AMBRE-enseignant : un module partenaire de l'enseignant pour faciliter l'intégration de AMBRE en classe

Nathalie Duclosson, Stéphanie Jean-Daubias, Stéphanie Riot

AMBRE-enseignant : un module partenaire de l'enseignant pour faciliter l'intégration de AMBRE en classe

Nathalie Duclosson, Stéphanie Jean-Daubias, Stéphanie Riot

LIRIS et ERTé e-praxis
Université Claude Bernard - Lyon 1
Nautibus
43 Boulevard du 11 novembre 1918
69622 Villeurbanne Cedex

{Nathalie.Guin-Duclosson, Stephanie.Jean-Daubias}@liris.univ-lyon1.fr

RÉSUMÉ. *Si les enseignants utilisent peu d'EIAH avec leurs élèves, c'est peut-être parce qu'ils n'ont pas la possibilité d'agir sur les environnements qu'on leur propose. Or, pour s'approprier un EIAH, il faut que l'enseignant puisse l'adapter au contexte d'apprentissage et à sa démarche pédagogique. C'est pourquoi nous proposons dans le cadre du projet AMBRE un module destiné à l'enseignant pour l'EIAH AMBRE-add. Ce module, AMBRE-enseignant, permet à l'utilisateur de configurer l'EIAH destiné à être utilisé en classe à l'école primaire. Nous avons, en collaboration avec des enseignants, identifié les fonctionnalités de AMBRE-enseignant nécessaires à l'adaptabilité de l'EIAH. Nous avons en particulier conçu un système à base de connaissances qui permet à l'enseignant de générer les problèmes de son choix pour les proposer à ses élèves dans l'EIAH.*

MOTS-CLÉS : *adaptabilité des EIAH, rôle de l'enseignant, génération de problèmes, connaissances.*

1. Introduction

Le projet AMBRE (Apprentissage de Méthodes Basé sur le Raisonnement à partir de l'Expérience) est une étude pluridisciplinaire en informatique, sciences cognitives, didactique des mathématiques et sciences de l'éducation, dont le but est de concevoir des environnements d'apprentissage pour l'acquisition de méthodes [GUIN-DUCLOSSON et al. 02]. Cette étude s'appuie sur des recherches en didactique des disciplines sur l'enseignement de méthodes pour la résolution de problèmes. Dans chaque domaine d'application, une méthode est fondée sur le classement par l'apprenant des problèmes et des outils de résolution. Nous proposons pour faire acquérir de telles méthodes d'utiliser le raisonnement à partir de cas, paradigme développé en Intelligence Artificielle et issu de recherches en psychologie cognitive sur le raisonnement par analogie. Après que le système lui ait présenté des problèmes-types résolus qui lui serviront de modèles, le travail de l'apprenant pour la résolution du problème comporte cinq étapes : lecture de l'énoncé, reformulation du problème, choix d'un modèle pour guider la résolution, rédaction du plan de résolution du problème par adaptation du plan de résolution du modèle, puis association du problème résolu à la classe à laquelle il appartient. Après l'évaluation d'un premier prototype traitant des problèmes de dénombrement en terminale scientifique [NOGRY et al. 02], un système complet concernant des problèmes additifs en école primaire, AMBRE-add a été développé et expérimenté [NOGRY et al. 04].

Afin de favoriser l'intégration de AMBRE à l'enseignement, nous avons conçu un module s'adressant aux enseignants, AMBRE-enseignant, leur permettant notamment de créer de nouveaux problèmes et des séquences personnalisées. Dans cet article, nous montrons tout d'abord les différents rôles que peut avoir l'enseignant dans les EIAH, avant de préciser ceux qui lui sont attribués dans AMBRE. Nous présentons ensuite les différentes fonctionnalités que nous avons conçues pour AMBRE-enseignant, avant de décrire plus en détail le générateur de problèmes.

2. AMBRE : "which role for the teacher?"

La majorité des EIAH sont aujourd'hui encore centrés sur le duo ordinateur/apprenant masquant ainsi la plupart du temps le rôle pourtant capital de l'enseignant dans ces environnements [VIVET 90]. Les EIAH ne considèrent que rarement l'enseignant comme utilisateur potentiel distinct de l'apprenant. On peut d'ailleurs constater que seule une très faible partie de l'intelligence des EIAH est dévolue à l'enseignant. Or il apparaît aujourd'hui comme primordial de considérer que la place de l'enseignant est distincte de celle de l'élève dans les environnements interactifs, et qu'une partie du système devrait être développée à son intention [BOUHINEAU 96], [LEROUX 02]. Pour cela, il est nécessaire d'identifier les différents rôles que peut jouer un enseignant dans les EIAH : concepteur, auteur, prescripteur ou utilisateur.

Concepteur ou partenaire de conception L'enseignant peut participer directement à la conception de l'EIAH, ce qui peut se faire en l'intégrant de manières différentes [JEAN-DAUBIAS 04]. La démarche de conception participative permet aux enseignants d'être acteurs de la conception : ils sont non seulement observés et interrogés sur leurs pratiques, mais aussi intégrés dans le processus de conception, en faisant des propositions novatrices et même en participant directement aux choix de conception, permettant ainsi la création d'un logiciel en adéquation réelle avec les besoins et pratiques réels des enseignants. La conception informative quant à elle fait appel aux enseignants en tant qu'informateurs dans la conception, sans les cantonner à un rôle passif, mais sans pour autant les considérer comme des partenaires à part entière. Ils peuvent par exemple travailler avec les concepteurs sur des prototypes, mais ils ne participent pas aux décisions finales.

