and the difficulties of asynchronous interactions. Our problem is to increase the Web educational portals thanks to animated agents and to mobile supports to relocate synthetic information adapted to an actor (learner or tutor) in a certain context. We present an epiphyte architecture which overlaps on an already existing educational portal. We illustrate this work through two complementary scenarios, the one dedicated to the experiment of the mobility of the agent on a Moodle platform and the second in the help to the tutor via a mobile system, such as an iPhone.

■ KEYWORDS : Help to the user, mobile supports, epiphyte architecture, increased Web portals, multi-agent system.

1. Introduction

Avec l'essor des technologies de l'information et de la communication, la mise en ligne de cours et de ressources sur des portails éducatifs (*e-learning portals*) s'est largement développée. Ces environnements en ligne ont pour objectif de favoriser une situation d'apprentissage à distance mais annoncent aussi des changements dans les pratiques d'enseignement et d'interaction avec la salle de classe traditionnelle. Un concepteur de cours dispose maintenant d'un large éventail d'outils lui permettant, par exemple :

- de mettre des ressources à la disposition des étudiants et de partager le matériel didactique,
- d'organiser des discussions en ligne,

- de recueillir et d'examiner les situations d'apprentissage,
- de suivre et d'aider des étudiants à distance.

Un grand nombre de portails d'apprentissage reposent sur des gestionnaires de contenu (*Course Management Software - CMS*), comme Moodle (Rice, 2006), Claroline (Claroline, 2008) ou Sakai (Sakai, 2008) pour ne citer que les plus communs. Les CMS permettent aux concepteurs des cours de structurer les informations mises à disposition sans toutefois recourir à des connaissances informatiques complexes. Cependant, de nombreux auteurs, comme Jafari (Jafari, 2002), soulignent les limites des portails d'e-learning pour les enseignants aussi bien que pour les étudiants. Du point de vue des étudiants, un apprenant est seul avec lui-même face à son ordinateur (aucune communication directe avec l'enseignant), la collaboration entre les apprenants est souvent laissée pour compte et les méthodes pédagogiques ne sont pas toujours adaptées aux besoins de chacun. En outre, la quantité d'information fournie par ces portails d'e-learning est si importante que les apprenants et les enseignants sont facilement perdus parmi tous les cours disponibles, les exercices à faire, les actualités, les forums, etc.

Si nous rajoutons à cela le fait qu'en plus de ces portails, les étudiants ont de plus en plus souvent accès à un Espace Numérique de Travail (ENT) très complet incluant, par exemple, leur planning et un espace de travail, ainsi qu'au site de leur université, de leur bibliothèque et de leurs enseignants, il est aisé de comprendre qu'un étudiant peut rapidement être noyé par la masse d'informations mises à sa disposition. Une des conséquences est qu'il peut passer complètement à côté d'informations importantes.

De plus, un apprenant, au cours de sa journée, n'a pas tout le temps accès aux mêmes supports technologiques, qu'il s'agisse de son ordinateur personnel, d'un ordinateur d'une salle de travaux pratiques (TP) ou d'un terminal de consultation d'information dans une bibliothèque. Dans certains contextes, il peut même avoir besoin de supports alors qu'il n'y en a pas de disponible : consultation de son emploi du temps dans les couloirs (les cours étant souvent distants du tableau d'affichage), discussion avec d'autres étudiants dans les transports en commun au sujet d'un exercice à faire, etc.

Il en est de même pour les tuteurs qui mettent des cours ou des devoirs (au sens Moodle) en ligne. Eux peuvent, en plus, être pris au dépourvu face aux informations dont ils disposent pour évaluer les étudiants.

Par contre, la plupart des utilisateurs (qu'ils soient étudiants ou tuteurs) possèdent des terminaux mobiles de plus en plus puissants, "incarnés" dans leur téléphone portable. Avec la fusion des assistants personnels et des téléphones, avec l'essor du WIFI, nous pouvons maintenant envisager des applications ubiquitaires, qui s'adaptent à la situation de travail de l'utilisateur pour s'afficher différemment selon le contexte.

Notre problématique est *d'augmenter* (au sens de la réalité augmentée) les portails éducatifs Web grâce à des agents animés et à des supports mobiles en vue d'y délocaliser des informations synthétiques adaptées à un acteur (apprenant ou tuteur) dans un certain contexte. Ce travail a débuté au Laboratoire LIUM et se prolonge actuellement au Laboratoire CREN, pôle manceau « Innovation en éducation ».

Notre proposition se décline en deux axes complémentaires : les agents et les supports mobiles. Tout d'abord, nous montrons comment nous avons introduit un agent conversationnel animé afin d'améliorer les interactions entre le portail Web et l'utilisateur, et faciliter en particulier l'accès à l'information. Nous présentons l'architecture épiphyte développée pour cet agent, ainsi que le module complémentaire apporté à cette architecture afin qu'elle supporte aussi des applications mobiles. Enfin, nous montrons comment un support mobile peut servir à synthétiser les informations *utiles* à un utilisateur dans un contexte donné, et cela en complément du portail Web qu'il manipule.

2. Contexte de recherche : Portails Web augmentés

2.1. Agents conversationnels animés pour l'aide en ligne

2.1.1. Agents conversationnels animés et e-learning

Depuis (Lester et al., 1997) et (Chan, 1996), beaucoup de recherches ont été réalisées autour des agents compagnons, des tuteurs intelligents ou des agents d'apprentissage (Capuanon et al., 2000) ; (Graesser et

al., 2004) ; (Johnson et al., 2000) ; (Lester et al., 1999) ; (Shaw et al., 2000). La plupart de ces agents sont mis en œuvre dans des logiciels « hors ligne » avec des applications interactives basées sur un niveau élevé de description des scénarios. Dans toutes les tâches que doivent faire les agents (aider à faire des exercices, apprendre de nouvelles notions, aider le tuteur à suivre les activités, etc.), la participation des étudiants suit un ensemble d'instructions et d'actions prédéfinies. De même, les agents suivent des scénarios pédagogiques prédéfinis.

