Université de Montréal
Faculté des Études Supérieures

Développement de cours pour un environnement distribué de formation à distance

Par :
El Bachir Gardouh

Département d'informatique et de recherche opérationnelle
Faculté des art et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de
Maître ès Sciences (M.Sc.)
en informatique
Octobre, 2000

Université de Montréal
la Faculté des études supérieures

Ce mémoire intitulé:

Développement de cours pour un environnement distribué de formation à distance

présenté par:
El Bachir Gardouh

A été évalué par un jury composé des personnes suivantes :

President-rapporteur : Houari Sahraoui

Directeur de recherche : Claude Frasson

Membre du jury : Esma Aïmeur

TABLE DES MATIÈRES

Liste des Figures

REMERCIEMENTS

Je tiens à remercier vivement le professeur Claude Frasson, mon directeur de recherche, pour m'avoir proposé ce sujet, pour sa disponibilité malgré ses nombreuses occupations, pour ses directives, ses conseils et son appui financier.

Je me tiens à cœur de remercier plus profondément mes chers parents qui ont fait de moi ce que je suis aujourd'hui. Que dieu, le tout puissant, leur donne santé et langue vie.

Des remerciements particuliers à ma très chère sœur Fatiha pour son soutien morale et ses encouragements tout au long de ma recherche.

Je remercie encore beaucoup mon frère El Bouazzaoui pour ses précieux conseils tout au long de mon chemin.

Mes remerciements aussi à mes chers amis, Abdellah, Abdessalam, Hicham, Kamal, Mustapha pour leurs encouragements et pour le climat familial et chaleureux qu'on a vécu ensemble tout le temps.

J'aimerai remercier tous les membres de l'équipe de Virtuel Age pour leur participation et leur dialogue constructif.

A mes chers parents, chers frères et sœurs

CHAPITRE I

INTRODUCTION

1.1 Les Systèmes Tutoriels Intelligents (STI)

Les STI ont pour objet de réaliser, à l'aide d'un ordinateur, un enseignement individualisé. La construction de tels systèmes nécessite une bonne étude, aussi bien du processus d'enseignement et/ou d'apprentissage, que du comportement des acteurs impliqués dans le ce processus (enseignants, analystes de la formation, apprenants). Il en découle plusieurs axes de recherches impliquant à la fois des aspects liés à l'éducation, à l'informatique et à la psychologie. L'idée générale de ces recherches est d'extraire les connaissances de ces trois domaines et de les reproduire dans un ordinateur ou alors de les faire produire par un ordinateur. Nous savons cependant que ces expertises sont complexes et requièrent de l'expertise humaine.

Du point de vue de l'informatique, les STI constituent une des applications de l'intelligence artificielle (IA) qui est un domaine assez récent ayant vu le jour vers la fin des années 50 (plus précisément en 1956), suite aux travaux McCarthy, Minsky, Shannon, Newell et Simon qui se proposaient d'étudier la possibilité de réaliser des programmes d'ordinateur doués d'intelligence. Ainsi, l'intelligence artificielle se propose d'étudier d'une part les mécanismes de l'intelligence avec l'ordinateur comme moyen de simulation et, d'autre part, de doter la machine de capacités habituellement attribuées à l'intelligence humaine : acquisition de connaissances, prise de décision, raisonnement, perception... C'est ce second point de vue qui est le plus couramment rencontré, il

consiste à émuler par un programme d'ordinateur des comportements intelligents, sans pour autant reproduire le fonctionnement correspondant de l'être humain. Ces deux approches restent cependant largement complémentaires, dans la mesure où une meilleure connaissance des mécanismes humains permet d'améliorer les performances des systèmes informatiques. C'est d'ailleurs ce qui a motivé les chercheurs dans le domaine des STI à s'intéresser aux types d'intelligence (expertises) impliqués dans l'enseignement d'un domaine, d'où leur intérêt pour l'éducation.

Dans le domaine de l'éducation, des théories relatives au processus d'enseignement (théories pédagogiques) ont été développées. Ces théories visent à étudier les méthodes et les moyens à mettre en œuvre pour favoriser l'acquisition de connaissances par un apprenant, suite à un enseignement. Il est évident que l'intégration ou la considération de ces théories dans un STI au moyen de l'IA, ne peut que contribuer à augmenter leur efficacité.

Enfin, l'une des difficultés dans un processus d'enseignement-apprentissage est la compréhension du comportement de l'apprenant. Un des apports de la psychologie cognitive se reflète à ce niveau. Il s'agit de développer des théories permettant d'analyser le raisonnement d'un étudiant dans un processus d'apprentissage, afin de produire un enseignement efficace, en employant les méthodes les plus susceptibles de favoriser cet apprentissage.

Depuis le début des années 70, des STI ont été développés dans le but d'expérimenter l'intégration de théories reliées aux domaines précédents (IA, sciences de l'éducation, psychologie cognitive) dans une machine. Les systèmes SCHOLAR [Carbonell 70], SOPHIE [Brown 75], BIP [Barr 76], BUGGY [Brown 78], WUSOR [Carr 77], Geometry-Tutor [Anderson 83], LISP-Tutor [Anderson 84, Reiser 85] STEAMER [Hollan 84], GUIDON [Clancey 82a], Word-Tutor [Imbeau 90] et autres, ont permis de comprendre la complexité reliée au développement d'un STI ; en particulier les problèmes de modélisation de l'apprenant, etc. Quelques solutions ont été apportés à certains de ces problèmes ; elles se sont, pour la plupart, concentrées sur l'étude même du processus d'apprentissage, en faisant souvent abstraction de l'étude de l'objet d'enseignement ou d'apprentissage, c'est à dire la matière à enseigner et les moyens à mettre en œuvre pour son enseignement. Or, cet aspect reste essentiel dans le domaine de

l'éducation qui considère la spécification et l'organisation de la matière à enseigner (encore appelée **curriculum**), comme l'entrée principale dans un système d'enseignement [Finch 86]. La psychologie (humaniste, béhavioriste ou cognitive) reconnaît la nécessite de créer un contexte riche et varié pouvant supporter la planification et le déroulement de l'enseignement [Grippin 84, Tardif 92].

Le concept de curriculum a été abordé dans quelques travaux [Barr 76, Halff 88, McCalla 90] bien qu'il reste peu considère dans les STI. Halff (1988) considère que le but du curriculum dans les STI est formuler une représentation des matériels et de sélectionner et ordonner les activités de formation particulières à partir de cette représentation. Ainsi, il oriente la représentation du curriculum sur le matériel. McCalla (1990) donne une définition beaucoup plus cognitive, selon laquelle, le curriculum dans un STI représente la sélection et l'ordonnancement des connaissances en vue de réaliser des buts d'enseignement appropriés au contexte courant et à l'apprenant courant. Il met l'accent sur le fait qu'un curriculum doit être flexible, évolutif et doit s'adapter aux besoins de l'apprenant et au déroulement de l'enseignement. D'autres travaux tels que ceux de Wipond et Jones [Wipond 88], Merrill [Merrill 91] abordent des problèmes de construction d'environnement destinés aux enseignants humains, pour le développement de cours.

S'il est vrai que l'accent doit être mis sur les connaissances que le STI veut faire acquérir à un apprenant, il n'en demeure pas moins que l'acquisition de ces connaissances se fait à travers une variété d'activités d'apprentissage et d'enseignement, véhiculées par des matériels (ressources didactiques). Ainsi, le curriculum, tout en se focalisant sur les connaissances relatives à une matière donnée, doit aussi se préoccuper des ressources didactiques nécessaires à l'acquisition de ces connaissances dans des situations d'enseignement ou d'apprentissage donné. Par exemple, l'acquisition d'une connaissance concernant la résolution d'un système d'équations linéaires nécessite la réalisation d'un certain nombre d'activités (enseignement de concepts associés à un système d'équations linéaires, résolution de problèmes ou d'exercices sur les systèmes d'équations linéaires...). Cette réalisation n'est possible que si les ressources didactiques concernées (contenu, exercices...) sont disponibles. Naturellement, la sélection de l'une ou de l'autre ressource est fonction du déroulement de l'enseignement et des besoins de l'apprenant.

Le curriculum doit donc pouvoir créer et organiser le contexte qui sera utilisé par les autres modules d'un STI dans le but de faire réaliser des apprentissages. Gauthier Imbeau et Girard [Gauthier 89, Imbeau 90, Girard 91], proposent une approche incluant les objectifs d'enseignement et les éléments du domaine a enseigner dans une hiérarchie. Cette approche reste limitée à cause d'un manque de flexibilité qui vient du fait que les niveaux contrôle (processus d'enseignement) et représentation (représentation de la matière enseignée) sont mélangés. L'idée de la modélisation du curriculum par un automate d'états finis [Gauthier 89, Imbeau 90] ou par un réseau de Petri [Ferraris 84] entraîne un mélange entre l'aspect statique et l'aspect dynamique (contrôle), qui rend flou la limite entre le curriculum et le planificateur. Ceci perturbe l'idée selon laquelle ces deux derniers sont considérés comme des objets qui communiquent entre eux. Il en est de même pour la plupart des approches qui ne permettent pas de bâtir un curriculum pouvant supporter l'enseignement d'un cours complet, vu que l'emphase est souvent mise plutôt sur un seul type de connaissance.
Par ailleurs, il se pose un problème de construction d'outils intelligents génériques pouvant aider à l'automatisation de la construction de curriculums en particulier et de l'enseignement en général ; des idées ont été avancées dans ce sens [Halff 93, Merril 93, Tennyson 93, Gagné 93].

Le modèle CREAM (Curriculum REpresentation and Acquisition Model) [Nkambou 96] a fait l'objet d'implémentation et d'expérimentation dans plusieurs cours au sein du projet SAFARI (Code de la route, Salle des soins intensifs, ...). Au cours de ces expériences plusieurs faiblesses qui limitent la pleine exploitation de tout le potentiel de ce modèle :

- creation difficile et complexe du contenu de la matiere.
- manque de strategie pour la progression de l'apprenant.
- manque de reutilisation de ressources ce qui engendre un cout de production elevé.
- manque de d'outils de restructuration et d'exploitation de ressources déjà existantes
- interface non conviviale.

- gestion non sécuritaire des données

1.5 Objectif de la recherche

L'objectif de cette recherche est axé principalement sur la résolution des problèmes suivants :

- la restructuration des connaissances déjà développées dans des cours linéaires (format HTML, Texte, Vidéo ou autres). On tente de doter le module expert d'une architecture décentralisée dans le but de permettre la réutilisation tel que définie dans le paradigme objet.
- La simplification des modèles déjà implémentés dans le cadre du projet SAFARI, pour les rendre plus accessible et plus productifs dans un environnement multi-utilisateurs.
- le support de la formation à distance qui limitait l'exploitation du projet SAFARI à une grande échelle. Dans ce sens, on tente de porter un STI traditionnel sur le WEB. Profitant ainsi de la prolifération de la technologie JAVA et les objets distribués qu'offre celle ci. Le problème de suivi de la progression de l'apprenant et la gestion du modèle cognitif de ce dernier pour la formation en question par le biais d'un agent pédagogique est aussi un problème au centre de notre sujet de recherche.

Cette recherche a été développée et financée au sein de l'entreprise Virtuel Age.

1.6 Plan du mémoire

L'aspect de modélisation de l'expertise du domaine et de l'expertise pédagogique est un élément très important dans la construction des STI. Dans le chapitre 2 "Modélisation des connaissances dans les STI", nous allons voir plus en détail comment la modélisation des connaissances est abordée dans différents STI. Nous discuterons des différentes approches de modélisation et de représentation dans ces STI ainsi que les différentes stratégies d'enseignement.
Cette analyse nous permet de mieux detailler notre choix et notre approche dans le chapitre 3 "Conception et Modélisation de la Matière ". Cette approche a l'avantage de combiner la représentation des connaissances du domaine à enseigner et celle de son expertise pédagogique. De plus, elle permet de supporter la génération de cours, la planification de l'enseignement et le déroulement d'un cours complet. Notre approche est basée sur les avantages du modèle CREAM, tout en simplifiant la complexité et les détails qui ont fait du modèle CREAM un modèle difficile à suivre dans un cadre pratique et opérationnel.

Dans le chapitre 4 "Environnement de Développement de Cours" nous verrons comment nous avons intégré cette approche dans notre travail avec les modifications et améliorations apportées dans le but d'aboutir à un environnement complet et opérationnel de développement de cours.

Dans le chapitre 5, on exposera notre implantation du tuteur dans un contexte WEB. Nous discuterons aussi de l'implantation de l'agent pédagogique dans un environnement distribué basé sur la technologie RMI (Remote Method Invocation).

Nous terminerons avec une conclusion dans le chapitre 6 où nous résumerons les buts initiaux et les résultats obtenus et nous exposerons quelques idées pour des travaux futurs liés à notre environnement.