Auteur L'enseignant peut être lui-même concepteur d'EIAH en tant qu'utilisateur d'un système auteur, outil mis à sa disposition pour lui permettre de réaliser des logiciels à vocation éducative. Les EIAH étant des logiciels extrêmement complexes, les systèmes auteurs se limitent généralement à la création de supports et de ressources de cours ou bien de tuteurs intelligents. Citons l'exemple de GenDoc développé dans le cadre du projet ARIADNE [DAVID et al. 02].

Prescripteur Dans la plupart des EIAH, l'enseignant endosse le rôle de prescripteur : c'est en effet lui qui choisit le système qu'il fera utiliser à ses élèves, en fonction de ses besoins, de ses préférences, de ses choix pédagogiques.

Utilisateur secondaire C'est le rôle qui est le plus souvent réservé, implicitement la plupart du temps, à l'enseignant dans les EIAH. Dans ce cas, l'EIAH est principalement centré sur l'apprenant, mais laisse tout de même une place à l'enseignant pour gérer son utilisation. Celui-ci peut ainsi adapter et paramétrer le système destiné à ses élèves, l'alimenter en situations pédagogiques, interagir avec les apprenants lors de l'utilisation de l'environnement, mais aussi faire le bilan de la session effectuée par les apprenants. L'enseignant utilise ainsi l'EIAH afin de l'adapter à sa stratégie pédagogique, et à ses utilisateurs principaux, les apprenants, comme le permettent par exemple les logiciels Roboteach [LEROUX 97] et Aplusix [NICAUD et al. 04].

Utilisateur principal Certains systèmes s'adressent directement à l'enseignant pour lui proposer des outils qui l'aideront dans son travail. Les logiciels de ce type ont ainsi une approche centrée sur l'enseignant et non plus sur l'apprenant. Citons le projet PERLEA qui vise à proposer un ensemble d'outils pour faciliter le suivi des apprenants par l'enseignant [JEAN-DAUBIAS 03].

Dans le projet AMBRE, si plusieurs enseignants ont joué un rôle de **partenaires de conception** dans le cadre d'une conception différenciée [JEAN-DAUBIAS 04], l'enseignant a pour rôle majeur celui d'**utilisateur principal** d'un module qui lui est exclusivement destiné. Il pourra ainsi à travers l'environnement proposé tenir en quelque sorte son rôle d'**utilisateur secondaire** du module apprenant de AMBRE, en adaptant et paramétrant l'environnement destiné à ses élèves. Enfin, l'enseignant conserve un rôle de **prescripteur**, car il choisit le logiciel proposé à sa classe.

3. AMBRE-enseignant

Dans le cadre du projet AMBRE, donnant lieu à la création d'EIAH destinés à l'apprentissage de méthodes, nous souhaitons proposer à l'enseignant un environnement lui permettant de paramétrer le logiciel destiné à ses élèves, de créer des séquences destinées à l'ensemble de sa classe ou à certains profils d'apprenants, et plus particulièrement de générer des problèmes à résoudre. Cet environnement, exclusivement destiné aux enseignants, doit donc répondre à leurs besoins, et leur permettre d'intégrer et d'adapter un EIAH AMBRE à leurs démarches, à leurs stratégies pédagogiques, mais aussi aux contextes dans lesquels ils évoluent. Pour cela, AMBRE-enseignant leur permet de générer des problèmes, de créer des séquences d'apprentissage (ensemble d'activités destinées aux apprenants) et des thèmes d'exercices, mais aussi de paramétrer l'environnement élève, de créer les listes d'élèves des classes et de distribuer le travail (séquences et exercices) aux classes ou aux élèves.

Le **paramétrage de l'environnement** élève consiste en la personnalisation par l'enseignant de l'interface du logiciel élève : choix des couleurs et de la langue principalement. AMBRE-enseignant doit également permettre aux enseignants d'établir la **liste des élèves** de leur(s) classe(s).

Un certain nombre d'exercices, créés par les concepteurs, seront fournis avec le logiciel élève. Mais afin de permettre aux enseignants qui le désirent de faire travailler leurs élèves sur des problèmes dont ils souhaitent préciser les caractéristiques, AMBRE-enseignant comporte un module de **génération de problèmes**. Cette génération doit être un plus pour les enseignants et non une obligation. Ce passage semble toutefois nécessaire pour augmenter et diversifier la batterie d'exercices de l'EIAH AMBRE, et adapter les énoncés au public à former. Générer un problème à résoudre dans l'EIAH AMBRE consiste à proposer un énoncé en langue naturelle ainsi qu'une description du problème compatible avec le résolveur utilisé par le logiciel élève. Du point de vue de l'enseignant, il s'agit de préciser un certain nombre de caractéristiques que doit comporter le problème. La génération du problème peut être plus ou moins automatisée selon le choix de l'enseignant : il peut préciser toutes les caractéristiques du problème, certaines seulement, ou au contraire aucune.