Sur les sites Web, les ACA (Agents Conversationnels Animés) sont en général dédiés à des tâches d'accueil en conseillant les utilisateurs sur le site Web ou à des tâches commerciales (As an Angel, 2009) en aidant les utilisateurs à faire leurs courses, par exemple. Notre objectif est d'introduire des ACA dans des portails de e-learning, afin de donner aux sites Web une dimension conversationnelle et animée pour améliorer l'interaction homme-machine (Cassel, 2000). De nombreux travaux ont montré une augmentation de l'attention, des interactions et de la motivation de l'apprenant face à un personnage virtuel (Baylor, 2002) ; (Lester et al., 1997). Il semble nécessaire d'offrir de tels outils aux utilisateurs de portails, afin de les aider dans leur formation en ligne (Jafari, 2002). Ces outils pourraient également être utilisés par des tuteurs pour suivre les progrès de leurs élèves. Les ACA rendent possible l'amélioration de l'interactivité entre le portail et l'utilisateur, ils augmentent aussi l'attrait de la plate-forme et la motivation de l'apprenant et ils apportent une assistance personnifiée et personnalisée à l'élève.

2.1.2. Scénario 1 : Aide à l'apprenant

Les portails de type Moodle ou Claroline permettent à l'enseignant de préparer des séances de travaux pratiques (TP) ou de mettre en ligne des devoirs à faire entre deux séances. Dans un scénario classique d'utilisation de la plate-forme, chaque étudiant (ou binôme d'étudiants) dispose d'un poste de travail sur lequel il peut télécharger les énoncés des exercices qu'il doit faire lors de la séance, puis, il doit mettre en ligne, au fur et à mesure, les résultats obtenus.

En prenant l'exemple d'un étudiant qui apprend à concevoir des sites Web, il doit afficher sur son poste de travail :

- le portail Web où il trouve les consignes et où il dépose ses résultats,
- l'éditeur Web lui permettant de réaliser son travail,
- le navigateur visualisant le rendu final de son travail,
- et, accessoirement, une ou deux autres pages Web lui permettant de valider son code ou de trouver des informations sur certaines balises.

De même, un étudiant devant faire une recherche bibliographique devra afficher sur son écran : le portail Web, ses documents sources et son document en cours de rédaction.

L'écran devient donc rapidement 'trop petit' et il devient vite difficile de fournir à l'étudiant un système d'aide en ligne visible à chaque instant, sauf à réduire la taille de l'écran dont il dispose. De là vient notre proposition d'augmenter le poste de travail de l'étudiant via un terminal mobile lui servant d'aide personnalisée en fonction des actions qu'il a déjà effectuées. Cette aide sera mise en œuvre, à terme, par l'intermédiaire de l'ACA qui aura la possibilité de migrer de l'écran vers le terminal mobile, et inversement, selon le contexte.

2.2. Délocalisation de l'aide en ligne sur des supports mobiles

2.2.1. Scénario 2 : Aide au tuteur

Ce scénario concerne l'instrumentation des activités du tuteur durant une séance de travaux pratiques (TP) avec des étudiants. Comme dans le scénario précédent, chaque étudiant peut télécharger les énoncés des exercices qu'il doit faire lors de la séance sur un portail Web. Il doit ensuite mettre en ligne, au fur et à mesure, les résultats obtenus. L'enseignant se déplace derrière les élèves en fonction des questions des uns ou des autres.

Lors de la séance de TP, et en fonction de l'autonomie laissée aux étudiants, l'enseignant qui passe derrière ses étudiants pour répondre aux questions, n'a pas forcément le temps nécessaire pour analyser

les dépôts de chacun. De plus, bien souvent, les enseignants vont voir en priorité les étudiants qui posent des questions, ce qui n'est pas toujours le cas des étudiants qui ont des difficultés. Notre proposition dans ce cas là est de fournir à l'enseignant une synthèse de l'avancement de chaque étudiant sur un support mobile qu'il peut garder avec lui lorsqu'il se déplace dans la salle de cours.

Les critères d'acceptabilité de cette synthèse sont :

- une visualisation de l'état d'avancement de l'ensemble des étudiants afin de distinguer rapidement ceux qui ne sont pas au même niveau que les autres ;
- une rapidité d'accès à l'information sur l'avancement individuel de chaque étudiant.

2.2.2. Typologie des systèmes d'aide à l'apprenant et au tuteur

De nombreux systèmes ont été conçus afin de faciliter le travail de l'enseignant. Ces systèmes peuvent être rangés en trois catégories :

- soit l'enseignant se connecte sur son propre poste de travail au portail Web afin de visualiser les dépôts de chaque étudiant (cas des plateformes développées sous Moodle) ;
- soit l'enseignant dispose d'une interface spécifique sur son propre poste de travail avec des informations synthétisant le travail de chaque étudiant (Teutsch et al., 2010) ;
- soit l'enseignant parcourt le travail de l'étudiant sur le poste de ce dernier.

Dans les deux premiers cas, la synthèse se fait généralement *a posteriori* de la séance de TP. Dans la dernière catégorie, c'est au tuteur de faire lui-même la synthèse pendant la séance de TP en fonction de ce qu'il voit sur les différents postes étudiants. Dans notre scénario, la synthèse est faite par le système au cours du TP.

3. iFrimousse : Architecture épiphyte aux portails Web éducatifs

Notre objectif principal est de définir une architecture suffisamment générique et facilement configurable, pour proposer des méthodes permettant d'*augmenter* les portails e-learning déjà existants de systèmes d'aide ambiants. Cette aide doit pouvoir, selon les contextes, être localisée sur la machine de l'utilisateur ou déportée sur un support mobile. Comme le site n'a pas forcément été conçu pour développer de nouveaux services distants, nous proposons une méthode indiquant l'emplacement du contenu pertinent (pertinent au sens du concepteur de la plateforme) dans une page du site web existant. Cette information permettra alors de pouvoir lier le site web avec un système mobile synthétisant d'autres informations liées au site.

3.1. Approches clients-serveurs

Généralement, trois types d'approches peuvent être admis afin de mettre en place une architecture pour les applications web (Richard, 2008) ; (Srivastava et al., 1999) :

- Approche orientée serveur (SOA) (Brusilovsky et al., 2003),
- Approche orientée client (COA),
- Approche orientée par proxy (POA) (Giroux et al., 1996).

Les SOA sont les approches dans lesquelles les traitements d'une architecture Web sont relégués au serveur d'hébergement. Généralement, ces approches sont totalement transparentes pour l'utilisateur : il ne sait même pas que le contenu qu'il reçoit a été intercepté et modifié. Elles permettent de filtrer des flux entiers de données, et leur intérêt principal est d'être capable de produire plus d'informations pour l'architecture, facilitant ainsi sa conception et le développement. Le contrôle sur ce qui est fait est alors absolu : la machine du client n'est responsable d'aucun traitement.