CHAPITRE 2

MODÉLISATION DES CONNAISSANCES DANS LES STI

Les STI, de par leur nom, sont supposés d'une certaine façon, apporter de l'intelligence aux systèmes d'enseignement basés sur l'utilisation de l'ordinateur. Il y a deux endroits clés où l'apport de cette intelligence est un élément important dans un STI. Le premier est relatif à la connaissance que possède le système sur la matière à enseigner. Le second est relatif aux principes par lesquels ce système enseigne et les méthodes par lesquelles il applique ces principes. Dans l'enseignement de tous les jours, un enseignant humain n'est efficace dans sa tâche d'enseignement que s'il possède les deux types d'intelligence; et un déficit dans l'une ou l'autre de ces deux composantes mène inévitablement à une diminution de la qualité de l'enseignement dispensé par celui-ci. Un enseignant ne peut enseigner de manière efficace dans un domaine où il a peu ou pas d'expertise, et d'un autre côté, un expert ne peut faire mieux s'il n'a pas une connaissance suffisante des principes de pédagogie et des méthodes d'enseignement.

Dans le domaine des STI la situation est assez similaire. Un STI efficace ne peut exister sans avoir à sa disposition une connaissance riche et détaillée sur le domaine à enseigner [Anderson 88], tout en possédant une démarche (dite stratégie) pédagogique pour aboutir aux objectifs d'enseignement fixés.

2.1 Architecture de base des STI

Bien que les STI diffèrent beaucoup dans leurs implémentations et leurs domaines d'application, il se dégage, à leur examen une architecture de base commune, plus au moins respectée. Cette architecture de base est composée de quatre composants : le module de l'apprenant, le module tuteur, le module expert et l'environnement d'apprentissage (Figure 1).certains approches distinguent un planificateur capable de décomposer les actions pédagogiques et la matière à présenter. Dans le schéma de base indiqué ci dessous, ce planificateur est dans le module Tuteur.

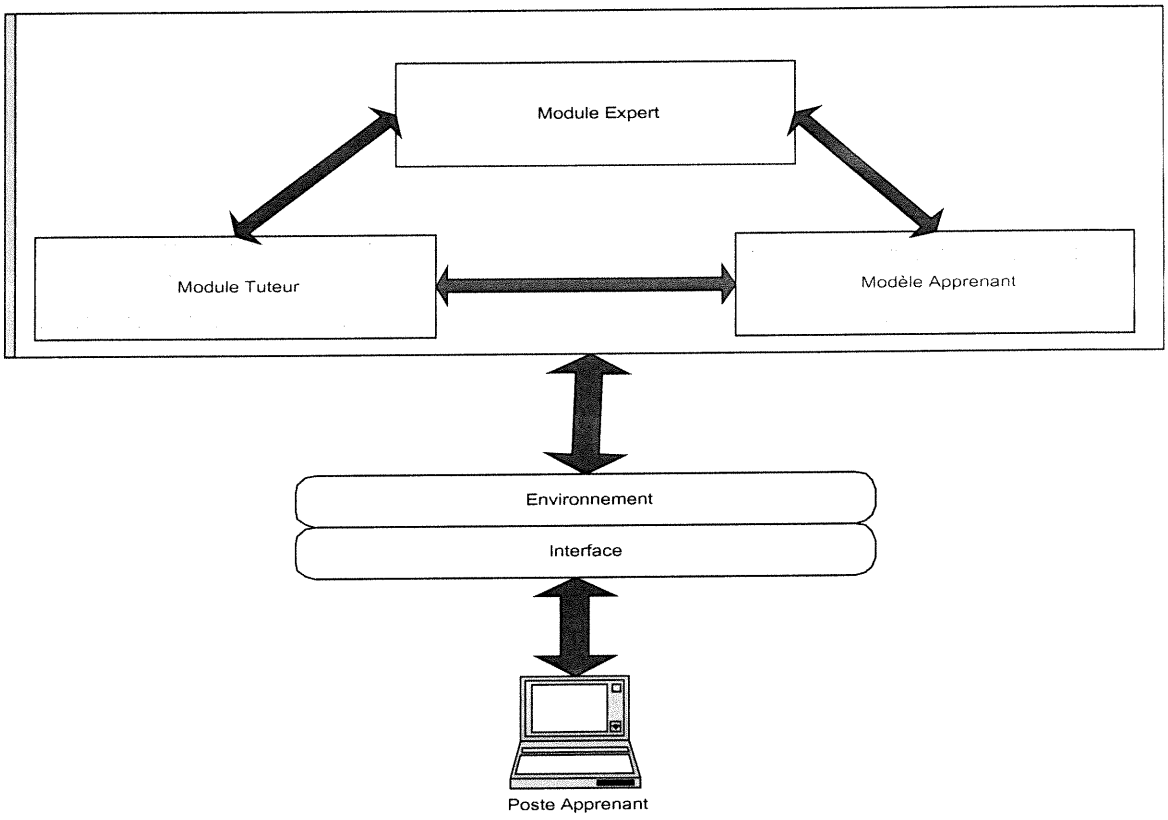


Figure 1 : Architecture de base d'un STI

- **Le module de l'apprenant** : l'un des rôles des STI est de pouvoir dispenser une formation individualisée. Le module de l'apprenant est le composant du STI chargé du suivi et de la mise à jour des informations sur l'apprenant (nom, prénom, profil cognitif, préférences, ...). L'information sur le profil cognitif de l'apprenant par exemple, permet aux STI de prendre en considération le niveau de connaissance de la matière à enseigner, propre à chaque apprenant. Et l'information sur les préférences de l'apprenant, leur permet d'adapter et d'individualiser leurs présentations (préférence pour des ressources dans une langue, pour des ressources visuelles, abstraites, ...). Ces informations sont contenues dans le modèle de l'apprenant.

- **Le module tuteur** : c'est le composant des STI responsable du contrôle des interactions (quand intervenir, sur quel point, quelles informations apporter) et de la gestion du déroulement du processus d'enseignement (contrôle global d'apprentissage). Il prend en considération la stratégie d'enseignement à suivre. Pour cela, le module tuteur dispose d'un ensemble de stratégies pédagogiques déterminées par une expertise pédagogique. Les actions de l'apprenant sont alors suivies et contrôlées et le modèle de l'apprenant est mis à profit pour fournir des interventions pertinentes. Le module tuteur permet à un STI d'avoir des comportements évolués résumés dans les points suivants [Halff 88]:
- Contrôler, sélectionner et ordonnancer la matière à enseigner.
- Répondre aux questions de l'apprenant sur les objectifs de l'enseignement en cours et son contenu.

- Avoir des stratégies pour déterminer quand un apprenant a besoin de l'aide du tuteur et pouvoir offrir l'aide adéquate.
- **L'environnement d'apprentissage** : l'environnement d'apprentissage est formé des éléments du STI qui supportent l'apprenant dans ces activités d'apprentissage. C'est un lien à double sens entre l'apprenant et le STI. Par son biais passe la communication entre les deux entités qui peut prendre des aspects très riches et variés allant de la simple page WEB (Word Wide Web) multimédia aux applications en réalité virtuelle avec casque et gants d'interaction.

- Le module expert : c'est le composant le plus important dans un STI. Il se base sur la connaissance modélisée du domaine pour donner au STI la capacité d'expliquer et de résoudre les problèmes dans le cadre d'un processus d'enseignement. Les composants du STI qui ont besoin de la connaissance du domaine pour réaliser leurs tâches, vont la chercher auprès du module expert. Ces connaissances peuvent être relatives au savoir ou au savoir-faire.

2.2 Agents Intelligents

Avant de commencer de définir les agents intelligents, nous essayons de trouver la définition du mot ''Agent''. Un dictionnaire généraliste qualifie un agent de "personne chargée des affaires et des intérêts d'un individu, d'un groupe ou d'un pays, pour le compte desquels elle agit." (Dictionnaire Robert).
L'association française de normalisation (AFNOR) pour sa part les définit ainsi : "Objet utilisant les techniques de l'intelligence artificielle : il adapte son comportement à son environnement et en mémorisant ses expériences, se comporte comme un sous-système capable d'apprentissage : il enrichit le système qui l'utilise en ajoutant, au cours du temps, des fonctions automatiques de traitement, de contrôle, de mémorisation ou de transfert d'information.

2.2.1 Autonomie

L'agent doit pouvoir prendre des initiatives et agir sans intervention de l'utilisateur final. Dans le contexte du WEB par exemple, il doit pouvoir agir alors que l'utilisateur est déconnecté (mode off-line).
La plupart des éditeurs de logiciels ''Agents'' contournent le problème de l'autonomie en programmant leurs logiciels afin qu'ils puissent, à intervalles réguliers de temps, se connecter automatiquement au WEB pour y effectuer les tâches que leur ont confiés les utilisateurs. Dans ce cas là, on ne peut véritablement parler d'autonomie car l'agent n'est pas actif en permanence et ne fait que reproduire comme un automate la requête programmée par l'utilisateur.

L'autonomie veut dire réellement qu'il n'existe plus de connexion continue entre l'agent et son maître. L'agent ne devient autonome que dans la mesure où il a des connaissances sur la volonté de son maître. Prenons le cas d'un agent chargé d'acheter des billets d'avion, la connaissance des compagnies préférées du maître (Air Canada, Royal Air Maroc, Air France, ...) ou les places préférées (hublot / corridor, fumeurs/non fumeurs...) peut économiser énormément de temps.

Cette propriété d'autonomie est très importante du fait qu'elle permet à l'agent de travailler tout seul sans avoir recours à une permanente connexion TCP.

2.2.2 Communication et Coopération

L'agent doit pouvoir échanger des informations plus ou moins complexes avec d'autres agents, avec des serveurs ou avec des humains.
Le service Firefly (un système de recommandation de sites, de films, de groupes musicaux basés sur les travaux de Pattie Maes et son équipe de chercheurs du MIT) permet, par exemple, à l'agent représentant un usager, d'échanger des informations avec d'autres agents représentant d'autres utilisateurs à fin de rechercher ceux d'entre eux ayant les mêmes centres d'intérêts.

2.2.3 Raisonnement

L'agent doit être capable de s'adapter à son environnement qui peut être composé d'autres agents, du WEB en général ou des utilisateurs et aux évolutions de celui-ci. Cette adaptation doit s'appuyer sur l'analyse de l'environnement extérieur des agents. Au départ, l'agent possède un certain nombre de règles et fonctionnalités lui permettant d'interagir avec l'utilisateur. Mais au fur et à mesure de l'évolution de l'apprenant dans le cours, l'agent devrait déduire de nouvelles règles de gestion pour augmenter son efficacité.

2.2.4 Mobilité

Les agents doivent pouvoir être multi-plate-forme et multi-architecture. Ils doivent pouvoir se déplacer sur le réseau où ils accomplissent des tâches sans que l'utilisateur ait le moindre contrôle sur celles-ci.

L'agent mobile est un programme qui se transporte dans une machine hôte (Remote Machine) et s'exécute. Plus encore, durant son exécution, il peut décider de se déplacer vers une autre machine.

2.3 Modélisation des connaissances dans le module expert

La modélisation des connaissances dans le module expert est un processus d'analyse et de codification relative à un domaine en vue de leur utilisation par un ordinateur à des fins d'enseignement. La conception et la mise au point de telles bases de connaissances est une opération longue, coûteuse, qui exige une grande technicité à la fois dans le domaine à modéliser et en matière de représentation informatique des connaissances [Quéré 91].

Cette représentation peut prendre plusieurs formes parmi lesquelles :
- La représentation en ***logique du premier ordre*** : l'outil principal utilisé pour la formulation des connaissances dans cette approche est le langage PROLOG, comme dans le système EXP'AIR [Cabrol et al., 91]. L'un des avantages d'utiliser ce langage est d'être à la fois un langage d'expression de connaissances et un résolveur utilisant ces connaissances.

- La représentation en ***règles de production*** : Les règles de production sont des règles de type "Si <condition> alors <expression>". Associées à un moteur d'inférence, elles constituent un moyen simple de représentation de connaissances, car le savoir humain s'exprime souvent par des règles de ce type. Leur utilisation dans les STI est apparue avec GUIDON de Clancey [Clancey 82]. Le défaut majeur de cette approche est que son raisonnement est une recherche en arrière exhaustive qui ne correspond pas au raisonnement d'un humain. Ce qui rend très difficile l'explication de ce raisonnement à un apprenant.
- La représentation en ***réseaux sémantiques*** : Les réseaux sémantiques sont un mode de représentation très utilisé en intelligence artificielle, notamment dans les systèmes de compréhension du langage naturel. Ils permettent une organisation des connaissances de

type relation entre concepts; ils sont faciles à lire et à comprendre et permettent une expression graphique. Leur point faible dans le passé était l'absence de représentation des aspects dynamiques (procédures) de l'expertise. Cette lacune a été comblée dans le système SYNERGY [Rouane 96], qui est un langage de programmation multi-paradigmes basé sur l'utilisation des graphes conceptuels, développé dans le cadre du projet SAFARI.

D'autres essais de modélisation se sont basés sur l'utilisation d'autres modes de représentation, comme l'utilisation des réseaux de Petri [Ferraris 84]. Cependant la majorité de ces modélisations souffrent d'une faiblesse soit au niveau de la représentation de l'expertise du domaine, soit au niveau de celle de l'expertise pédagogique. Mais l'une des modélisations les plus complètes est celle proposée par le modèle CREAM basé sur la modélisation de l'expertise sous forme de réseaux de connaissances.

Le modèle CREAM est le fruit des travaux réalisés dans le cadre du projet SAFARI. Son originalité vient du fait qu'il considère la matière à enseigner sous deux aspects importants en même temps:
- l'expertise du domaine : exprimée dans un modèle de capacités sous la forme d'un réseau de connaissances comprenant les capacités à acquérir et les relations entre ces capacités.