Nous voulons également offrir la possibilité à l'enseignant de **créer des séquences** d'apprentissage (ensemble de problèmes à résoudre dans une ou plusieurs séances d'utilisation du logiciel) en utilisant le matériel pédagogique (les problèmes) qu'il a créé. Cette fonctionnalité lui permettra de mettre en place sa stratégie, sa démarche pédagogique, en intégrant comme il le souhaite les problèmes de son choix dans une séquence. On propose pour cela à l'utilisateur deux manières de créer une séquence : manuelle ou automatisée. Pour la création manuelle, l'enseignant sélectionne de façon chronologique ou de façon aléatoire les exercices qu'il souhaite intégrer dans la séquence qu'il crée. Il peut par exemple choisir un ordre de présentation des problèmes en fonction de leur niveau de difficulté. L'enseignant peut ensuite définir le comportement du logiciel élève pour les exercices de la séquence : nombre d'essais autorisé par étape de l'exercice (et que faire si ce nombre est atteint ?), nombre d'accès à la vérification ou à l'aide autorisés pendant l'exercice, fonctionnement du diagnostic (toujours diagnostiquer ou laisser l'apprenant se tromper à certaines étapes de la résolution), etc. On définit par défaut ce comportement pour tous les exercices de la séquence. Toutefois, si l'enseignant souhaite intégrer une progression dans la séquence, il a la possibilité de définir un comportement différent en fonction des exercices (il peut ainsi par exemple mettre un diagnostic maximum en début de séquence, et le diminuer au fur et à mesure). Pour la création automatisée d'une séquence, l'enseignant ne choisit que le(s) dossier(s) d'exercices dans lequel le système prendra les exercices, ainsi que le nombre d'exercices qu'il souhaite voir figurer dans la séquence. Charge ensuite au système de choisir aléatoirement les exercices. L'enseignant peut ensuite s'il le souhaite modifier les propositions du système (supprimer, ajouter ou remplacer un problème), ou ordonner les exercices différemment, définir un comportement particulier, etc.

AMBRE-enseignant doit également permettre de **distribuer le travail** aux élèves, c'est-à-dire associer à l'ensemble de la classe ou à chaque élève une ou plusieurs séquences. L'enseignant peut en effet donner les mêmes séquences à tous les élèves ou choisir d'individualiser l'enseignement en proposant des séquences spécifiques à certains élèves.

Afin que l'enseignant puisse entièrement adapter les problèmes générés à son environnement, au contexte dans lequel ses élèves évoluent, AMBRE-enseignant pourra à terme lui permettre de **gérer les traits de surface** qui seront utilisés dans les exercices. Les traits de surface sont les éléments qui permettent d'habiller les problèmes mais qui ne sont pas pertinents pour la résolution. L'enseignant pourra donc créer, modifier et supprimer des thèmes d'exercices, ainsi que les traits de surface liés aux thèmes (par exemple des objets et des personnages spécifiques, les actions qui y sont associées, etc).

4. Un générateur de problèmes pour AMBRE

4.1. Quel type de générateur pour AMBRE ?

Dans la plupart des EIAH fondés sur la résolution de problèmes, les exercices proposés aux apprenants sont issus d'une bibliothèque prédéfinie. Leur diversité et leur nombre en sont alors limités. Une solution répondant à ce problème consiste en l'utilisation de générateurs de problèmes.

Les premiers systèmes génératifs ont vu le jour dans les années 70. Certains générateurs sont entièrement automatiques, les problèmes étant générés selon les besoins identifiés dans le modèle de l'apprenant [BURTON 82] [MITROVIC et al. 96], ou en vue de créer une batterie de tests. Ce type de systèmes n'offrant aucune interaction avec l'enseignant, ce n'est pas l'approche que nous avons retenue pour AMBRE-enseignant.

À l'inverse des générateurs automatiques, les générateurs manuels ne construisent pas eux-mêmes les énoncés des problèmes, ni leur solution. C'est à l'enseignant que revient cette tâche. Selon l'approche des systèmes auteurs, ces environnements permettent de générer des exercices de types divers. L'enseignant construit chaque énoncé, en précisant la ou les réponses attendues. Le système ne possède aucune connaissance de l'exercice qu'il propose, les réponses sont préenregistrées. Le système n'est donc pas apte à résoudre les problèmes qu'il permet de créer et n'offre pas de fonctionnalité d'aide ou de diagnostic des réponses de l'apprenant. Le seul diagnostic possible consiste à indiquer à l'apprenant s'il a répondu juste ou faux aux questions posées. Quant à l'aide, elle doit être préalablement définie par l'enseignant. Cette approche, qui laisse une totale liberté à l'enseignant, ne nous convenait toutefois pas dans le cadre du projet AMBRE. En effet, un EIAH AMBRE est fondé sur un système à base de connaissances qui s'appuie sur un résolveur de problèmes et permet de fournir à l'apprenant de l'aide, un diagnostic de ses réponses ainsi que des explications sur ses erreurs [DUCLOSSON 04]. Les problèmes posés à l'apprenant doivent donc être compréhensibles par le résolveur afin que l'EIAH puisse fournir aide, diagnostic et explications à l'apprenant. Par conséquent, il n'est pas possible de laisser l'enseignant saisir un énoncé en langue naturelle sur lequel le résolveur ne pourrait pas raisonner.