Toutefois, l'architecture doit être en mesure de gérer les accès entrants simultanés, et il est impératif de

s'assurer que les changements opérés sur le serveur soient minimes, car l'architecture ne doit pas affecter sérieusement les performances d'un serveur de production.

Les COA sont les approches dans lesquelles les traitements sont gérés par la machine de l'utilisateur. Concrètement, ces approches dépendent du langage de script ou de l'adaptation du navigateur existant. On peut être amené à modifier le comportement du navigateur afin d'obtenir l'ensemble des traces utiles. Leur principal intérêt est qu'ils permettent de profiler le comportement et d'adapter le contenu aux goûts de l'utilisateur, permettant de cette façon une aide plus pertinente. Malgré cela, un tel concept implique que l'utilisateur ait un rôle actif dans le processus et qu'il change ses habitudes.

Les POA sont des approches où l'ensemble de l'architecture est branché entre le serveur Web et les navigateurs clients. Elle repose sur l'interception des données du client par la passerelle (Proxy), puis sur la redistribution de ces données au serveur Web, après un traitement potentiel. Cette approche offre ainsi un contrôle sur l'ensemble des données transmises. La mise en cache de la passerelle peut être utilisée pour réduire le temps de chargement d'une page Web par les utilisateurs (Cohen et al., 1998). L'intérêt d'une telle approche est d'être considérée comme une solution hybride des deux autres, en gardant les avantages des deux. En effet, comme l'approche SOA, un contrôle conséquent est possible sur les données transmises par les deux parties. De plus, comme pour le COA, les en-têtes des demandes peuvent être utilisées pour capturer les informations pertinentes. L'inconvénient majeur est d'obliger une configuration manuelle de la part de l'utilisateur, ce qui signifie qu'un contrôle de l'utilisation de l'architecture n'est pas possible : si l'utilisateur oublie d'activer la passerelle, toute l'architecture sera contournée. L'usage de l'approche POA implique donc une coopération pleine et entière de l'utilisateur pour fiabiliser les traces obtenues. En outre, les informations capturées ne sont pas équivalentes à la somme des informations des deux autres approches. En effet, les informations relatives à l'usage de l'interface utilisateur ne sont pas recueillies par les systèmes POA. De plus, tout le trafic de la plate-forme est intercepté par l'architecture, abaissant ainsi la performance de l'ensemble du système.

3.2. Approche proposée

La figure 1 montre l'architecture proposée et explique ses mécanismes. Notre contexte est un peu particulier, nous souhaitons développer une architecture suffisamment générique afin de mettre en relation des systèmes d'aide ambiants à des plateformes d'apprentissage existantes (Pierre, 2007). Nous voulons en particulier lier des systèmes mobiles (avec la présence ou non d'ACA) avec des portails existants sans les modifier en profondeur. Les plateformes Web possèdent un jeu de balises HTML (Hypertext Markup Language) qui ne permettent pas de connaître les emplacements des informations sémantiquement utiles pour nos besoins. Notre premier travail a donc consisté à définir une architecture permettant de localiser sur le site certaines informations liées au cours (auteur, devoir, dates limites, etc.).

Nous proposons une architecture globale en couches orientée serveur (SOA), basée sur le concept de middleware. Elles sont constituées à partir de quatre parties : 'réseau', 'balises de localisation', 'système mobile' et 'ACA dynamique'. Chacune de ces couches utilise une approche spécifique, et est conçue à des fins de réutilisation et de généricité. Nous soulignons l'importance d'avoir une architecture totalement transparente pour l'utilisateur.

La première couche est la couche '*réseau*' qui peut être considérée comme la base de l'architecture. L'efficacité globale du système repose sur sa capacité à intercepter tous les flux de données nécessaires pour leurs manipulations ultérieures par notre système. Elle est essentiellement vue comme une couche de communication permettant aux applications d'interagir entre les divers logiciels et les environnements réseau. Ce concept a été normalisé pour le langage Python, sous le nom WSGI (Web Server Gateway Interface) (Eby, 2006) et est spécifique pour une interface entre les serveurs Web et des applications Web Python, afin de promouvoir la portabilité des applications Web à travers une variété de plates-formes.