- l'expertise pédagogique : exprimée dans deux modèles, le modèle des objectifs comprenant les objectifs pédagogiques et leurs relations, et le modèle pédagogique exprimant les relations entre objectifs et capacités.
Ce modèle se distingue par le fait qu'il propose des mécanismes explicites pour l'intégration du module de connaissances avec les autres composants d'un STI et pour la génération automatique des cours individualisés.

2.1 Module expert

Le module expert est l'entité d'un STI chargée d'emmagasiner et de fournir les connaissances sur le domaine à enseigner.

L'extraction de l'expertise d'un domaine donné et sa codification en vue d'un enseignement informatisé, est une tâche longue et exigeante. La grande quantité de connaissance exigée pour l'enseignement de domaines complexes, fait du développement du module expert une des phases les plus exigeantes en effort et la plus consommatrice en ressources. D'où la grande importance qu'on accorde aux outils d'édition de connaissance dans les STI, qui peuvent apporter un support très important aux équipes de conception par le biais d'interfaces d'utilisation adéquates et de fonctions d'aide à la conception.

La codification de la connaissance dans les STI peut être faite selon trois approches. La première approche consiste à trouver une façon de raisonner dans le domaine à enseigner sans avoir à codifier ou représenter la connaissance sous-jacente à l'intelligence humaine. Par exemple, un système informatique peut calculer la valeur d'une intégrale par un processus numérique itératif, alors qu'un humain peut calculer la même intégrale par processus symbolique. Le système ne possède pas les connaissances humaines pour faire un traitement symbolique du problème, mais peut arriver au même résultat que l'humain. Seulement il ne peut pas fournir d'explication à un apprenant humain puisqu'il n'utilise pas le même système de résolution. On appelle alors ce genre de système, "système à boîte noire ".

La seconde approche pour la codification de la connaissance dans les STI est le système à boîtes transparentes. Par opposition à la première approche où la connaissance était totalement encapsulée, cette approche rend la connaissance et le raisonnement interne du système, transparents au monde extérieur, dont l'apprenant fait partie. La représentation de cette connaissance et le type de raisonnement doivent être proches de celles de l'apprenant pour que les explications fournies par le système soient compréhensibles et assimilables par celui-ci.

La troisième approche pour la codification de la connaissance dans les STI, est une approche basée sur l'utilisation du modèle cognitif où on tente de développer une simulation de la démarche humaine dans la résolution des problèmes, et de décomposer la

connaissance d'une façon qui s'approche le mieux de la décomposition faite par un humain.

Les trois approches de codification ont été présentées dans l'ordre croissant de leur efficacité dans la construction d'un STI. Mais c'est aussi dans l'ordre de difficulté et d'effort croissant nécessaire à leur développement. La Figure 2 donne la corrélation qui existe entre l'efficacité pédagogique d'un STI donné, et l'effort d'implémentation de son module expert.

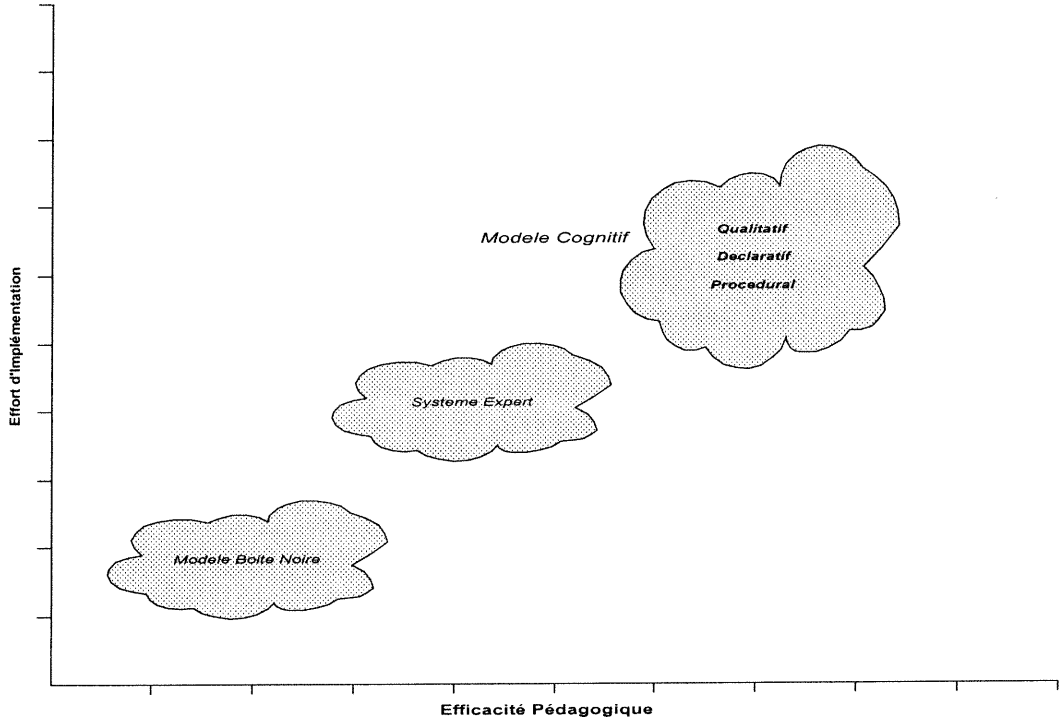


Figure 2 : Corrélation entre l'efficacité pédagogique et l'effort d'implémentation

2.1.1 Modèle expert à boîte noire

Le modèle expert à boîte noire est un modèle capable, vis-à-vis d'un état à son entrée (un événement, une situation...), de générer le comportement correct et opportun, ce qui permet de l'utiliser comme un juge ou référentiel pour la détermination de la validité ou non des actions prises dans un processus d'enseignement. Cependant, les mécanismes internes par les quels il aboutit à ses conclusions, ne sont pas accessibles de l'extérieur ou ne sont pas utilisables dans la tâche d'enseignement en cours. L'exemple classique de ce type de système est constitué par les travaux originaux du système SOPHIE [Brown et al., 75]. Ce système utilisait un simulateur de composantes électroniques appelé SPICE II qui était utilisé pour enseigner à des élèves comment détecter les pannes dans les circuits électroniques. Le tuteur utilise le simulateur pour déterminer la validité des mesures faites par l'élève dans sa tentative de détermination de la cause de la panne. Comme le système SPICE procède par résolution d'un ensemble d'équations, plutôt que de travailler comme aurait fait un humain, il était incapable d'expliquer en détail les décisions qu'il prenait.

Le fait de ne pas pouvoir expliquer en détail les décisions prises et le fait d'encapsuler la connaissance limitent l'utilisation des modèles experts à boîte noire. Ils sont généralement utilisés avec des tuteurs réactifs simples qui ont pour rôle de déterminer si les actions de l'apprenant sont bonnes ou mauvaises, et peut-être, de lui indiquer la bonne action à prendre (Figure 3).

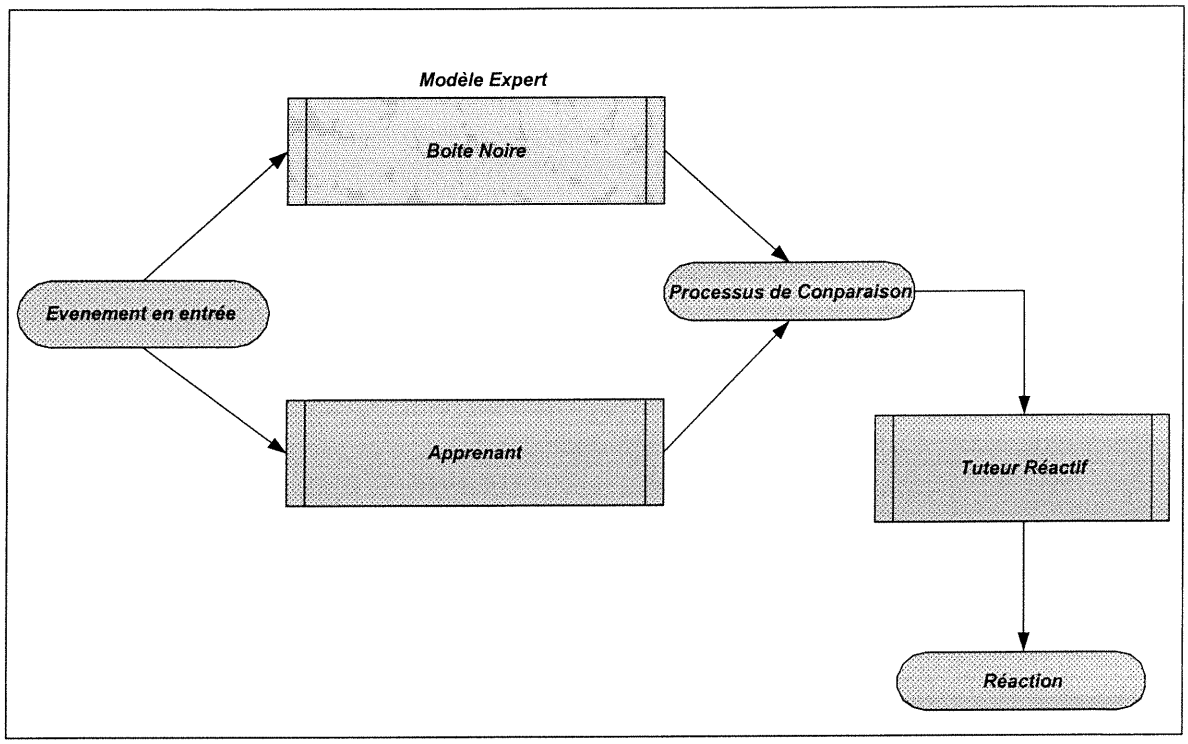


Figure 3 : Tuteur réactif

La combinaison modèle expert à boîte noire et tuteur réactif possède un avantage intéressant. Il offre le moyen le plus économique pour convertir un système expert en un système d'enseignement.

Cependant, un STI ne peut être vraiment efficace s'il se base uniquement sur un dialogue restreint lié aux trois notions : 'Bonne action', 'Mauvaise action' et 'Action à faire'. Une des solutions proposée consiste à dégager des modèles dans le comportement de l'expert et dans le comportement de l'apprenant et de leur rattacher des instructions pour que le système fournisse des explications ou des exemples à la détection de l'un de ces modèles de comportement. Seulement, cette analyse de comportement est une analyse de surface. Et l'accès à la connaissance interne du module expert est nécessaire pour fournir des explications profondes et adéquates.

2.1.2 Modèle expert à boîte transparente

La seconde approche pour la codification de la connaissance dans les STI est basée sur l'utilisation des systèmes experts traditionnellement utilisés dans le domaine de l'intelligence artificielle. La construction de ces systèmes experts est une tâche très difficile impliquant des ingénieurs de la connaissance et des experts du domaine, qui doivent identifier la nature du problème et le rayon d'étude, énumérer et formaliser les concepts clés dans le domaine, réaliser le système qui va représenter cette connaissance, et puis, de façon itérative, tester et raffiner ce système.

Ce qui est intéressant dans cette démarche, est qu'on cherche à étudier et à modéliser l'expertise du domaine en prenant en considération la façon dont l'homme raisonne et utilise cette expertise. Par analogie à l'exemple de calcul de l'intégrale, on va considérer la manière dont l'homme utilise le raisonnement symbolique pour le calcul de cette intégrale, plutôt que de considérer le calcul par itérations numériques, qui donnera certes le même résultat, mais qui a peu d'intérêt d'un point de vue pédagogique. Et du fait que la connaissance dans les systèmes experts est une représentation directe de la connaissance humaine avec l'intelligence humaine sous-jacente, les modules experts basés sur les systèmes experts sont plus disposés à l'enseignement que les modules experts à boîte noire.

Dans la littérature des STI, l'exemple classique de module expert basé sur les systèmes experts est GUIDON de Clancy [Clancey 82] qui est un système d'enseignement du diagnostic des infections bactériologiques, basé sur le système expert MYCIN. MYCIN est un système expert très populaire dont le domaine d'expertise est le diagnostic des infections bactériologiques. Il possède une base de connaissances comprenant quelque 450 règles (de type Si-Alors), qui codifient les étapes et les démarches dans un acte de diagnostic médical. La figure suivante donne un exemple d'une telle règle.