Nous avons donc choisi une approche relevant des générateurs dits semi-automatiques, qui construisent eux-mêmes les énoncés des problèmes, mais en laissant intervenir l'utilisateur dans le processus de création. L'enseignant a donc la possibilité avec de tels systèmes d'influer sur les problèmes qui seront générés, en spécifiant un ensemble de contraintes sur l'exercice à générer. Par exemple, le système CEP [GIROIRE 89] permet de créer des exercices liés au monde réel, nécessitant l'introduction de directives par l'utilisateur et permettant de produire des énoncés en langue naturelle. De

même, le système SYGEP [PECEGO 98] est un système à base de connaissances qui crée des énoncés de problèmes et leur solution sous forme textuelle pour des domaines utilisant des dispositifs ou des expérimentations qui sont à la base du problème et donnent lieu à des questions (circuits électriques, expériences de mécanique…). L'utilisateur (humain ou système pilote) peut fixer ses objectifs pédagogiques en spécifiant un ensemble de contraintes (tel que le niveau de difficulté) qui calibreront le problème généré.

Pour AMBRE-enseignant, le générateur de problèmes doit permettre à l'enseignant d'adapter les problèmes produits à son contexte d'apprentissage et à sa démarche pédagogique. Il est donc important que celui-ci puisse caractériser les exercices à générer, c'est-à-dire saisir un ensemble de contraintes qui spécifieront ces problèmes. Les problèmes étant construits par le système à partir de ces contraintes, le résultat de la génération sera non seulement un énoncé en langue naturelle, mais aussi un modèle du problème qui sera compréhensible par le résolveur.

Le projet AMBRE a pour objectif la conception d'EIAH pour apprendre des méthodes dans des domaines variés. Il est donc important que l'approche choisie pour la génération de problèmes soit indépendante du domaine. Nous avons donc essayé de concevoir une architecture indépendante du domaine, à laquelle il faut fournir des bases de connaissances spécifiques à un domaine donné pour obtenir un générateur de problèmes pour ce domaine.

Nous présentons dans la partie suivante l'outil de génération de problèmes tel que nous l'avons élaboré pour le domaine des problèmes additifs, puis nous décrivons l'architecture qui permet de réaliser un tel générateur de problèmes.

4.2. L'environnement de génération de problèmes pour l'enseignant

Notre objectif concernant le module de génération de problèmes est de permettre à un enseignant de créer ses propres problèmes et de les adapter au public à qui il les destine. De plus, il nous semblait important de pouvoir lui permettre de construire avec exactitude un énoncé souhaité, ou au contraire qu'il puisse demander au système d'en construire un automatiquement. Pour arriver à une telle souplesse, nous avons décidé d'utiliser l'approche de la génération à partir de contraintes : l'enseignant définit les contraintes qu'il souhaite voir vérifiées par le problème que le système va construire.

Afin que la tâche de création de problèmes par l'enseignant soit simplifiée, nous voulions lui éviter d'avoir à créer un à un les nouveaux énoncés. Nous avons donc décidé qu'un même jeu de contraintes pourrait permettre la création de plusieurs exercices. Moins il y aura de contraintes, plus les problèmes générés seront variés.

Nous avons conçu et développé l'outil de génération de problèmes destiné à l'enseignant pour le domaine des problèmes additifs sur lequel porte l'EIAH AMBRE-add destiné aux élèves de l'école primaire. Les problèmes de ce domaine décrivent une situation concrète, par exemple un jeu de billes : « Alex avait 32 billes. À la fin de la récréation, il en a 45. Combien a-t-il gagné de billes pendant la récréation ? ». Ce domaine a été largement étudié en didactique des mathématiques, et plusieurs classifications ont été établies. Celle que nous utilisons dans AMBRE-add est présentée dans [DUCLOSSON 04].

Pour le domaine des problèmes additifs, les contraintes que l'enseignant peut définir relèvent de quatre catégories différentes : traits de structure, traits de surface, valeurs et complication.

La **structure** d'un problème à générer correspond à la classe du problème. Cette classe est définie par plusieurs attributs qui peuvent ou non être fixés.

Les **traits de surface** sont les éléments qui permettent d'habiller les énoncés produits. L'enseignant peut préciser certains éléments de cette catégorie, par exemple les thèmes, les objets et les personnages pour les problèmes additifs.

Pour les problèmes additifs, l'enseignant peut choisir les **valeurs** des données qui seront utilisées dans les problèmes. Il peut définir un intervalle global de valeurs pour les éléments et l'écart (également par un intervalle) souhaité entre les deux valeurs. Une autre possibilité consiste à fixer une valeur ou un intervalle pour chaque élément connu de l'énoncé. L'enseignant peut également, s'il veut simplifier les opérations, interdire l'utilisation de la retenue dans les calculs.

La **complication** concerne toutes les options qui permettront de compliquer l'énoncé du problème pour l'adapter au niveau des élèves. Cette partie a nécessité une collaboration étroite avec les enseignants, afin de connaître leurs besoins. Pour les problèmes additifs, l'environnement propose différents types de difficultés à intégrer dans les exercices : des complications de la langue (difficulté du vocabulaire employé et des tournures des phrases) et de l'énoncé lui-même (écriture des nombres en lettres, modification de l'ordre des propositions de l'énoncé, ajout de phrases inutiles à la résolution, introduisant éventuellement dans le problème des données non pertinentes).