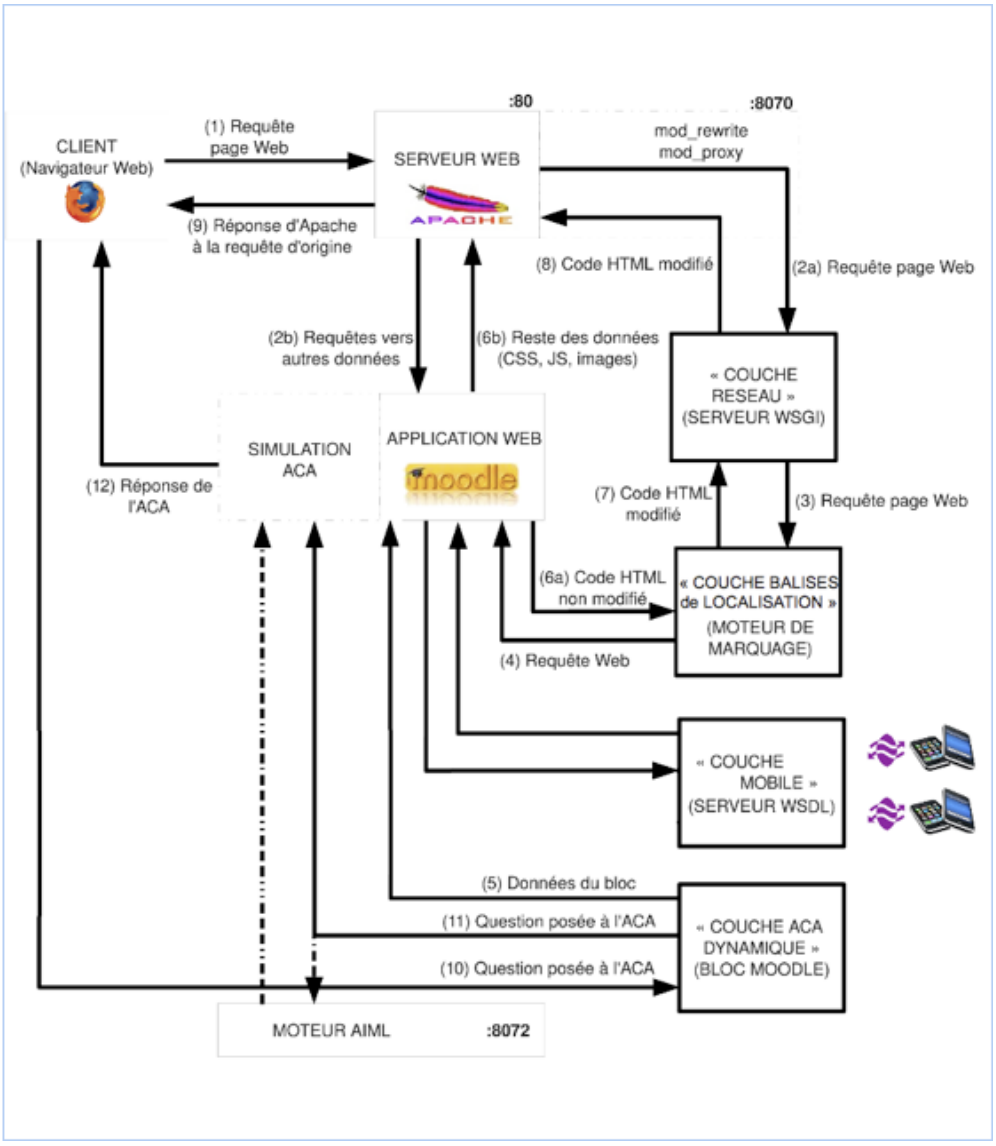

Figure 1 • Organisation de l'architecture épiphyte.

L'architecture modulaire d'Apache assure le lien entre le serveur Web d'origine et notre couche 'réseaux', en redirigeant les requêtes importantes. La couche consiste en un serveur dédié WSGI, qui est le seul point d'entrée pour toute l'architecture, développée en utilisant un framework Python (Hellegouarch, 2007). Cette conception nous permet de capturer uniquement le trafic souhaité, et de laisser tous les contenus inutiles passer par la voie standard. De cette façon, les performances du serveur Web pour la plateforme cible ne sont pratiquement pas affectées, et les autres applications Web tournant sur le serveur ne sont pas touchées. En outre, l'utilisateur n'a pas conscience du traitement réalisé sur serveur.

La seconde couche concerne la couche de '*balises de localisation*'. Son mécanisme consiste à insérer du contenu supplémentaire dans les pages transmises au navigateur du client. Dans le cas présent, ces données supplémentaires sont destinées à des fins de localisation, sous la forme de balises considérées comme des points d'ancrage pour la dernière couche. Cette couche est structurée par un moteur de marquage, responsable de l'insertion de balises et/ou de modules, cette couche étant dédiée à chaque plateforme spécifique. Chaque balise a un type (très utile pour l'analyse de bloc d'informations, par exemple) et un identifiant, qui est unique sur une page donnée. La décision de marquage est basée sur la reconnaissance de modèles contenus dans les pages de la plate-forme d'apprentissage. Une autre fonctionnalité proposée par cette couche est la génération de liste de balises d'une page au format XML, permettant à un système tiers d'interroger l'architecture sur le contenu de la page, afin de cartographier la plate-forme, par exemple. Comme nous le verrons plus tard, l'architecture de notre système mobile vient

se greffer à ce niveau là. Le développement a été assuré en utilisant un Framework Python HTML/XML (Anguenot, 2005). Tout ce travail de marquage assure l'obtention de journaux d'utilisation de l'architecture et de la plate-forme. La création de la base de données des journaux reste toujours valide, puisque nos deux premières couches ne peuvent pas être contournées par l'utilisateur, contrairement à l'approche POA.

La troisième couche est la couche '*Système mobile*'. Elle a pour but de créer une passerelle entre la plateforme d'e-learning et la flotte de systèmes mobiles. Dans le cadre d'architecture orientée objet, nous utilisons les services Web, permettant l'interopérabilité entre plusieurs systèmes logiciels (agents) sur un réseau informatique (Chauvet, 2002). Plus spécifiquement, nous utilisons les normes du W3C ; l'interface du système est définie par un langage lisible par un ordinateur client. Le langage WSDL (Web Services Definition Language) (WSDL, 2001) décrit l'interface au service. En utilisant XML Schema, WSDL définit les paramètres d'entrée et de retour d'un appel au service Web. D'autres systèmes logiciels vont communiquer avec le service Web selon sa description en utilisant le protocole SOAP (Simple Object Access Protocol) (SOAP, 2007). SOAP offre le transport d'objets sérialisés et autres données en XML et l'appel de procédures distantes. Pour être capable d'utiliser un 'Web Services' et de programmer un client (dans notre cas, un système mobile comme un smartphone), il est nécessaire d'en connaître la définition. Un fichier WSDL a donc été défini pour faire le lien avec le serveur Moodle. Dans le cas de l'usage de notre architecture avec un autre type de CMS, le concepteur devra réaliser un fichier WSDL adapté afin de maintenir l'interopérabilité entre le serveur et le système mobile.

La dernière couche est la couche '*ACA dynamique*', elle gère toutes les interactions avec le client, et a un double objectif : déplacer une entité sur une page Web afin de produire de la rétroaction attendue à l'utilisateur (mettre en évidence des liens ou des informations, par exemple), et faciliter son intégration avec un système sous-jacent. En raison de son caractère hautement interactif, cette couche utilise l'approche COA, basée sur les technologies AJAX. Le premier objectif est atteint en utilisant les balises précédemment insérées pour localiser les zones pertinentes de la page et obtenir leurs coordonnées pour faire déplacer l'ACA. Ceci est réalisé avec l'aide d'un Framework AJAX. Le deuxième objectif est atteint en développant des méthodes spécifiques sous forme de fonctions API de base pour un système sous-jacent. Par exemple, l'API nous permettra de déplacer notre ACA (avec des paramètres tels que le texte souhaité à afficher, l'angle de l'ACA utilisé pour pointer le contenu, le retour à son point d'origine après le déplacement, le mouvement de la boîte de dialogue et l'attitude de l'ACA) et de fournir des informations textuelles. Le code résultant peut être inséré à l'aide des mécanismes de plate-forme spécifique (comme le Bloc Moodle) ou avec un moteur spécifique pouvant être facilement développé, car il est similaire à une couche de 'métadonnées'. Les communications entre le client et le serveur sont gérées par les objets XMLHttpRequest (Rice, 2008).