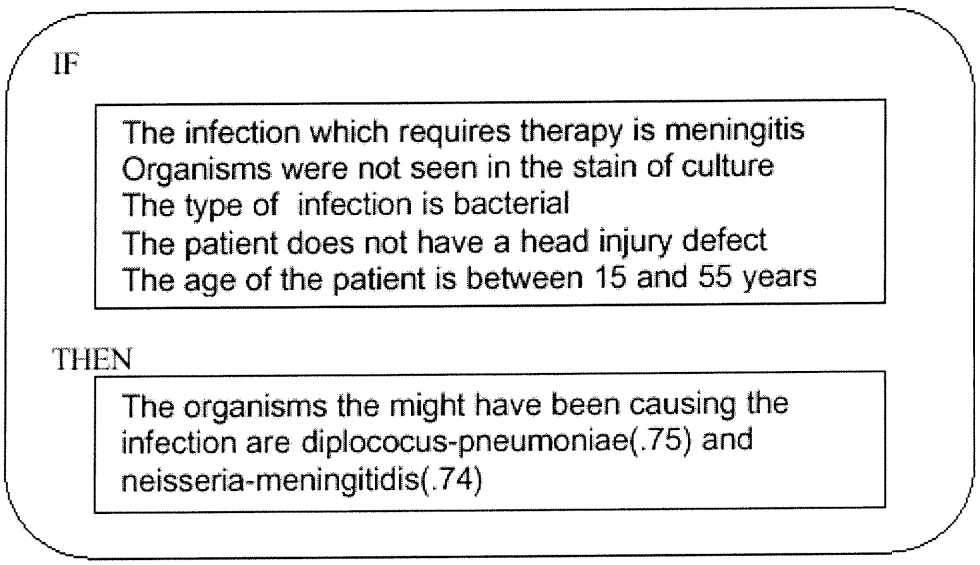


Figure 4 : Exemple de règle dans MYCIN

La tâche d'enseignement dans le système GUIDON est contrôlée par des règles pédagogiques spéciales, dite règles de types t-rules. Bien que ces règles se basent sur la comparaison des comportements de l'expert et de l'apprenant comme dans les modules experts à boîtes noires, elles se différencient par le fait qu'elles prennent en considération le processus de raisonnement de l'expert. Et les entités auxquelles elles réfèrent sont des entités internes au module expert lui-même, chose qui était impossible dans les modules experts à boîtes noires. Un exemple de ces règles pédagogiques de type t-rule est donné par la Figure 5.

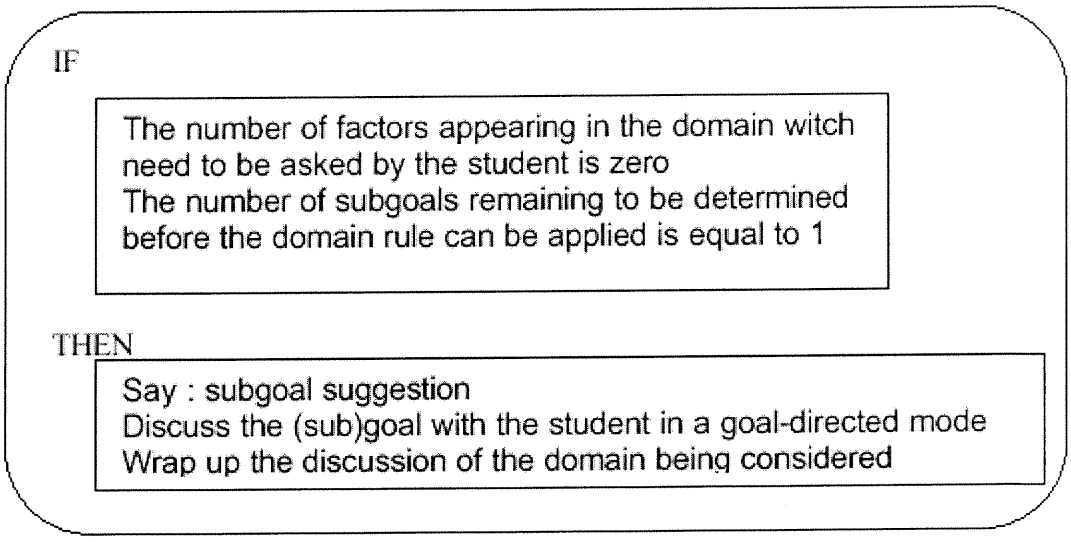


Figure 5 : Exemple de règle tutorielle dans GUIDON

Cependant, le véritable mécanisme de raisonnement utilisé par MYCIN pour déployer sa connaissance, qui est une recherche en arrière exhaustive, est très différent des mécanismes de raisonnement chez l'humain. De plus, les règles dans MYCIN, de nature optimisée, étaient très difficiles à justifier par GUIDON. Et plusieurs des règles dans MYCIN, bien que très appropriées pour un expert, étaient très compliquées pour être directement enseignées à un novice.

Toutes ces difficultés ont conduit à la conception de NEOMYCIN, dans lequel on a tenté d'introduire une structure de contrôle différente du domaine de connaissance constitué par la base de règles. La structure de contrôle devient une entité indépendante constituée d'un ensemble de règles sur comment utiliser et enseigner les règles de la base de connaissance. Et pour faciliter les explications, les règles de type t-rule ont été mises dans une structure séparée.

Les travaux sur le système GUIDON ont permis de dégager des conclusions importantes qui ont beaucoup influencé le domaine de développement des STI. Une des conclusions dégagées indique que pour la construction d'un STI vraiment efficace, il est nécessaire de prêter attention, non seulement aux connaissances à enseigner mises dans le modèle expert, mais aussi à la façon dont cette connaissance sera présentée et enseignée à l'apprenant.

2.1.3 Modèle expert à modèle cognitif

Le but de l'approche avec modèle cognitif est de développer une simulation de la démarche humaine dans la résolution des problèmes, où la connaissance utilisée est une décomposition du domaine tentant de mieux s'approcher de la décomposition de cette connaissance fait par un humain.
Cette manière de décomposer la connaissance donne à cette approche le pouvoir de communiquer facilement cette connaissance à un apprenant, d'une façon naturelle et profonde. Cependant, elle apporte des difficultés sur deux niveaux :

- La première des difficultés est relative aux efforts et aux coûts de développement bien supérieurs à ceux demandés dans la construction d'un système expert. Ce qui fait ressortir l'importance des outils de construction qui peuvent par leurs apports augmenter la productivité des concepteurs de ces systèmes et diminuer les coûts de réalisation.
- La deuxième de ces difficultés est relative à la complexité de ces systèmes dont l'exploitation peut exiger des moyens de traitement assez puissants. Ce qui peut faire augmenter les coûts de mise en œuvre à grande échelle. Mais la constante augmentation de puissance et diminution des prix des moyens de traitement, font que les effets de cette difficulté sont de plus en plus négligeables.

Dans les modèles experts à modèle cognitif on distingue deux types de connaissances qui peuvent être enseignées :

- Les connaissances procédurales : ce sont les connaissances dont l'information à communiquer à l'apprenant est une procédure qui montre comment réaliser une tâche ou résoudre un problème. Un exemple de ce type de connaissances est la résolution de problèmes dans le domaine du calcul. Dans la représentation de cette connaissance on s'inspire beaucoup des techniques de l'intelligence artificielle dans le domaine des systèmes experts. Le formalisme standard généralement utilisé est celui des systèmes à base de règles car c'est l'une des techniques qui modélise le mieux la démarche humaine dans la résolution des problèmes. On retrouve cette approche dans des systèmes comme LISP Tutor [Reiser 85]. Le grand avantage des systèmes de production à base de règles dans un processus d'enseignement est le fait que chaque règle de production est une entité de connaissance indépendante. Ce qui signifie qu'on peut communiquer cette règle à l'apprenant de façon unitaire sans avoir à communiquer toute la structure de connaissances dans laquelle elle apparaît.
- Les connaissances déclaratives : ce sont des connaissances à enseigner relatives à des faits. Le domaine de la géographie est un exemple de domaine qui s'apprête bien à ce type de représentation.

2.2 Stratégies d'enseignement dans les STI

Pour avoir un STI efficace, l'expertise du domaine doit être jumelée à une expertise pédagogique. Le STI doit posséder des démarches pédagogiques, appelées stratégies d'enseignement, pour atteindre les objectifs d'enseignement. Une stratégie d'enseignement est une analyse du comportement d'un STI dans trois contextes: la présentation du matériel pédagogique, l'intervention pour corriger les erreurs de l'apprenant et les réponses aux questions de l'étudiant [Halff 88].

2.2.1 Les STI procéduraux

Le but de ce type de STI est d'enseigner aux apprenants des habiletés de type procédurales. Le système se comporte comme un observateur vis-à-vis des actions de l'apprenant dans une activité d'enseignement qui se déroule (résolution d'un problème par exemple). Et l'analyse de ces actions permet au système d'intervenir au moment opportun pour interrompre l'activité en cours et présenter des éléments d'information (comme des rappels, indications, explications, ...) qui peuvent aider à l'atteinte de l'objectif visé.

Le choix du moment d'intervention est primordial. Plusieurs stratégies ont été déterminées pour le choix de ce moment :
- Stratégie avec " Model Tracing" : c'est une stratégie très simple à mettre en œuvre. L'observation des actions de l'apprenant par le système se fait à la trace et le système intervient dès qu'une erreur est survenue [VanLehn 88].
L'inconvénient de cette stratégie est que l'étudiant ne peut pas explorer des chemins erronés dans l'espace du problème, et retrouver de lui-même ses erreurs en faisant face à des contradictions ou des impasses.

- Stratégie avec "Issue-based" [Burton 82] : c'est une stratégie plus élaborée que la précédente. Connue aussi sous le nom de possibilités et exemples, elle est basée sur le principe d'exploitation d'opportunités. La démarche consiste pour le système à préparer d'avance des opportunités possibles pour l'apprenant dans l'étape en cours d'exécution, et qui sont considérées comme étant les meilleures opportunités. Le

système intervient alors dans les cas où l'apprenant fait le choix d'une opportunité, même si elle est jugée correcte, mais qui n'appartient pas à cet ensemble. Comme chaque opportunité est assortie d'exemples et d'explications, l'étudiant peut revoir sa démarche pour dégager ses faiblesses et pouvoir ainsi s'améliorer dans le futur.

- Stratégie avec intervention sur demande ou lors d'actions dangereuses : les tuteurs des systèmes basés sur les deux stratégies précédentes sont appelés *coachs* [Nkambou 96]. Leur intervention est une réaction à des situations particulières dans le déroulement de l'enseignement et n'attendent pas la sollicitation de l'apprenant. À la différence de ces systèmes, les systèmes basés sur des stratégies avec conseiller n'interviennent qu'à la demande de l'apprenant (ou d'actions jugées très dangereuses). Sherlock [Lajoie 89] est un exemple de ces systèmes.
Le choix entre coach et conseiller dans de tels systèmes, peut être fonction du niveau du public qui l'utilise : un coach est plus approprié à un public novice ou intermédiaire, tandis qu'un conseiller est plus approprié à un public expert.

2.2.2 Les STI socratiques

Basés sur un jeu de question/réponse, ce type de système permet à l'apprenant des autocritiques et de découvrir de lui-même ses erreurs et contradictions avec les inférences qu'il fait. Le système engage un dialogue continu avec l'apprenant, et des décisions pédagogiques sont prises en fonction des réponses fournies par ce dernier. Chaque fois qu'une erreur est commise, le système cherchera à entamer un nouveau dialogue qui peut amener l'apprenant dans des contradictions et de là découvrir ses erreurs.
Un des systèmes représentatif de cette catégorie de STI est le système SCHOLAR [Carbonell 70]. C'est un système destiné à l'enseignement de la géographie de l'Amérique du Sud.

Un autre système important de cette famille est WHY [Stevens 77]. Son importance vient du fait qu'il est le premier système qui explicite de manière abstraite, les règles pédagogiques utilisées dans la génération automatique des questions dans les dialogues.

Ce genre de système peut être utilisé plus convenablement dans l'enseignement des informations factuelles ou l'enseignement de règles.

2.2.3 Les STI critiques

C'est une variance des systèmes procéduraux avec coach ou conseiller. Leur démarche consiste à critiquer les actions de l'apprenant dans son processus de résolution de problème. La nature de la critique peut être de nature positive si l'apprenant excelle dans son action durant la tâche en cours. Comme elle peut être de nature négative si l'apprenant commet des erreurs ou des violations des règles de bonne conduite dans le processus de résolution. Ce genre de systèmes est parfaitement adapté à l'enseignement des systèmes de conception. Le système dispose de l'expertise d'un expert dans un domaine de conception donné. Le système peut alors suivre la démarche de l'apprenant dans le processus de conception et critiquer celle-ci en la comparant à la démarche de l'expert et en vérifiant la non-violation des règles de bonne conception imposée par ce dernier.

Un des systèmes représentatifs de cette catégorie et CRACK [Fisher 88]. Il est destiné à l'enseignement de la conception des cuisines. Il dispose des normes de la conception d'une bonne cuisine sous forme de règles et peut suivre, analyser et critiquer la démarche de conception d'un apprenant dans ce domaine. De plus ce système peut enrichir sa base de connaissance chaque fois que l'apprenant réalise une performance correcte mais non connue du système. Ce qui permet au système de modifier et d'améliorer son comportement au fil de son utilisation.

2.2.4 Les environnements de découverte interactifs

Le but de ce genre de systèmes est de représenter une situation du monde réel dans laquelle l'apprenant peut changer à volonté la valeur de tel ou tel facteur, ce qui provoque une modification de l'image du monde représenté et permet à l'apprenant de découvrir les règles qui régissent ce monde.