Précisons qu'aucune des contraintes citées précédemment n'est obligatoire pour la création d'exercices. Les contraintes non spécifiées par l'enseignant seront définies aléatoirement par le système. L'enseignant peut de plus attribuer aux problèmes créés un nom, un niveau (par exemple CE1 1er trimestre) et une description, afin de les retrouver plus facilement pour la création de séquences.

Ces quatre catégories de contraintes définies pour les problèmes additifs ne seront pas réutilisables telles quelles pour un autre domaine d'application de AMBRE. Néanmoins, il est certain que les catégories « traits de structure » et « traits de surface » seront présentes dans tous les domaines. La catégorie « valeurs » sera uniquement présente pour les domaines numériques, alors que la catégorie « complication » sera probablement utile dans tous les domaines. Les contraintes figurant dans chacune de ces catégories dépendront bien évidemment du domaine.

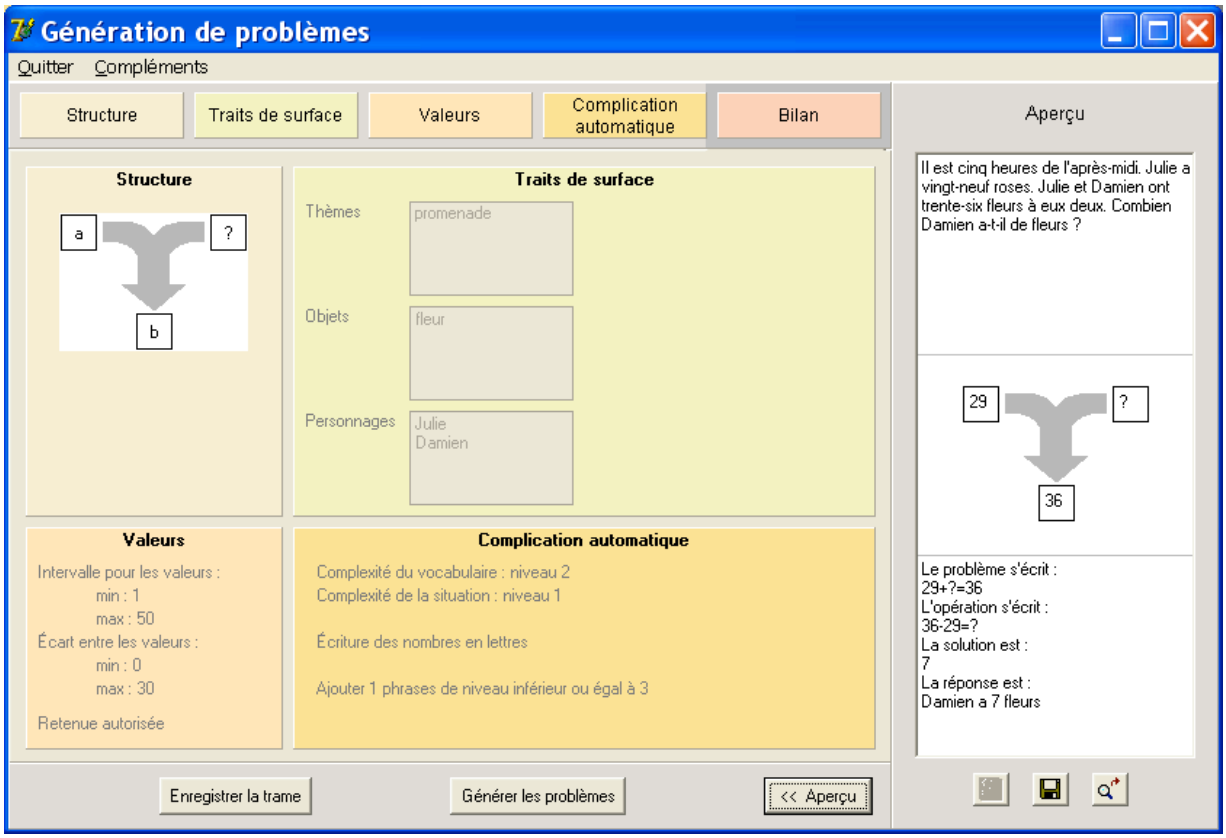

Figure 1. *L'écran de bilan du module enseignant de AMBRE-add.*

La Figure 1 présente l'écran de bilan du module enseignant de AMBRE-add. Cet écran résume les contraintes définies par l'enseignant pour les quatre catégories (structure, traits de surface, valeurs et complication) et présente un exemple de problème qui pourra être généré à partir de ces contraintes. Le schéma de la partie "structure" sert à représenter la classe du problème.

4.3. L'architecture GenAMBRE

Le processus de génération de problème que nous avons mis en œuvre dans l'architecture GenAMBRE prend en entrée le jeu de contraintes spécifié par l'enseignant et fournit en sortie deux éléments : l'énoncé du problème en langue naturelle destiné aux apprenants et une formulation de l'énoncé appelée modèle descriptif du problème, destinée au résolveur de problèmes de AMBRE, issu de l'architecture SYRCLAD [GUIN-DUCLOSSON 99].