Comme le montre la figure 1, le navigateur du client envoie une requête pour une page Web sur le serveur Web (1). La demande est ensuite redirigée vers la couche 'réseau' (2a), alors que toutes les demandes de données sans intérêt (images, fichiers de script, les feuilles de style) sont directement envoyées à l'application Web (2b). Cette couche transmets la demande à la couche 'balises de localisation' (3). Une fois la demande transmise à l'application (4), la couche 'ACA dynamique' transmet ses données (5) devant être intégrées dans le code HTML d'origine rendu par la couche 'balises de localisation' (6a). Pendant ce temps, les données inutiles sont directement renvoyées au serveur Web (6b). Puis, le moteur de marquage de la couche 'balises de localisation' modifie le code HTML, et les envoie vers la couche 'réseau' (7), dont le rôle est de redistribuer la page vers le serveur Web (8), et ainsi de renvoyer au client original la page préparée avec les modifications (9). Avec cette architecture, le client peut poser une question à l'ACA via la couche 'ACA dynamique' (10). La couche décide du travail à faire pour répondre à la question (il traite directement ou le transmet à l'ACA (11)), puis le client reçoit la réponse à celle-ci (12). Avec une telle architecture, tous les utilisateurs doivent visiter la plate-forme en utilisant le nom de domaine dédié, au lieu de l'adresse habituelle du site.

L'administrateur du site peut être le seul à connaître l'adresse de base de la plate-forme. Il peut de fait, choisir de la désactiver, ainsi les utilisateurs n'ont pas conscience de la modification en ligne des pages visitées. De plus, la performance de l'ensemble du système n'est pas véritablement impactée. Les tests de performances effectués montrent l'aspect négligeable du temps de traitement supplémentaire par demande, en effet il ne dépasse pas 30 milli-secondes en moyenne.

Notre architecture a été mise en place avec la plateforme Moodle et des systèmes mobiles de type iPhone / iPod Touch. Les systèmes mobiles utilisés font partie de la catégorie des smartphones. L'iPod Touch d'Apple allie un écran panoramique et un concentré de technologie numérique incluant Internet, le mail, la musique, et toutes les fonctionnalités d'un assistant numérique personnel. L'iPhone, quand à lui, rajoute des fonctionnalités de téléphonie mobile et un système de positionnement (Global Positionning System : GPS). Ultérieurement, le GPS nous permettra de localiser l'utilisateur (sur le campus, dans les transports, chez lui) en fonction des activités en cours.

Afin de tester la faisabilité de notre architecture, nous l'avons mise en œuvre sur les deux scénarios présentés dans la section 2. L'intérêt de ces scénarios, relativement simples, est de permettre l'implémentation de notre architecture dans un contexte contrôlable en expérimentant de façon indépendante l'ACA dans le premier scénario et une première étape vers la mobilité dans le second scénario.

4. Apprentissage ambiant au travers d'un assistant conversationnel mobile

4.1. CAMELEON : un agent conversationnel animé pour l'aide à la navigation de la plateforme de e-learning

Pour illustrer l'usage de l'architecture précédemment présentée, nous proposons d'utiliser les 'balises de localisation' afin de permettre à un agent conversationnel animé (ACA) de guider l'utilisateur lors de sa navigation Web sur une plateforme d'e-learning. En effet, comme nous l'avons déjà souligné, l'un des problèmes des portails de e-learning est la quantité d'informations contenues sur une plateforme d'apprentissage. Notre objectif est alors d'apporter un ACA capable de se déplacer automatiquement sur quelques pages en pointant l'information recherchée par un utilisateur.

L'Université du Mans utilise une plate-forme Moodle pour mettre en œuvre la formation du C2i (C2i, 2009). Le C2i est un certificat d'aptitude à la maîtrise des outils et des réseaux. Il est établi afin de développer, renforcer et valider la maîtrise des technologies de l'information et de la communication par les étudiants en formation dans les établissements d'enseignement supérieur. Cet environnement est utilisé par plusieurs centaines d'étudiants et sert à valider leur formation. Avec l'approche proposée ci-dessus, nous pouvons insérer des 'balises de localisation' sur les pages, sans avoir à modifier le site lui-même.

La difficulté de faire des tests en parallèle de cette plateforme, nous amène à expérimenter nos outils de navigation sur le Web uniquement avec un groupe d'étudiants, tandis que les autres continuent à utiliser la plate-forme normalement (cf. figure 2). Dans cet exemple illustré dans la figure 2, des balises ont été rajoutées pour indiquer la boite de connexion, la liste des cours en ligne et les différents enseignants. Dans le cadre des tests et afin d'illustrer l'existence de nos balises, le logo ϕ apparaît chaque fois qu'une balise de localisation est positionnée sur la page.

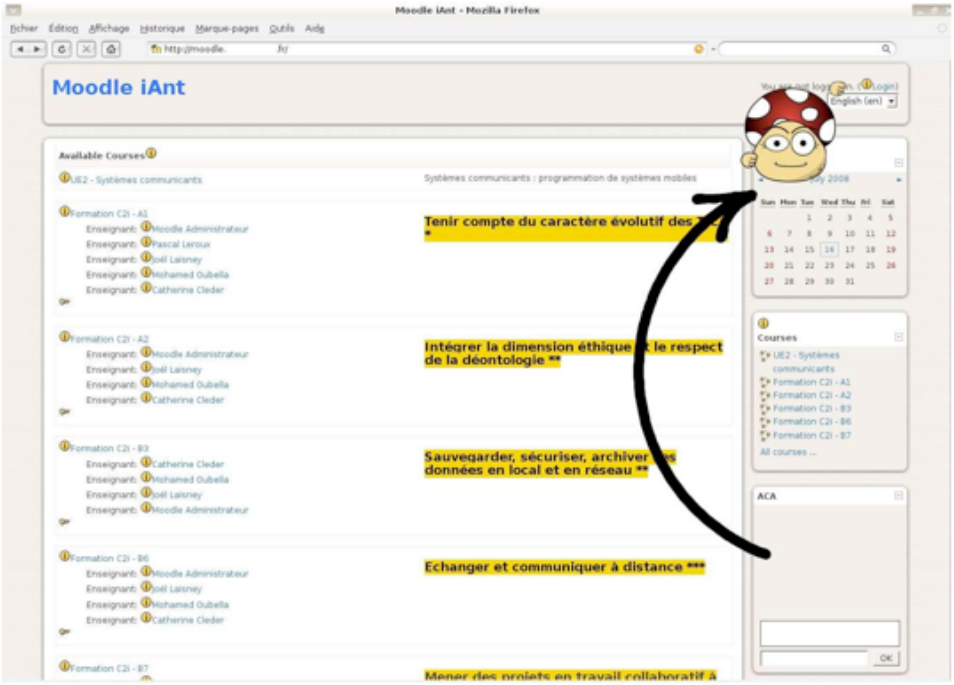

Figure 2 • Exemple de balises de localisation et mouvement de l'ACA vers la balise de connexion.