En intégrant des outils d'exploration, ces systèmes permettent de réaliser un apprentissage tout en explorant l'objet à enseigner lui-même. Cet objet prend vie par une simulation présentée à l'apprenant, qui peut alors interagir avec l'objet simulé. Au cours de cette interaction, le système peut dans ses interventions se comporter soit à la manière d'un Coach et prendre l'initiative à chaque fois qu'il le juge nécessaire (on parle de mode guidé), soit à la manière d'un Conseiller et laisser la liberté à l'apprenant de manipuler l'objet et de prendre les initiatives (on parle de mode exploration). L'apprentissage devient alors une succession de découvertes faites par l'apprenant et où sa curiosité joue un grand rôle. Le système LOGO [Papert 80] incorpore parfaitement la philosophie de ce type de systèmes.
Quand l'objet de l'étude est un dispositif physique (avions, appareils médicaux, ...) on parle alors d'environnement d'apprentissage basé sur la simulation (Simulation-based training).
On distingue plusieurs types de ces systèmes. On parle de simulation modélisante, de simulation dynamique, de simulation méthodologique et de simulation opérationnelle [Quéré 91] :

- Simulation modélisante : c'est une simulation dans laquelle le modèle simulé est inconnu de l'apprenant, qui doit alors l'étudier et le découvrir.
- Simulation dynamique : c'est une simulation dans laquelle l'apprenant connaît le modèle simulé et doit étudier l'influence des paramètres sur le modèle et découvrir les règles qui les régissent.
- Simulation méthodologique : c'est une simulation dans laquelle l'apprenant étudie l'adéquation et la validité du modèle simulé.
- Simulation opérationnelle : c'est une simulation dans laquelle l'apprenant conduit et dirige le modèle simulé (par exemple, les simulateurs de vol).

Ces systèmes, en plus d'avoir les mécanismes pour réaliser les simulations, doivent avoir une explicitation des connaissances relative au domaine de la simulation pour pouvoir analyser les actions de l'apprenant et pouvoir lui apporter des interventions pertinentes.

2.2.5 Les STI multi agents

L'architecture des STI s'était articulée sur quatre composantes de base : le module expert chargé de l'expertise du domaine à enseigner, le module tuteur chargé de l'aspect pédagogique, le modèle de l'apprenant chargé de la modélisation de l'apprenant, et l'interface chargée de l'aspect présentation et communication.

Pour enrichir cette architecture, certains STI font intervenir d'autres composantes. Le système d'apprentissage avec compagnon [Chan 90], fait intervenir un agent simulant un compagnon de formation pour l'apprenant réel qui utilise le système. Le système tente d'exploiter la collaboration et la compétition qui peuvent naître de la présence d'un compagnon, pour stimuler l'apprenant humain dans sa tâche d'apprentissage.

Une autre approche dans ce type de systèmes consiste à faire intervenir un agent perturbateur dans le dialogue tuteur-apprenant [Aïmeur 97]. C'est une entité qui a des connaissances du domaine au même niveau que le tuteur, et qui prend l'initiative de faire des suggestions et des indications à l'apprenant, et qui ne sont pas toujours correctes. Ce qui pousse l'apprenant à toujours les prendre avec réserve, en les analysant et le critiquant, l'amenant ainsi à augmenter sa confiance dans les actions qu'il prend (Figure 6).

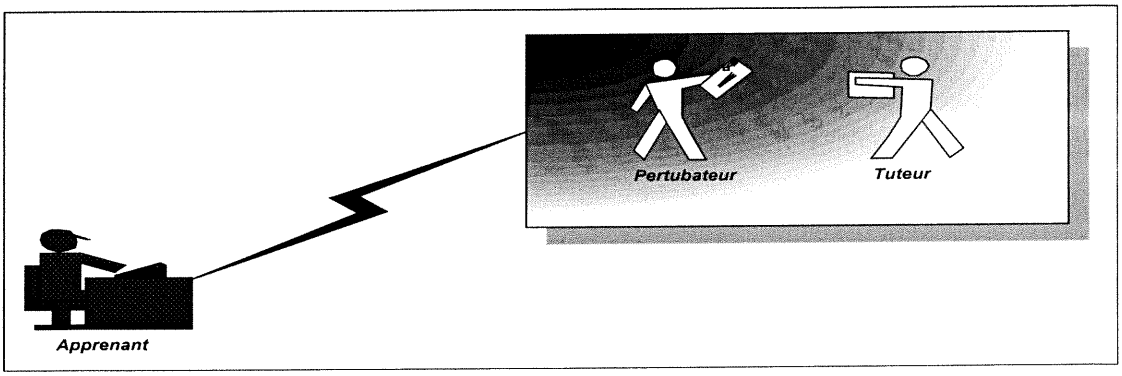


Figure 6 : Apprentissage avec Perturbateur

Ces nouveaux agents ne sont pas nécessairement des applications informatisées. Ils peuvent aussi être des agents humains intervenants dans le processus : un ou plusieurs tuteurs, un ou plusieurs apprenants collaborant ensemble dans une tâche de formation commune. Dans le système GRACILE [1] [Ayala 95] pour l'enseignement de la grammaire japonaise, une communauté d'étudiants travaillent ensemble dans un processus d'apprentissage commun. Un agent médiateur est responsable de la gestion de la communication entre les étudiants.

[1] Japanese GRAmmar Collaborative Intelligent

2.3 Conclusion

Le module expert est une composante très importante dans un STI. La façon dont l'expertise du domaine est modélisée dans le module expert et son accessibilité ont une influence sur l'efficacité pédagogique du STI. L'intégration d'une expertise pédagogique dans le module expert est nécessaire dans les STI pour supporter le processus d'apprentissage.

Nous présenterons dans le prochain chapitre notre approche pour la représentation de connaissances dans un STI. Nous examinerons comment nous abordons les aspects de modélisation reliés à l'expertise du domaine et à l'expertise pédagogique.

CHAPITRE III

CONCEPTION ET MODELISATION DE LA MATIERE

The primary purpose of the instructional design process is to structure the environment so as to provide a learner with conditions which will support learning processes.
(Gagné,Briggs et Wager, 1992).

3.1 Les Composants de la matière

La modélisation des connaissances passe par une étape qui consiste à identifier les connaissances que l'on veut représenter. Ce n'est qu'après cette étape qu'on pourra penser à un langage ou à un formalisme de représentation qui aboutira à un modèle de connaissances. Ainsi, il est nécessaire dans un premier temps de déterminer quelle est la connaissance à représenter, puis d'étudier comment l'organiser, comment l'exprimer et quelles contraintes à lui imposer pour quelle soit en accord avec l'univers modélisé.

Dans la suite, nous présentons une approche de modélisation et de représentation structurée de la matière à enseigner en terme d'unités cognitives (capabilities) au sens de Gagné [Gagné 92], de niveaux de maîtrise dont l'atteinte contribue à l'acquisition d'unités cognitives et de ressources didactiques (exercices, problèmes, vidéos, simulations, etc.). Tous ces éléments sont organisés dans des structures de connaissances pour soutenir l'enseignement d'une matière.

Notre idée est de créer et d'organiser un environnement riche et varié pouvant supporter la génération de cours , la planification et le déroulement de l'enseignement ainsi que

certains aspects reliés à la modélisation de l'apprenant. Cette approche met l'accent sur la séparation du cours et de la matière. Plusieurs cours peuvent être construits ou générés à partir d'une même base de connaissance. En outre, l'un de nos soucis est de pouvoir permettre au STI de supporter l'enseignement de cours complets.

3.1.1 Le Concept de cours

Un cours est un objet qui est souvent considéré comme une spécification des objectifs généraux et spécifiques décrivant l'intention du concepteur du cours. C'est un contrat entre l'enseignant (ici, le STI) et l'apprenant. Cette description sera utilisée par le STI pour réaliser des activités d'enseignement devant favoriser l'atteinte, par l'apprenant, des objectifs du cours.

Dans la littérature, plusieurs approches de description de cours existent ; un cours est souvent défini par un ensemble d'objectifs d'enseignement (du point de vue de l'enseignant) ou d'apprentissage (du point de vue de l'étudiant) ayant une finalité précise. Cet ensemble d'objectifs est généralement organisé dans une structure hiérarchique appelée hiérarchie d'apprentissage [Gagné 92] ou dans des structures plus souples faisant ressortir une organisation relative aux thèmes et connaissances en jeu.

Dans la pratique, un cours est souvent décrit non en terme d'objectifs à atteindre (comportements que l'étudiant devrait démontrer une fois le cours suivi avec succès), mais plutôt en termes de notions (connaissances) que l'on veut faire acquérir aux étudiants.

Les deux grandes approches de description de cours ne sont pas du tout contradictoires. Si on connaît les capacités que l'on veut faire acquérir à un apprenant, on peut toujours définir des objectifs dont la réalisation permettra à ce dernier de posséder les capacités ciblées.

La composition du cours que nous proposons dans notre modèle repose sur les éléments décrits ci-après.

3.1.2 L'Unité cognitive

Le contenu de la matière à enseigner est basé principalement sur un réseaux d'unités cognitives. Chaque unité cognitive (UC) est un objet qui décrit un contenu bien précis dans la base de connaissance. Un objet de type UC doit avoir un titre (T) et un descriptif (D). Pour permettre à l'agent pédagogique une bonne planification lors d'une session d'apprentissage, le concepteur est appelé à fournir un temps estimatif d'apprentissage pour chaque UC. Chaque UC est liée à une ressource didactique (R). Le contenu de la ressource fait parti de l'ensemble (texte, html, vidéo, simulation, etc.).

Pour supporter le modèle de l'apprenant ainsi que son évolution dans le cours, le concepteur définit des niveaux de contrôles pour les unités cognitives pertinentes. Un niveau de contrôle est un objet qui décrit le niveau de difficulté, il est décrit par un titre et un descriptif. L'ordre des niveaux au sein d'une unité cognitive est ordonné du moins difficile au plus difficile.

Soient CR = Cours candidat à l'enseignement, UC = Unité Cognitive, CP = Concept, NM= Niveau de Maîtrise, EX = Ressource Exercice, GEXP = Groupe d'Explications , EXP= Explication, TEA=Temps Estimé d'Apprentissage , PR=Prérequis , R=Ressource didactique.

Soit E = Ensemble de la base de connaissance.

$CR = \{$ Titre, Descriptif $\}$ ***Union*** $\{ UC_1, UC_2, UC_3, \ldots. UC_n / \ UC_i \in E$ pour $i = 1, ..., n\}$.

$UC = \{$ Titre, Descriptif, TEA, R $\}$ ***Union***

$\{ CP_1, CP_2, CP_3, \ldots, CP_n \ / CP_i \in E$ pour $i = 1, \ldots, n \}$ ***Union***

$\{ NM_1, NM_2, NM_3, \ldots, NM_m / NM_i \in E$ pour $i = 1, ..., m \}$ ***Union***

$\{ PR_1, PR_2, PR_3, \ldots, PR_p / PR_i \in E$ pour $i = 1, \ldots, p \}$

soit T_i = temps d'apprentissage estimé pour acquérir l'unité cognitive UC_i, et soit E l'ensemble des sous unités cognitives de UC_i, on doit vérifier la contrainte suivante :

$$T_i >= \sum_{j=1}^{n} T_j .$$

avec T_j = temps d'apprentissage estimé pour acquérir l'unité cognitive C_j

$UC_j \in E$

et $n = Card(E)$.

3.1.3 Les Concepts

Dans l'objectif de faciliter la maîtrise par l'apprenant de l'unité cognitive, on définit un réseau de concepts au sein de chaque UC. Les concepts définis doivent figurer dans la ressource didactique liée à l'unité cognitive. Chaque concept est un objet défini par un titre, un identifiant unique dans la base de connaissance et la fréquence d'apparition dans la ressource. Pour chaque concept le concepteur définit un groupe d'explications qui sera exploité par l'agent explicatif lors d'une session d'apprentissage. Un concept est à la propriété d'une seule unité cognitive, ce pendant, il peut être référencé par n'importe quelle autre unité cognitive dans la base de connaissance.

3.1.4 Les Explications

Une explication est un objet représenté d'une manière unique dans la base de connaissance. Chaque explication est liée à une ressource qui représente son contenu. L'objet explication peut être référencé par un concept, un exercice ou une unité cognitive. Les explications sont catégorisées dans des groupes selon leur degré d'importance. Les objets d'explication servent de matière première pour l'agent explicatif. Toutefois, les explications feront toujours l'objet d'un processus d'amélioration de la part du concepteur suite à un feedback de l'agent pédagogique qui établit au cours de chaque session des statistiques auprès de tous les apprenants révélant ainsi les explications

pertinentes de celles qui ne le sont pas. C'est ce processus qui permet d'améliorer considérablement l'efficacité d'un STI.

3.1.5 Les Niveaux de maîtrise

Pour tester les connaissances de l'apprenant par rapport à une unité cognitive, on a développé la notion de Niveau de Maîtrise (NM). Ce dernier est un objet représenté dans la base de connaissance par un identifiant, un code qui identifie le degré de difficulté et un libellé. Un NM est propre à une unité cognitive. Pour chaque NM on peut développer un ensemble d'exercices qui seront présenté par l'agent pédagogique à l'apprenant. La notation suivante montre bien ce que c'est un NM.

NM = { Code, Libelle } ***Union*** { Ex_1, Ex_2, Ex_3, …, Ex_n }

Avec Ex_i : Exercice i de niveau NM.

Le niveau de maîtrise traduit le niveau de connaissance de l'apprenant pour une unité cognitive donnée. Notre modélisation est le résultat d'une simplification du sous modèle des objectifs dans le modèle CREAM.