L'architecture du générateur de problèmes pour AMBRE est présentée en Figure 2 et chacun de ses constituants est présenté dans la suite de l'article. Pour un domaine D (par exemple celui des problèmes additifs), il faut définir les cinq bases de connaissances du « niveau domaine » en utilisant les formalismes de représentation des connaissances indépendants du domaine imposés par le « niveau génération ». Le processus de génération du problème s'effectue en deux temps : le système construit un modèle de génération du problème, puis construit l'énoncé en langue naturelle ainsi que le modèle descriptif du problème. Ces deux processus sont indépendants du domaine. Associés aux bases de connaissances du domaine D, ils forment un générateur de problèmes pour le domaine D : GenAMBRE-D.

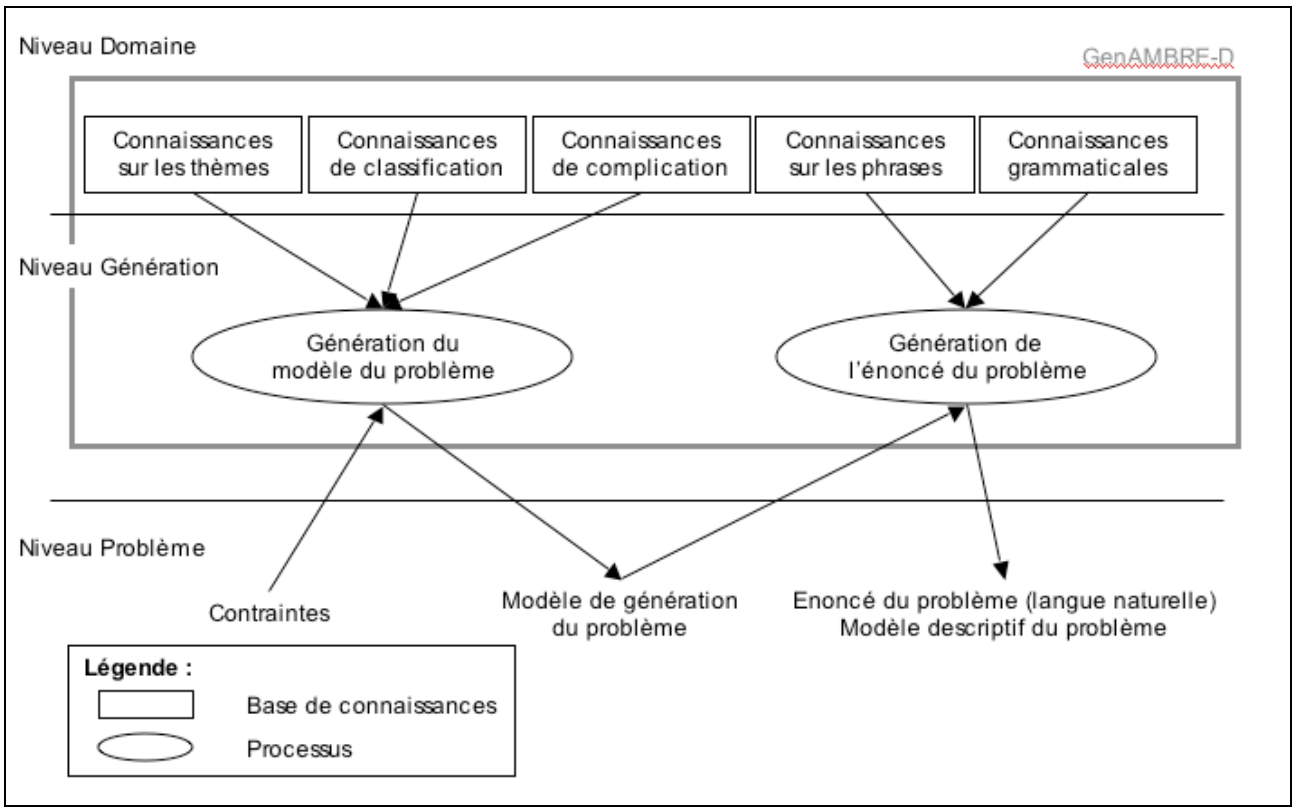

Figure 2. *L'architecture GenAMBRE*

Connaissances de classification Dans chaque domaine d'application, un expert fournit au résolveur SYRCLAD, et donc par extension au générateur GenAMBRE, un graphe de classification de problèmes. Ce graphe est utilisé dans le résolveur pour classer le problème. C'est une hiérarchie de classes dépendante du domaine, mais dont la représentation est la même quel que soit le domaine.

Connaissances sur les thèmes Pour pouvoir générer un problème, nous avons besoin de connaître le thème concerné et les traits de surface qui lui sont associés (par exemple les objets, personnages et actions pour le domaine des problèmes additifs). De telles connaissances sont dépendantes du domaine pour lequel elles sont utilisées, mais leur représentation est indépendante du domaine. Les connaissances sur les thèmes sont actuellement fournies par l'expert, elles pourront à terme être créées par l'enseignant lui-même, via le module de gestion des traits de surface de AMBRE-enseignant.