L'ACA a été développé à partir du modèle Cameleon décrit dans (Courtine et al., 2007). Ce modèle est dédié aux portails de e-learning. Contrairement à des environnements logiciels hors ligne, les portails de e-learning sont toujours en évolution : tous les utilisateurs peuvent ajouter, mettre à jour et supprimer des matériels d'apprentissage. Dans ce contexte, l'agent doit mettre en œuvre des capacités d'apprentissage et d'adaptation. La particularité de l'architecture Cameleon est que l'ACA n'est que la partie visible d'un système multi-agents (SMA) filtrant les informations.

Le module de décision est composé de deux catégories d'agents (cf. figure 3) : les agents espions et les agents prédictifs (Courtine et al., 2007). Tout d'abord, la spécification des activités d'apprentissage conduit à implémenter des agents de prédiction. Les agents espions analysent les traces générées par le CMS. La difficulté actuelle est liée au fait que chaque CMS a sa propre représentation de l'information, il n'existe aucune norme ou standard. Cependant, certaines données sont communes à la plupart des plateformes d'apprentissage, tels que les journaux de connexion, les documents téléchargés ou le suivi des messages internes. Les agents d'espionnage sont instanciés, dès que ces données sont présentes dans la base de données CMS. Ces agents disposent d'informations sur ce qui se passe réellement sur le portail. Puis la prédiction et les agents d'espionnage doivent interagir et négocier en fonction des domaines de connaissances du contexte afin de comparer leurs informations et d'informer le module d'interaction de l'ACA.

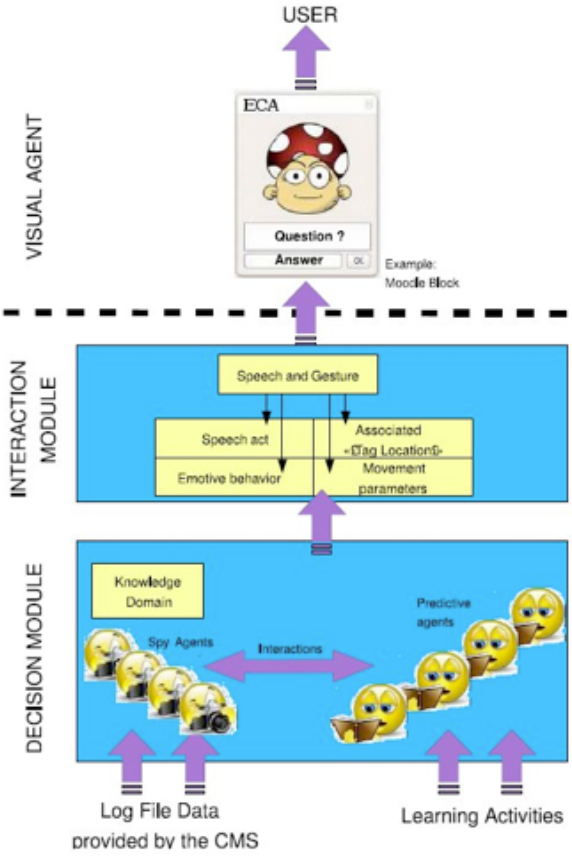

Figure 3 • CAMELEON : un ACA pour l'aide à la navigation sur Internet

Le module d'interaction est dédié à la 'parole' et à la gestuelle. Les mouvements de l'ACA sont définis dans ce module par rapport à l'acte de parole. Pour résumer, un acte de comportement est alors composé d'un acte de parole, d'un comportement dit 'émotif', d'une balise de localisation associée et d'un certain nombre de paramètres de mouvements.

Par exemple :

- * Action de parole :
- question : "Comment puis-je m'identifier ?"
- réponse : "La zone de connexion est ici"
- * Comportement émotif :
- "souriant"
- * Balise de localisation associée :
- "connexion-top"
- * Paramètres de mouvements :
- position de l'ACA par rapport à l'information : "up-left"
- mouvement de la boite de l'ACA : "back-on"

Dans cet exemple, l'agent se déplace donc en souriant vers la zone de connexion, en se positionnant en haut à gauche de cette zone. L'agent revient ensuite à sa position initiale. Ensuite, le module d'interaction est lié à un agent visuel dont le rôle est simplement de s'intégrer dans une boîte de Moodle afin d'interagir

avec l'utilisateur final (cf. figure 3). Cet agent visuel représente l'ACA pour un utilisateur, et il est capable de se déplacer à l'aide de l'API fournie par l'architecture épiphyte.

4.2. Quand le téléphone mobile devient l'assistant des tuteurs

Cette application correspond à la mise en œuvre du scénario 2, évoqué dans la section 2.2.1, afin de faciliter la vision par l'enseignant de l'ensemble des réalisations de ses étudiants lors de la séance de travaux pratiques (TP), et non *a posteriori*.

L'enseignant passe dans les rangs avec son iPhone pour aider les étudiants à réaliser leur TP. Lorsqu'il arrive derrière un étudiant, il peut sélectionner l'icône correspondant à l'étudiant sur son iPhone pour avoir un tableau résumant sa progression, ses exercices réalisés ou non et ainsi donner une aide immédiate à l'étudiant en cas de besoin. Pour chaque étudiant, l'enseignant dispose des informations suivantes :

- la liste des exercices déposés sur la plate-forme,
- pour chaque exercice, une barre de progression indiquant si un fichier a été déposé ou non sur la plate-forme ; et une barre réduite si le poids du fichier n'atteint pas un certain seuil prédéfini.