3.1.6 Les Exercices

Un exercice est un objet lié dans la base de connaissance à un niveau de maîtrise. Il sert de base pour le test des connaissances de l'apprenant par rapport à l'unité cognitive en question. Un objet exercice est identifié par Identifiant unique, un titre, un type, une description concernant l'objectif de l'exercice, un seuil de réussite, un seul d'échec, le rang de l'exercice dans le niveau de maîtrise, un temps estimatif pour la réalisation de l'exercice et la ressource qui reflète le support de l'exercice (une page html, une séquence vidéo, une simulation, etc…). Les exercices sont divisés en deux catégories, le premier concerne des exercices génériques type exercice à choix multiple, exercice à choix multiple-multiple, problème à étape et autres. La deuxième catégorie concerne les exercices spécifiques à un domaine particulier comme un exercice de déroulement d'une simulation dans le domaine de la réalité virtuelle, etc…

3.1.7 Les Prérequis

Avant d'arriver à maîtriser une unité cognitive UC. L'agent pédagogique doit s'assurer que l'apprenant a déjà maîtrisé tous les prérequis nécessaires avant de procéder à l'apprentissage de UC. Un objet prérequis est tout simplement une unité cognitive combinée avec le niveau de maîtrise que l'apprenant doit atteindre pour l'acquérir. L'étudiant est appelé à faire les prerequis lorsque l'agent pédagogique détecte son échec pour acquérir le premier niveau de maîtrise de UC dans le cas ou ce dernier est une unité cognitive avec niveaux de maîtrise. Dans le cas ou UC expose juste le contenu sans niveaux de maîtrise, l'agent pédagogique doit s'assurer automatiquement si l'apprenant a déjà acquis tous les prerequis.

3.2 Modélisation de l'Apprenant

Le modèle de l'apprenant est le composant du STI qui se charge des connaissances de l'apprenant. Ce dernier comprend l'état courant des connaissances de l'apprenant dans le domaine étudié. Il comprend aussi un processus qui analyse, puis évalue les performances de l'apprenant pour mettre à jour son modèle, appelé diagnostic [Burns et Capps,1988] [VanLehn, 88]. Ces deux composants devront être conçus ensemble pour constituer le modèle de l'apprenant.

Si le système dispose de stratégies tutorielles, celles-ci seront mises en oeuvre afin d'adapter l'enseignement à l'apprenant. C'est pour cela que le système dispose d'un modèle de l'apprenant plus ou moins fidèle, c'est-à-dire représentant plus ou moins bien l'apprenant humain (à la limite, dans un système sans modèle de l'apprenant, c'est l'interface qui gère les stratégies, sans connaître l'apprenant; donc l'enseignement n'est pas adapté).

Dans un STI, le modèle de l'apprenant est destiné à répondre aux questions concernant l'apprenant. Selon Self (1987) ces questions sont de quatre niveaux:

- Que peut faire l'étudiant?
- Que sait l'étudiant?
- A quel type d'étudiants le système s'adresse?

- Qu'a fait l'étudiant?

Selon VanLehn (1988) le modèle de l'apprenant est utilisé selon quatre buts:

- Indiquer le prochain sujet enseigné par le STI;
- Indiquer au tuteur le bon moment pour offrir de l'aide à l'étudiant;
- Permettre au système de présenter à l'étudiant des problèmes adaptés à son niveau de maîtrise de la matière;
- Indiquer au tuteur le bon moment pour produire des explications personnalisées pour l'étudiant.

3.2.1 Problème de modélisation de l'apprenant

Les difficultés liées à la conception du modèle de l'apprenant sont décrites dans la littérature comme étant le problème de modélisation de l'apprenant.
Les modèles des apprenants ont pour objet de représenter toutes les facettes de ce denier. Les informations qui doivent y être contenues sont liées à l'ensemble des taches pédagogiques supportées par le système, ainsi qu'aux caractéristiques des matières à enseignées.

Deux types de modèles sont proposés:

1. **Modèle de la matière** (Subject matter Model (SM)) : des chercheurs proposent que les contenus du modèle de l'apprenant doivent tenter de représenter le plus fidèlement possible l'état de ces connaissances (réelles) dans le domaine étudié. Un SM est donc un modèle qui contient ces types de connaissances; il contient par exemple, les connaissances qu'il maîtrise, ses incompréhensions, etc. Un exemple de système avec un tel modèle est le 'Lisp Tutor' [Reiser *et al.,* 1985].
2. **Modèle des informations pédagogiques** : parfois les connaissances du domaine sont organisées selon les objectifs pédagogiques. Un PM est donc un modèle qui contient les informations sur l'évaluation des connaissances de l'apprenant par rapport aux connaissances pédagogiques stockées dans le système. Par exemple, WEST [Burton, 82] modélise la force ou la faiblesse de l'étudiant dans les différentes habilités qui sont nécessaires pour le jeu.

Il y a un fossé entre les connaissances de l'apprenant dans le système et les connaissances du tuteur. La question qui se pose lorsque l'on veut modéliser les connaissances de l'apprenant est la suivante: 'comment peut-on établir un pont sur ce fossé?'.
Cette question qui est identifiée par les chercheurs comme étant le problème de diagnostiquer les connaissances de l'apprenant, a attiré l'attention de beaucoup de chercheurs, par exemple on peut mentionner ici, les travaux de VanLehn (1988), Frasson et de La Passadiere (1990), Holt et al. (1994), Verdejo (1994), et Ragnemalm (1996).
L'objet des sections qui suivent est de présenter en détails ce problème et d'examiner quelques approches présentées dans la littérature pour la modélisation de 1'apprenant. Nous commençons tout d'abord par décrire les composants d'un modèle de l'apprenant.

3.2.2 La vue d'un modèle de l'apprenant par ses composants

De façon générale un STI maintient toutes les informations relatives à l'étudiant dans le modèle de l'apprenant. Il reflète la perception qu'a le STI de cet étudiant en modélisant ses forces, ses faiblesses, et son niveau de connaissances du domaine. Nous détaillons ici les contenus de ce modèle en commençant par l'analyse de connaissances de l'apprenant.

3.2.2.1 La connaissance de 1'apprenant

3.2.2.1.1 Types de connaissance de l'apprenant

VanLehn (1988) propose deux types de connaissances utilisées dans les systèmes : les connaissances *déclaratives* (les faits, les schémas, ...) et les connaissances *procédurales* (les règles de production). Une connaissance déclarative ne contient pas son mode d'emploi; par contraste, une connaissance procédurale indique explicitement comment elle est utilisée.

Définition de connaissances (déclaratives et procédurales) : les connaissances

déclaratives sont représentées par des structures de données et sont liées aux savoirs,

tandis que les connaissances procédurales sont représentées par des ensembles de règles qui infèrent des structures déclaratives et sont liées aux savoir-faire [Barr et Davidson, 1981]:

- Un savoir (ce que l'étudiant connaît à propos de la matière, par exemple la définition d'un objet, etc.) est une connaissance déclarative concernant un domaine donné représentée sous forme de faits.
- Un savoir-faire (ce que l'étudiant peut faire, par exemple, la réalisation d'une tache ou la résolution d'un problème) est une connaissance procédurale, concernant un domaine donné, qui réfère à l'utilisation efficace de processus cognitifs et moteurs, relativement stables, dans la réalisation efficace d'une tache.

Définition du modèle de l'apprenant: c'est une base de connaissances qui, par rapport au domaine enseigné, contient les informations relatives:

* aux savoirs acquis et aux savoir-faire maîtrisés par l'apprenant:

- Un savoir acquis est une connaissance que possède à la fois le module expert du domaine et l'apprenant.
- Un savoir-faire maîtrisé est une compétence que possède à la fois le module expert du domaine et l'apprenant.

* aux savoir non-acquis et aux savoir-faire non-maîtrisés par l'apprenant

- Un savoir non-acquis est une connaissance que possède le module expert du domaine, mais que l'apprenant ne possède pas.
- Un savoir-faire non-maîtrisé est une compétence que possède le module expert du domaine, mais que l'apprenant ne possède pas.

* aux incompréhensions de l'apprenant:

- Une incompréhension est une connaissance que l'apprenant possède, mais que le module expert du domaine ne possède pas.

3.2.2.1.2 Synthèse de la connaissance de l'apprenant

Largeur de bande (bandwidth) : VanLehn (1988) précise une notion importante, celle de largeur de bande. Il s'agit de l'ampleur des données sur lesquelles se base le module apprenant pour appliquer les règles de diagnostic. Plus le «bandwidth» est élevé, plus les stratégies d'instruction peuvent être précises.

Selon VanLehn, le modèle de l'apprenant doit faire face à trois dimensions bien particulières de la modélisation de l'apprenant:

1. La première dimension (appelée **«La plage des informations»**) vise la quantité d'informations connue par le système au sujet des agissements de l'apprenant face à un problème. Le minimum d'information consiste en la connaissance de la réponse d'un apprenant face à une question. Mais d'autres activités intermédiaires de l'apprenant peuvent être utiles, comme sa façon d'utiliser le clavier pour répondre. La plage des informations disponibles est repartie selon trois niveaux : les états *mentaux,* les états *intermédiaires* et les états *finaux.* Les états mentaux correspondent aux divers états cognitifs de l'apprenant. Les états intermédiaires correspondent aux diverses étapes de la résolution de problèmes par l'étudiant et les états finaux correspondent aux actions ou aux réponses fournies par lui.
2. La deuxième dimension, appelée type de connaissances, est représentée par une division des connaissances selon qu'elles soient déclaratives ou procédurales, avec une division supplémentaire au niveau de ces dernières (séquentielles ou hiérarchiques).
3. Enfin, la troisième dimension se rapporte à la capacité du système à établir les différences entre l'apprenant et l'expert.

Composants du modèle de l'apprenant

Dans cette section, nous étudions différentes vues d'un modèle de l'apprenant selon les modules qui le composent, par exemple:

- Un premier module représente les connaissances générales de l'apprenant qui ne sont pas directement reliées à la matière enseignée. Ces données peuvent par exemple

représenter l'âge, le nom et l'origine de l'apprenant, etc. On peut aussi conserver ses aptitudes générales, son niveau de stress, etc.

- Un deuxième module est consacré exclusivement à la matière enseignée. C'est là que l'on retrouve le niveau de l'apprenant sur les différentes unités cognitives de la matière. Ce module s'emboîte généralement avec un curriculum représentant l'espace des connaissances à enseigner.
- Enfin, un troisième module, s'occupe de faire la mise à jour continuelle des deux modules précédents afin d'avoir à tout moment une représentation la plus fidèle possible de l'apprenant.

Les deux premiers modules forment la partie statique du modèle de l'apprenant, alors que le troisième module forme la partie dynamique.

En effet, on s'accorde généralement pour séparer le modèle de l'apprenant en différentes parties. Carr et Goldstein (1977) et Clancey (1979) citent par exemple, quatre informations nécessaires pour maintenir ce modèle : la difficulté du sujet de la matière à enseigner, les questions directement posées par l'apprenant, les performances et l'expérience d'apprentissage de l'apprenant.

Self (1987) définit le modèle de l'apprenant comme un 4-tuplet (connaissance procédurale, connaissance conceptuelle,caractéristiques individuelles,l'historique).

Un modèle de l'apprenant comprend toujours des connaissances liées au domaine d'enseignement : ce que l'apprenant sait et ce qu'il sait faire. Dans son état le plus complet, ce modèle comprend aussi des connaissances indépendantes du domaine enseigné. Les unes concernent ses mécanismes d'apprentissage : comment l'apprenant fonctionne-t-il ? comment découvre-t-il de nouveaux concepts ? de nouvelles techniques ? etc. Les autres concernent les stratégies pédagogiques correspondantes : quels sont les types et les modalités d'intervention les plus efficaces ? etc.

Selon Nkambou (1996) le modèle de l'apprenant est identifié en trois parties : une partie cognitive (modèle cognitif), une partie inférentielle (modèle d'inférences), et une partie affective (modèle affectif).

Modèle cognitif : cette partie contient des informations sur l'état des connaissances de l'apprenant par rapport à la matière considérée. Ces informations portent sur:

- *les capacités :* l'information sur les capacités traduit le niveau de l'étudiant par

rapport aux connaissances. Robert M. Gagné (1976) a classifie les capacités apprises en cinq catégories: les informations verbales, les habiletés intellectuelles, les attitudes, les habiletés motrices et les stratégies cognitives.

- *les objectifs :* l'information sur les objectifs traduit le fait que l'étudiant a déjà réalisé ou non un objectif. Par exemple, l'étudiant en médecine a acquis le savoir-faire d'une technique de repérage de la plaque dentaire.
- *les ressources :* l'information sur les ressources (exercices, problèmes, tests, simulations, démonstration, etc.) traduit le fait qu'une ressource a déjà été utilisée par un apprenant, et le contexte dans lequel cette ressource a été utilisée.

  et éventuellement les relations : l'information sur les relations traduit le fait que l'étudiant a réussi ou pas *à* établir une relation (par exemple: de type analogie, abstraction, cas particulier, etc.) entre deux connaissances (et donc, la connaissance d'une relation entre deux connaissances est aussi une connaissance (meta-connaissance)).

Modèle d'inférences : c'est le moteur d'inférences qui tourne tout le temps (ou suite à un avènement) pour ajuster le modèle de l'apprenant. Elle contient des règles qui lui permettent de raisonner sur le modèle cognitif et sur le modèle psychologique (modèle affectif) pour inférer de nouvelles connaissances dans le modèle de 1'apprenant.