Connaissances de complication Pour pouvoir compliquer un exercice et donc son énoncé, il est nécessaire de fournir des connaissances de complication au système de génération. Pour les problèmes additifs, compliquer un énoncé revient principalement à changer l'ordre des phrases et à ajouter des phrases inutiles. Les connaissances de complication permettent alors de répondre aux questions suivantes : comment et dans quel cas est-il possible de changer l'ordre des phrases utiles du problème ? Quelles phrases inutiles peut-on ajouter aux problèmes et où les placer ?

Processus de génération du modèle du problème À partir des trois bases de connaissances décrites précédemment et des contraintes entrées par l'enseignant, le système doit générer ce que nous appelons le modèle de génération du problème. Ce modèle est en fait un modèle descriptif étendu, puisqu'il contient le modèle descriptif du problème mais précise également quelques informations absentes du modèle descriptif, tels que la classe du problème ou encore son thème. Pour construire ce modèle, le processus complète les contraintes définies par l'enseignant, en choisissant notamment aléatoirement des « valeurs » pour celles qui n'ont pas été précisées.

Connaissances grammaticales L'expert du domaine doit fournir une grammaire au système de génération, c'est-à-dire un ensemble de structures de phrases qui pourront être utilisées dans le domaine. Dans le cadre des problèmes additifs, cette grammaire permet de générer des phrases affirmatives et interrogatives. Les unités syntaxiques de base sont les groupes sujets, les verbes et les compléments. Cette grammaire est dépendante du domaine dans le sens où, pour un domaine donné, on ne prévoit pas de pouvoir créer toutes les phrases de la langue française. Par exemple, pour le domaine des problèmes additifs, nous n'utilisons pas de propositions relatives ni d'adjectifs. Lorsque nous appliquerons GenAMBRE à d'autres domaines, nous pourrons probablement avoir une seule grammaire dont nous spécifierons pour chaque domaine les parties à utiliser.

Connaissances sur les phrases Les énoncés générés, notamment la structure de leurs phrases, dépendent de la classe à laquelle le problème appartient. Il est donc nécessaire pour pouvoir générer l'énoncé en langue naturelle, de savoir

quelles structures de phrases (de la grammaire) pourront être utilisées pour le problème à générer. Ce sont ces informations que contiennent les connaissances sur les phrases. Elles permettent d'associer à la classe du problème les structures de phrases utilisables, et les éléments du problème qui seront associés à ces structures.

Processus de génération de l'énoncé du problème Pour générer un énoncé en langue naturelle, le processus utilise les connaissances sur les phrases et la grammaire du domaine, ainsi que le modèle de génération du problème créé auparavant. Les connaissances sur les phrases permettent au système de prendre les décisions d'ordre conceptuel (« décider quoi dire ») : on spécifie ici la structure des phrases qui seront générées en précisant pour chaque phrase les éléments qui la composeront (issus du modèle du problème : objets, personnages…). Une fois les décisions d'ordre conceptuel prises, le processus passe à l'étape de génération de texte, et met donc en place différents mécanismes linguistiques : traitements syntaxiques (« décider comment le dire »), lexicaux et morphologiques (« décider comment l'écrire »).

4.4. Exemples de problèmes générés

Pour les contraintes données par l'enseignant dans l'exemple de la figure 1, le générateur de problèmes lui propose l'énoncé suivant : « Il est cinq heures de l'après-midi. Julie a vingt-neuf roses. Julie et Damien ont trente-six fleurs à eux deux. Combien Damien a-t-il de fleurs ? ». Si l'on demande au système de générer un autre problème à partir des mêmes contraintes, il pourra proposer : « Damien a dix billes. Il a dix-sept fleurs. Si Damien et Julie mettent leurs fleurs ensemble, ils en ont trente-cinq. Combien Julie a-t-elle de fleurs ? ». Ces deux énoncés sont proches parce que l'enseignant a dans cet exemple défini des contraintes très précises (Julie et Damien qui ont des fleurs).

Si l'enseignant choisit la même classe de problème dans l'onglet "structure" que dans l'exemple précédent, mais ne définit aucune contrainte supplémentaire dans les onglets "traits de surface", "valeurs" et "complication", le générateur produira des énoncés plus variés, tels que ceux ci-dessous :

• Damien est dans la cour de récréation avec Paul. Il a 32 billes vertes. Si Damien et Paul mettent leurs billes vertes ensemble, ils en ont 46. Cherche le nombre de billes vertes de Paul.

• Il est 11h. Sophie a 25 billes. Sophie et Romain en ont 38 à eux deux. Calcule le nombre de billes de Romain.

• Trouve le nombre de roses rouges de Sabrina, sachant qu'en ajoutant les roses rouges de Sabrina, Julie et Sabrina en ont 40 à elles deux, qu'elles ont 10 ans et qu'elles sont amies, et que Julie a 7 roses rouges.

• La semaine dernière, c'était l'anniversaire de Jean, maintenant il a 11 ans. Il a 9 billes jaunes. Quand Jean et Julie mettent leurs billes jaunes ensemble ils en ont 37. Trouve le nombre de billes jaunes de Julie.