Pour accéder rapidement à un étudiant, l'enseignant peut le sélectionner dans une liste représentée sous la forme d'un carrousel (cf. figure 4), la couleur de fond de l'icône de chaque étudiant correspondant aux données issues des barres de progression. Cette visualisation permet à l'enseignant de voir immédiatement la progression générale de son TP et de détecter les étudiants ayant des difficultés.

Figure 4 • Application mobile sur l'iPhone/iPod Touch d'Apple

Cette représentation est issue de travaux sur la plateforme DividingQuest (Girard et al., 2006), un EIAH couplé d'une plateforme logicielle d'évaluation pour l'étude de l'impact sur l'usage d'une interface avec des personnages émotifs pour l'aide à l'apprentissage. Le DividingQuest est un environnement d'apprentissage intelligent de type Vygotskien (Vygotsky, 1978) utilisé dans les écoles anglophones du primaire. Le système de couleur des feux de régulations du trafic routier est utilisé comme métaphore dans la plupart des écoles anglaises invitant l'élève à être un participant de son propre processus d'apprentissage. Le but est de lui permettre de comprendre ses résultats accomplis et de définir ses prochains objectifs d'apprentissage. Le DividingQuest contient 15 activités mathématiques liées à l'apprentissage des divisions, organisées en cinq zones contenant elles-mêmes trois niveaux d'expertise sur une division

spécifique dont chaque enfant doit obtenir les compétences. Le système possède une carte de progression dont la couleur de chaque zone représente l'expertise de l'enfant par rapport à cette compétence particulière. Tous les domaines sont rouges en début de l'apprentissage. Une fois que l'enfant gagne un certain niveau d'expertise dans le domaine, la zone devient orange, et enfin passe au vert une fois que l'enfant maîtrise la compétence de cette division.

Nous avons transposé cette représentation en modifiant la couleur de fond de la carte trombinoscope dans le carrousel. Les couleurs rouge, orange, vert reprennent les trois niveaux de progression que l'enseignant a défini lors de la configuration de l'architecture épiphyte pour la réalisation de son activité sur la plateforme de e-learning. La couleur bleue représente la neutralité d'un étudiant dans le suivi de l'activité en cours, c'est le cas, par exemple, d'un étudiant qui ne souhaite pas prendre part à l'expérience ou qui a déjà terminé l'activité chez lui. Nous sommes en cours de validation de notre processus d'expérimentation pour le tuteur. L'étape suivante sera d'expérimenter cette même interface pour les apprenants. Comme dans le DividingQuest, l'étudiant devrait alors bénéficier des retours de son propre processus d'apprentissage tout au long du TP.

5. Perspectives

Afin de promouvoir l'interaction dans le processus d'apprentissage, le projet iFrimousse a pour ambition d'aboutir à un système d'apprentissage ambiant et mobile, permettant à un assistant virtuel de suivre l'apprenant en fonction de ses besoins et de son contexte de travail en *augmentant* les portails d'e-learning existants.

Cet article présente la première étape de ce projet qui nous a permis d'aboutir à une architecture épiphyte, c-à-d à une architecture capable de se greffer sur des portails existants sans, pour autant, modifier la plateforme d'origine. Pour réaliser cette première étape, il a été nécessaire de mettre en œuvre des balises de localisations à partir de connaissances spécifiques au domaine d'application du site. Ces balises nous permettre de localiser des informations particulières. Nous avons ensuite défini deux scénarios, l'un orienté agent, l'autre orienté terminal mobile, qui nous ont permis d'expérimenter notre architecture dans des contextes réduits et contrôlables.

Notre objectif est maintenant de complexifier les scénarios afin d'étendre la mobilité des acteurs. De nombreuses pistes de recherche s'ouvrent : la première, évidente, est de permettre le déplacement de l'agent du portail Web au support mobile selon le contexte d'utilisation. De nombreuses questions se posent alors : par exemple, que se passe-t-il quand le cours est fini et que l'enseignant a encore des 'alertes' sur son iPhone. L'étudiant peut-il avoir accès à une interface similaire à celle de l'enseignant afin de réaliser son état d'avancement par rapport au groupe ? Dans quels contextes l'ACA doit-il se déplacer sur le terminal mobile afin d'aider l'apprenant ?

Notre architecture étant épiphyte, différentes perspectives s'ouvrent afin d'adapter cette plateforme à d'autres contextes et en particulier au contexte de l'aide à domicile d'une personne à mobilité réduite. Il s'agit alors de coupler notre architecture sur un portail Web domotisé. Notre objectif est alors de permettre à la personne de disposer d'un support mobile intelligent, se déplaçant avec elle dans son domicile afin de l'assister dans ces activités quotidiennes.

BIBLIOGRAPHIE

As an Angel (2009). http://www.asanangel.com/.

ANGUENOT J., (2005). Python et xml. *Proc. of RMLL 2005*.

BAYLOR A. L. (2002). Agent-based learning environments for investigating teaching and learning. *Journal of Educational Computing Research*, Vol. 26 N°3 p. 249–270.

BRUSILOVSKY P., DEBRA P., SANTIC T. (2003). A flexible layout model for a web-based adaptive hypermedia architecture. *Proc. of AH 2003: Workshop on Adaptive Hypermedia and Adaptive Web Based Systems*, p. 77–86.

C2i. (2009). http://www2.c2i.education.fr/.

CAPUANO N., SANTO M. D., MARSELLA M., MOLINAR M., SALERNO S. (2000). A multiagent architecture for intelligent

tutoring. *Proceedings of the international conference on advances infrastructure for electronic business. Science, and Education on the Internet*.

CASSELL J. (2000). Embodied conversational agents: Representation and intelligence in user interface. *AI Magazine*, Vol. 22 N°3 p. 67– 83.

CHAN T. (1996). Learning companion systems, social learning systems, and the global social learning club. *Journal of Artificial intelligence in Education*, Vol. 7 N°2 p. 125–159.

CHAUVET J.-M. (2002) *Services web avec SOAP, WSDL, UDDI, ebXML...*, Editions Eyrolles.

CLAROLINE. (2008). http://www.claroline.net/.

COHEN E., KRISHNAMURTHY B., REXFORD J. (1998). Improving end-to-end performance of the web using server volumes and proxy filters. *Proc. ACM SIGCOMM*, p. 241–253.