Modèle affectif : ce modèle est un ensemble de données permettant de cerner la personnalité et les différentes facettes d'un étudiant. Il contient des connaissances relatives aux caractéristiques particulières permanentes ou momentanées de l'apprenant. Parmi celles-ci, nous avons:

- les connaissances relatives aux conditions mentales, par exemple l'apprenant est spatial ou verbal, il est réfléchi ou impulsif, etc.;
- les connaissances relatives aux sentiments et à la personnalité, par exemple l'apprenant est calme ou anxieux, il est attentif ou distrait, etc.

3.2.3 Processus de Diagnostic

Le processus de diagnostic est divisé en trois phases [Ragnemalm, 1996].

1. *Acquisition des connaissances :* l'acquisition des connaissances est le processus qui effectue l'extraction des connaissances utiles pour la modélisation de l'apprenant.
2. *Transformation* : c'est le processus qui analyse les connaissances acquises dans la phase précédente (ou qui analyse les connaissances du tuteur) afin d'extraire les informations pertinentes et utiles dans le jugement d'habilités de l'apprenant.
3. *Évaluation :* l'évaluation est le processus qui évalue les connaissances ou les conduites de l'apprenant. Cette évaluation se fait par rapport aux connaissances (celles du tuteur) stockées dans le système.

En fait, la fonction de diagnostic permet au STI de mettre à jour les modèles de l'apprenant. Cette fonctionnalité est réalisée par le processus de diagnostic qui utilise différentes sources d'information pour exécuter son travail:

- *des informations implicites :* ces informations s'obtiennent directement en comparant les actions de l'étudiant avec celles de l'expert (exemple: le niveau de connaissances de l'apprenant inféré par le système):
- *des informations structurales* : ces informations proviennent du réseau des

dépendances organisationnelles de la matière (exemple: liens entre les connaissances comme l'analogie, ou bien les liens entre une unité cognitive et ses sous- unité cognitive, etc.);

- *des informations explicites* : ces informations proviennent de l'interaction avec l'apprenant (exemple: demander directement à l'apprenant combien de fois il a exécuté une tache);
- *des informations historiques :* ces informations permettent d'initialiser le modèle de l'apprenant (exemple : les informations statiques comme le nom de l'étudiant, ses aptitudes générales, etc.).

3.2.4 Les approches de modélisation de l'apprenant

Parmi les techniques de modélisation connues, nous pouvons citer:

3.2.4.1 Modèle de Recouvrement

La première approche consiste à construire le modèle de l'apprenant de façon comparative. Les performances de celui-ci sont comparées à un modèle idéal qui représente les connaissances d'un expert de la matière enseignée. Carr et Goldstein (1977) appellent cette technique, la *méthode de recouvrement* des connaissances de l'étudiant «overlay model». Le modèle de l'apprenant est alors considéré comme un sous-ensemble du modèle idéal.

3.2.4.2 Modèle de Reconstruction

Quelques STI se sont également intéressés à la détection des erreurs en utilisant une méthodologie basée sur la reconstruction des erreurs plutôt qu'à partir d'une liste d'erreurs prédéfinie dans une librairie. Dans ce cas, le module de diagnostic utilise deux librairies : une librairie de prédicats et une librairie d'actions. Dans la théorie de réparation, les erreurs sont générées en altérant un groupe de règles adéquates pour créer une situation où aucune règle ne s'applique (i.e. «impasse») [Brown et VanLehn, 1980]. Une réparation est une action qui consiste à contourner une impasse.

3.2.4.3 L'approche de Goldstein (Graphe Génétique)

Goldstein (1982) introduit l'idée de modélisation des connaissances de l'apprenant par l'intermédiaire d'un graphe génétique représentant les relations d'analogie, de généralisation-spécialisation, de raffinement-simplification et de déviation-correction.Le graphe génétique offre une représentation des connaissances procédurales. Les règles d'une procédure sont représentées comme des nœuds, et les liens représentent les relations entre les règles.

3.2.4.4 L'approche Objet (Meurrens)

Meurrens expose une gestion du modèle de l'apprenant à la manière de la programmation orientée objet [Meurrens, 1989]. Il s'agit en fait d'un système auteur permettant le développement de didacticiel dans n'importe quel domaine. L'écriture des règles modélisant l'expert et les stratégies tutorielles incombent à l'auteur du didacticiel, et sont sous forme d'objets ayant des caractéristiques et étant interconnectés entre eux. Le modèle est conservé en mémoire sous la forme d'un tableau de données ou d'un ensemble de règles. La granularité des objets (i.e. leur degré de définition) doit être choisie pour que la plus petite erreur ou mauvaise compréhension d'un élève puisse être détectée. Le système gère en plus pour chaque objet (un objet symbolise une connaissance) deux paramètres : le niveau de maîtrise et le niveau de pertinence. Le niveau de maîtrise sera d'autant plus élevé que l'objet a été plus souvent ou mieux exécuté, compris et utilisé. Plus l'apprenant appliquera une stratégie pédagogique ou comprendra un texte ou atteindra un objectif de manière efficace, plus le niveau de pertinence de l'objet en cause sera élevé.

C'est de cette approche objet qu'on s'est inspiré pour modéliser le profil de l'apprenant dans notre cas. C'est une approche qui s'adapte le mieux à l'architecture distribuée de notre système. Ainsi, pour chaque Unité Cognitive UC impliquée dans le contenu de la formation acheminée, on conçoit un objet qui contrôle la progression de l'apprenant dans UC. Le changement de niveau est contrôlé par l'agent pédagogique qui décide entre une progression ou une régression dans le niveau de maîtrise de l'apprenant.

3.2.5 Présentation du modèle de l 'apprenant dans quelques STI

Dans ce qui suit, nous présentons quelques mode les d'apprenant dans des STI existants:

♦ **Le système SCHOLAR**

Le modèle de l'apprenant dans le système SCHOLAR [Carbonell, 1970b] ne contient que les connaissances que l'on suppose assimilées ainsi que les liens qui les relient. On utilise pour cela, la méthode de recouvrement appliquée sur le réseau sémantique représentant

les connaissances du système. A un instant donné, le modèle de l'apprenant consiste en plusieurs régions distinctes de ce réseau que le tuteur tente de connecter selon une certaine stratégie d'enseignement afin d'unifier les connaissances du système et celles de l'apprenant.

♦ **Le système BIP**

BIP [Barr *et al.,* 1976] utilise un modèle de l'apprenant et un historique de l'étudiant pour la phase de diagnostic du travail de l'étudiant et pour la sélection de sa prochaine tâche. Le modèle de l'apprenant dans ce système est une forme de recouvrement.

♦ **Le système WEST**

Le système WEST [Burton, 1982] emploie un modèle de l'apprenant basé sur le modèle de recouvrement, pour analyser les réponses de l'étudiant. Dans WEST, lors de l'évaluation d'un savoir-faire, le processus de diagnostic est capable de découvrir les erreurs de l'apprenant en comparant ce que ce denier est en train de faire avec ce qu'un expert ferait. Ils cherchent ainsi à pousser l'apprenant à optimiser ses actions et à maintenir son intérêt de participation durant le jeu.

♦ **Les systèmes ALGEBRA et GREATERP**

Les systèmes ALGEBRA [Lantz *et al.,* 1983] et GREATERP [Reiser *et al.,* 1985] sont également des systèmes qui utilisent:

- La représentation sous forme de règles de production pour représenter leurs bases de connaissances.
- La méthode de recouvrement pour représenter leur modèle de l'apprenant.
- Le modèle tuteur de ces systèmes renferme un certain nombre de stratégies sous forme de règles de production et guide constamment l'étudiant dans la résolution de ses problèmes.

♦ **Le système SOPHIE**

La philosophie adoptée dans le modèle de l'apprenant du système SOPHIE [Brown *et al.,* 1982] est un amalgame de la méthode de recouvrement et de détection des concepts erronés de l'étudiant. SOPHIE peut détecter toute inconsistance entre les actions de l'étudiant et la compréhension que le système suppose que celui-ci a du circuit (en utilisant le modèle de l'apprenant). Il décèle les conceptions erronées de l'étudiant au sujet du circuit et de son fonctionnement. Cette technique est appelée : identification des conceptions erronées.

♦ **Le système BUGGY**

BUGGY [Brown et Burton, 1978] utilise un modèle de 1'apprenant, basé sur le modèle correctionnel (avec une librairie d'erreurs). Ce système emploie une librairie contenant les mauvaises conceptions pour le domaine de l'arithmétique (soustraction).

♦ **Le système WHY**

Le système WHY [Stevens *et al.,* 1979] emploie un modèle de l'apprenant basé sur un concept qui consiste à identifier et corriger les conceptions erronées de 1'apprenant (le

modèle correctionnel). Il est à noter que lorsqu'il y a plus d'une erreur dans la réponse de l'étudiant, le système utilise des heuristiques pour décider quelle erreur il va corriger en premier.

♦ **Le système ACM**

Le système ACM [Langley et Ohlsson, 1984] utilise un modèle de reconstruction. II reconstruit la séquence des opérateurs utilisés par l'apprenant à partir d'inférences sur les conditions d'application des opérateurs consistant en une conjonction de prédicats et d'une action élémentaire. En effet, le module de diagnostic utilise deux librairies : une librairie de prédicats et une librairie d'actions.

♦ **Le système PIME**

Le système PIXIE [Sleeman et Brown, 1982] utilise également un modèle de l'apprenant basé sur le modèle de reconstruction. Il genre des règles incorrectes en introduisant une action incorrecte de l'apprenant dans une procédure correcte.

♦ **Le système Lisp Tutor**

Le 'Lisp Tutor' [Reiser *et al.,* 1985] maintient un modèle idéal de l'étudiant qui représente les «bonnes» réponses aux divers problèmes que le système peut poser à l'étudiant, ainsi qu'un modèle d'erreurs contenant les différentes déviations de l'étudiant par rapport au modèle idéal.

♦ **Le système Geometry Tutor**

«Geometry Tutor» [Anderson *et al.,* 1985] est basé sur un modèle idéal indiquant la façon dont les apprenants devraient résoudre les problèmes soumis, et un modèle erroné représentant les erreurs typiques commises par des débutants. Le système guide l'étudiant dans la preuve des théorèmes en géométrie en lui donnant une rétroaction chaque fois qu'il est nécessaire. Les caractéristiques principales du style d'interaction de «Geometry Tutor» peuvent se résumer comme suit:

- A chaque étape de la construction d'une démonstration par l'apprenant, le tuteur génère toutes les possibilités d'application d'une règle dont il dispose.
- Il compare ensuite le pas de démonstration de l'apprenant avec celles-ci.
- S'il ne correspond à aucune possibilité, «Geometry Tutor» demande la modification du pas de démonstration.
- Au bout d'un certain nombre d'essais infructueux, il construit lui-même le pas par rapport à ce qu'il considère être la meilleure règle.

3.3 Conclusion

Dans ce chapitre, nous avons procédé à une description des différents composant de la matière à acheminer pour la formation à distance. Nous retiendrons essentiellement l'unité cognitive qui va constituer la partie centrale dans toute activité qui sera programmée par l'agent pédagogique lors d'une session d'apprentissage.

Ce chapitre nous a permis également d'analyser le modèle de l'apprenant. Nous retiendrons par la suite, les points importants suivants:

- Le rôle du modèle de l'apprenant est central au STI, puisque ce composant est la clé de la formation individualisée, dans l'interaction avec l'étudiant.
- La modélisation de l'apprenant a été identifiée comme une tâche complexe, difficile mais importante par les chercheurs.
- Plusieurs techniques de modélisation de l'apprenant existent, mais toutes ces approches ne permettent pas réellement de représenter fidèlement l'état des connaissances de l'apprenant en situation de formation.

Pour mettre en pratique notre modélisation, nous allons décrire plus en détaille dans le prochain chapitre la méthodologie adoptée avec les différents modules reliés au processus de restructuration et de mise en oeuvre de la matière à enseignée dans un environnement distribué.

CHAPITRE IV

ENVIRONNEMENT DE DEVELOPPEMENT DE COURS

L'objectif de notre démarche est de fournir un environnement complet de développement de Cours. En se référant aux environnements développés dans le cadre du projet SAFARI, il nous apparaît nécessaire d'apporter des améliorations dans le but de :

- faciliter les opérations de modélisation,
- faciliter le partage et la réutilisation des connaissances,
- augmenter l'adaptabilité de la formation,
- organiser le processus de conception.

L'environnement de développement de cours que nous proposons se compose de trois modules relatifs à la gestion globale de la conception, la gestion des composantes de cours et la génération automatisée de cours.

4.1 Environnement distribué de conception

Pour profiter pleinement de la richesse des technologies distribuées : Remote Method Invocation (RMI) [Sun Microsystems, www.javasoft.com] (expliqué plus en détail dans le chapitre 5), Système de gestion des bases de données distribuées (Oracle comme exemple) [Oracle, www.oracle.com], nous avons jugé très utile de développer notre environnement dans le cadre d'une architecture distribuée. Cet environnement permettra sans doute d'augmenter considérablement la productivité et la collaboration des différents concepteurs qui peuvent bien travailler sur même sujet ou sur des sujets différents en réutilisant des composants communs de la base universelle de connaissances (voir figure 7).