5. Conclusion, perspectives

Dans cet article, nous avons présenté comment nous avons conçu un module destiné à l'enseignant pour l'EIAH AMBRE-add, pour permettre à l'enseignant d'adapter l'EIAH au contexte d'apprentissage et à sa démarche pédagogique. En adoptant une démarche générique, nous avons identifié en concertation avec des enseignants les fonctionnalités que doit posséder un module enseignant pour un EIAH AMBRE (AMBRE-enseignant). Nous avons ainsi prévu que l'enseignant puisse paramétrer l'environnement, générer des problèmes de son choix, créer des séquences d'apprentissage adaptées à ses élèves en choisissant les problèmes et le comportement de l'EIAH, distribuer ces séquences à ses élèves, et créer de nouveaux thèmes d'exercices.

À l'exception de la création de thèmes d'exercices que nous souhaitons traiter prochainement, ces fonctionnalités ont été implémentées pour le domaine des problèmes additifs. Nous avons pour cela conçu un système de génération de problèmes dont l'architecture est indépendante du domaine. Même si nous avons intégré des enseignants à la conception de AMBRE-enseignant, il nous faut à présent évaluer le module enseignant de AMBRE-add en situation réelle auprès d'un nombre significatif d'enseignants. Enfin, nous devons également valider les fonctionnalités définies pour AMBRE-enseignant ainsi que la généricité de l'architecture GenAMBRE en développant un module enseignant pour un EIAH AMBRE portant sur un autre domaine.

6. Références

[BOUHINEAU 96] Bouhineau, D., Channac, S., « La programmation logique par contraintes pour l'aide à l'enseignant », Intelligent Tutoring Systems, Canada, 1996.

[BURTON 82] Burton, R.R., « Diagnosing bugs in a simple procedural skill». Intelligent Tutoring Systems, Academic Press, London, 1982.

[DAVID et al. 02] David, J.P., Guilloux, C., Flament, A., « A learning objects generator with xml-xslt technology». Conférence TICE2002, session ARIADNE, 2002.

[DUCLOSSON 04] Duclosson, N., « Représentation des connaissances dans l'EIAH AMBRE-add », Conférence TICE'2004, Compiègne, 20-22 octobre 2004, p. 164-171.

[GIROIRE 89] Giroire, H., Un système à base de connaissances pour la génération d'exercices dans des domaines liés au monde réel, Thèse de l'Université Paris 6, 1989.

[GUIN-DUCLOSSON 99] Guin-Duclosson, N., « SYRCLAD : une architecture de résolveurs de problèmes permettant d'expliciter des connaissances de classification, reformulation et resolution », Revue d'Intelligence Artificielle, vol. 13-2, Hermès, 1999.

[GUIN-DUCLOSSON et al. 02] Guin-Duclosson N., Jean-Daubias S., Nogry S. : « The AMBRE ILE: How to Use Case-Based Reasoning to Teach Methods », ITS'2002 proceedings, Springer, 2002, Lecture Notes in Computer Science vol. 2363, p. 782-791.

[JEAN-DAUBIAS 03] Jean-Daubias, S, « Exploitation de profils d'apprenants », Conférence EIAH'2003, Strasbourg, avril 2003, p. 535-538.

[JEAN-DAUBIAS 04] Jean-Daubias, S., « De l'intégration de chercheurs, d'experts, d'enseignants et d'apprenants à la conception d'EIAH », Conférence TICE'2004, Compiègne, 20-22 octobre 2004, p. 290-297.

[LEROUX 97] Leroux, P., « ROBOTEACH : un assistant pédagogique logiciel dédié à l'alphabétisation en technologie », Actes du cinquième Colloque International sur la Robotique Pédagogique,, Montréal, Canada, 12-15 août, 1997, p. 45-63.

[LEROUX 02] Leroux, P., Machines partenaires des apprenants et des enseignants – Étude dans la cadre d'environnements supports de projets pédagogiques, Habilitation à Diriger des Recherches en Informatique de l'Université du Maine, Le Mans, 2002.

[MITROVIC et al. 96] Mitrovic, A., Stoinemov, L., Djordjevic-Kajan, S., « INSTRUCT: Modelling Students by asking questions », User Modelling and User-Adapted Interaction, vol.6-4, p. 273-301.

[NICAUD et al. 04] Nicaud J.F., Bouhineau, D., Chaachoua, H., « Mixing microworld and cas features in building computer systems that help students learn algebra », International Journal of Computers for Mathmatical Learning 9, 2004, p. 169-211.

[NOGRY et al. 02] Nogry, S., Jean-Daubias, S., Guin-Duclosson, N., « La psychologie cognitive au service de la conception de l'environnement d'apprentissage AMBRE », Conférence TICE'2002, Lyon, 13-15 novembre 2002, p. 195-202.

[NOGRY et al. 04] Nogry, S., Jean-Daubias, S., Duclosson, N., « ITS Evaluation in Classroom: The Case of AMBRE-AWP », ITS'2004 proceedings, Springer, 2004, Lecture Notes in Computer Science vol. 3220, p. 511-520.

[PECEGO 98] Pecego, G., SYGEP, un Système de Génération d'Énoncés de Problèmes dans des domaines variés, Thèse de l'Université Pierre et Marie Curie, Paris VI ,12 juin 1998.

[VIVET 90] Vivet, M., « Uses of ITS: Which role for the teacher? », New Directions for Intelligent Tutoring Systems, NATO ASI series, Vol. F91, Springer-verlag, Sintra, 1990.