COURTINE M., RENAULT V. (2007). Cameleon, a generic model of embodied conversational agent for e-learning portals. ICCE'2007, *International Conference on Computers in Education*.

EBY P. J. (2006). PEP 333 – Python Web Server Gateway Interface v1.0. http://www.python.org/dev/peps/pep-0333.

GIRARD S., JOHNSON H. (2006). DividingQuest: Using emotive interface personas in educational software. *Technical Report CSBU-2006-20, Department of Computer Science, University of Bath*. ISSN 1740-9497.

GIROUX S., PAQUETTE G., PACHET F., GIRARD J. (1996). Epitalk, a platform for epiphyte advisor systems dedicated to both individual and collaborative learning. *ITS'96,* Springer-Verlag Lecture Notes in Computer Science.

GRAESSER A.-C., LU S., JACKSON G.-T., MITCHELL H., VENTURA M., OLNEY A., LOUWERSE M.-M. (2004). Autotutor: A tutor with dialogue in natural language. *Behavioral Research Methods, Instruments and Computers*, Vol. 36 p. 180–193.

HELLEGOUARCH S. (2007). *CherryPy Essentials*. Packt Publishing.

JAFARI A. (2002). Conceptualizing intelligent agents for teaching and learning. *EDUCAUSE QUARTERLY*, 25(3).

JOHNSON W. L., RICKEL J. W. (2000). Animated pedagocial agents: face-to-face interaction in interactive learning environments. *International Journal of Artificial Intelligence in Education*, Vol. 11 p. 47–78.

LESTER J., CONVERSE S., KAHLER S., BARLOW S., STONE B., BHOGAL R. (1997). The persona effect: Affective impact of animated pedagogical agents. *Proc. of the ACM CHI 97 Human Factors in Computing Systems Conference*, p. 359–366.

LESTER J., STONE B., STELLING G. (1999). Lifelike pedagogical agents for mixed-initiative problem solving in constructivist learning environments. *User Modelling and User-Adapted Interaction*, Vol 9 N°1-2 p. 1–44.

LESTER J., VOERMAN J., TOWNS S., CALLAWAY C. (1997). Cosmo: A life-like animated pedagogical agent with deictic believability. *Proc. of the IJCAI97, Workshop on Animated Interface Agents: Making them Intelligent*.

PIERRE S. (2007). *E-Learning Networked Environments and Architectures, a knowledge processing perspective*. Spinger.

RICE W. H. (2006). *Moodle: E-Learning Course Development*. Packt Publishing.

RICE W. H. (2008). *The XMLHttpRequest Object*. \\ http://www.w3.org/TR/XMLHttpRequest.

RICHARD B. (2008). Une approche épiphyte pour la conception de systèmes conseillers. *Thèse de l'Université du Maine*.

SAKAI. (2008) http://sakaiproject.org/.

SANDERS G. A., SCHOLTZ J. (2000). Measurement and Evaluation of Embodied Conversational Agents. Justine Cassell and Joseph Sullivan and Scott Prevost, in *Embodied Conversational Agents*.

SHARPLES M. (2006). "Big issues in Mobile Learning", *Report of a workshop by the Kaleidoscope Network of Excellence Mobile Learning Initiative*, http://mlearning.noe-kaleidoscope.org/repository/BigIssues.pdf.

SHAW E., JOHNSON W., GANESHAN R. (2000). Pedagogical agents on the web. *Proc. of the Third International Conference on Autonomous Agents*., p. 47–78, 1999.

SOAP (2007). *SOAP Version 1.2* Part 1: Messaging Framework (Second Edition). http://www.w3.org/TR/soap12/

SRIVASTAVA J., COOLEY J., DESHPANDE M., TAN P.-N. (1999) Web using mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations*, Vol. 1 N°2 p. 12-23.

TEUTSCH P, BOURDET J-F. (2010). Percevoir les trajets d'apprentissage en formation à distance. Conception pluridisciplinaire d'outils de visualisation pour le tuteur., *TSI, Technique et Science Informatiques, numéro spécial L'informatique à l'interface de l'activité humaine et sociale,* France, Volume 29.

TRICOT A, PLEGAT-SOUTJIS F., CAMPS J.-F., AMIEL A., LUTZ G, MORCILLO A. (2003). Utilité, utilisabilité, acceptabilité : interpréter les relations entre trois dimensions de l'évaluation des EIAH, *Conférence EIAH 2003*, Strasbourg, France, p. 391-402.

VYGOTSKY L.S. (1978). *Mind in society: the development of higher psychological processes.* Cambridge, MA: Harvard University Press.

WSDL (2001). Web Services Description Language (WSDL) 1.1. http://www.w3.org/TR/wsdl

A propos des auteurs

Florent CARLIER est maître de conférences en Informatique à l'Université du Maine depuis septembre 2004. Initialement rattaché au LIUM, il effectue actuellement sa recherche au pôle Innovation en Éducation de l'Université du Maine (INEDUM) rattaché au CREN de Nantes. Ses principaux axes de recherches reposent sur l'analyse des flots d'informations échangés entre une plateforme d'apprentissage et des systèmes mobiles. Ces architectures reposent sur des technologies ambiantes et de réalités augmentées. Des indicateurs de suivi de l'activité sont ainsi générés afin de normaliser la représentation des activités en situation de mobilité.

Adresse : CREN, Université du Maine, Avenue Messiaen, 72085 Le Mans Cedex 9

Courriel : florent.carlier@univ-lemans.fr

Valérie RENAULT est maître de conférences en Informatique à l'Université du Maine depuis septembre 2004. Initialement rattaché au LIUM, elle effectue actuellement sa recherche au pôle Innovation en Éducation de l'Université du Maine (INEDUM) rattaché au CREN de Nantes. Ses recherches reposent sur la conception de systèmes multi-agents, et en particulier d'Agent Conversationel Animé (ACA), facilitant l'appropriation par les apprenants de plateformes d'apprentissages en ligne ou mobiles.

Adresse : CREN, Université du Maine, Avenue Messiaen, 72085 Le Mans Cedex 9

Courriel : valerie.renault@univ-lemans.fr

Référence de l´article :

Florent Carlier, Valérie Renault, iFrimousse : Portails web éducatifs augmentés de terminaux mobiles, *Revue STICEF*, Volume 17, 2010, ISSN : 1764-7223, mis en ligne le 28/01/2011, http://sticef.org