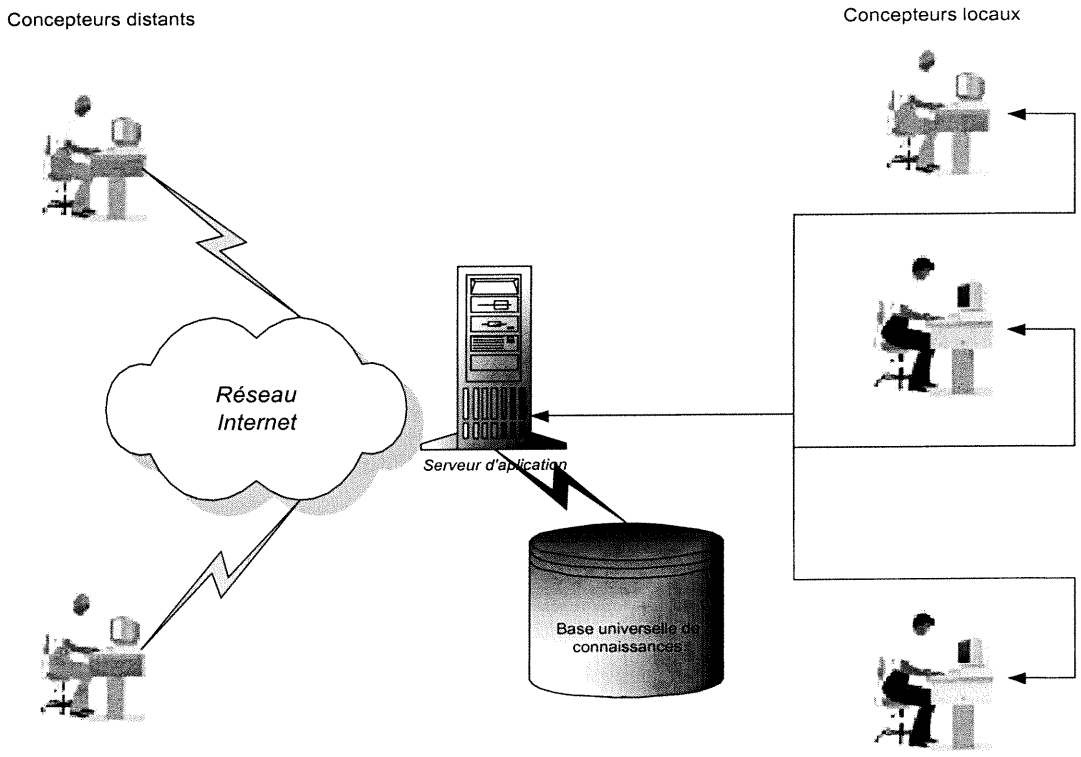


Figure 7 : Environnement distribué de conception

4.2 Gestion globale de la conception

La plupart des approches suivies pour la mise en œuvre d'un environnement de développement de cours dans le cadre des STI considère le processus de modélisation dans un contexte mono-concepteur, mono-curriculum. Or, dans un contexte de construction de STI à grande échelle dans des institutions multidisciplinaires comme les universités ou les instituts de formation professionnelle, la tâche de modélisation des connaissances n'est généralement pas une tâche individuelle, mais souvent un processus collectif avec plusieurs équipes de conception apportant chacune l'expertise dans son domaine d'activité. De plus les domaines de connaissances ne sont pas, dans la plupart des cas, des domaines strictement disjoints.

Le fait de considérer dans notre approche cette modélisation comme étant un processus collectif portant sur plusieurs cours, nous a emmené à introduire dans notre environnement des solutions et mécanismes pour la gestion globale de la conception, parmi lesquels : la base de connaissances globale, l'organisation hiérarchisée de la connaissance, l'unification des vocabulaires et la gestion des droits d'accès.

4.2.1 Base globale de connaissances

Dans l'environnement qu'on propose, on tente de favoriser au maximum le partage et la réutilisation des connaissances mises dans le système. Par exemple, l'unité cognitive "limites de fonctions" défini dans une modélisation du domaine des integrations de fonctions (les mathématiques) peut bien servir dans une modélisation du domaine des Mouvements des solides (la physique) ; plus encore le concept "héritage" défini dans une modélisation du domaine de l'approche orienté objet peut bien servir dans une modélisation du domaine des langages de programmation orienté objet (exemple C++). On ne considère plus les connaissances modélisées comme appartenant à un domaine en particulier (mathématique, physique, informatique,...), mais bien à un ensemble plus vaste qui regroupe toutes les connaissances et expertises des différents domaines réunis, et qu'on a nommé: ***Base globale de connaissances***. Dans nos recherches, nous expérimenterons par rapport au domaine informatique, mais le principe est applicable à tous les domaines.

De ce fait, les domaines d'expertises ne sont plus considérés de façon unitaire et isolée mais deviennent des vues particulières sur un ensemble de connaissances unifié. Le fait de procéder ainsi procure plusieurs avantages :

- ***Élimination de la redondance*** : toute connaissance liée à l'expertise du domaine ou à celle de la pédagogie n'est définie qu'une et une seule fois dans tout le système. Ce qui élimine le problème de redondance et facilite les opérations de mise à jour.

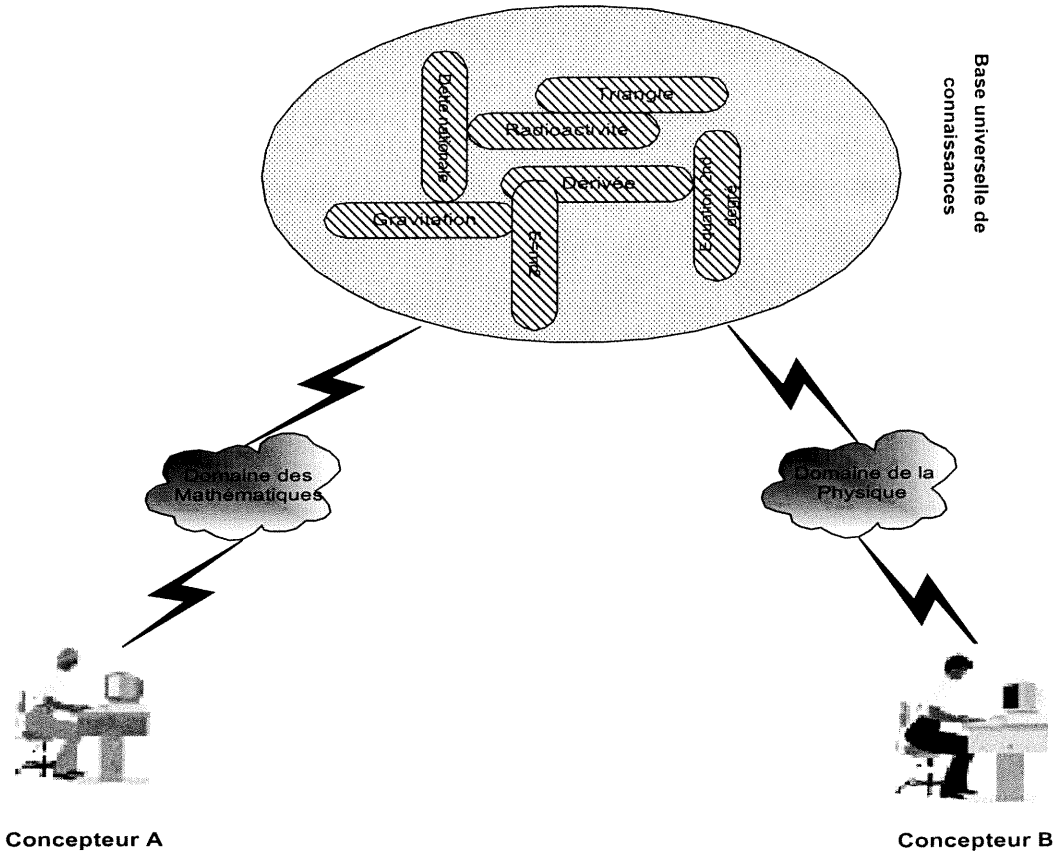


Figure 8 : Partage et réutilisation des connaissances

- ***Optimisation des coûts de réalisation*** : l'analyse et la modélisation des connaissances est une étape longue et coûteuse. Mais le fait de réutiliser une connaissance déjà modélisée dans le système diminue considérablement son coût de production initial.

- ***Augmentation progressive de la productivité*** : les domaines d'expertise sont généralement plus au moins interconnectés. Les nouveaux domaines à construire seront rapidement mis au point par réutilisation des connaissances des domaines déjà modélisés et qui leur sont proches. Ceci augmentera la productivité des équipes chargées de la conception. Par exemple dans un cours portant sur un système de gestion des bases de données (exemple, ORACLE), le concepteur n'aura qu'a réutiliser l'unité cognitive relative à SQL (Structured Query Langage) déjà définie dans le domaine des bases de données.

- ***Augmentation de la qualité générale*** : Un concepteur peut aller chercher dans la base globale de connaissances les connaissances qui lui font défaut dans son domaine d'expertise et qui sont déjà définies dans le système par d'autres concepteurs ayant cette expertise manquante. Ce facteur influence favorablement sur la qualité générale de sa conception et par conséquent, sur celle du système en entier.

4.2.2 Gestion des droits d'accès

Pour maximiser le partage et la réutilisation des connaissances dans le système, tout en minimisant les problèmes dus aux chevauchements entres les travaux, une gestion minimale des opérations d'accès est nécessaire. Pour cela, on considère les opérations de consultations et les références de réutilisation de cette connaissance comme étant des opérations libres et permises à tous les concepteurs travaillants sur le système. Mais les opérations de mise à jour de cette connaissance (modification et suppression) sont des opérations à accès contrôlé (Figure 9). Car une modification dans une connaissance faite par un concepteur donné, peut être parfaitement pertinente pour son travail, mais complètement incorrecte pour un autre qui réutilise cette même connaissance.

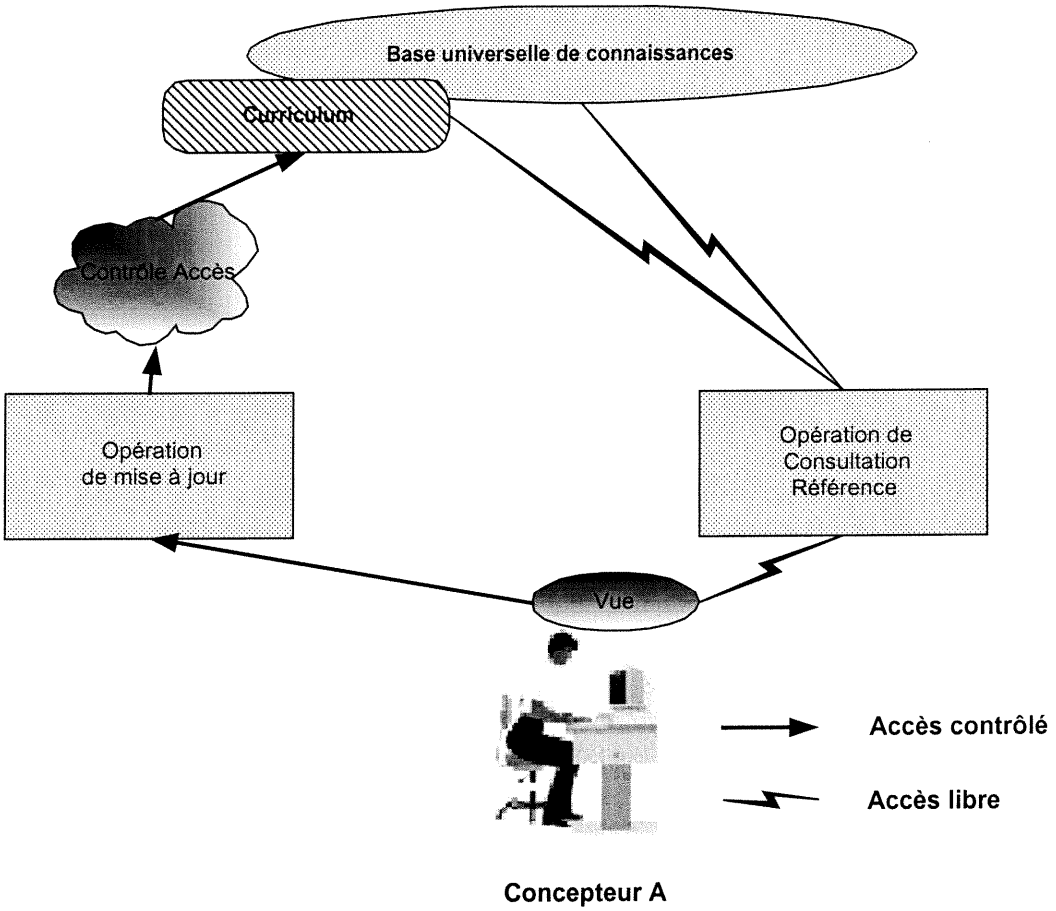


Figure 9 : Contrôle des opérations de mise à jour

4.3 Gestion des composants de cours

L'environnement de développement de cours (Figure 10) intègre l'ensemble des modules de la gestion des composants de cours. Orienté multiconcepteurs, ces modules doivent favoriser la réutilisation des composants tout en contrôlant l'intégrité de la base de connaissances.

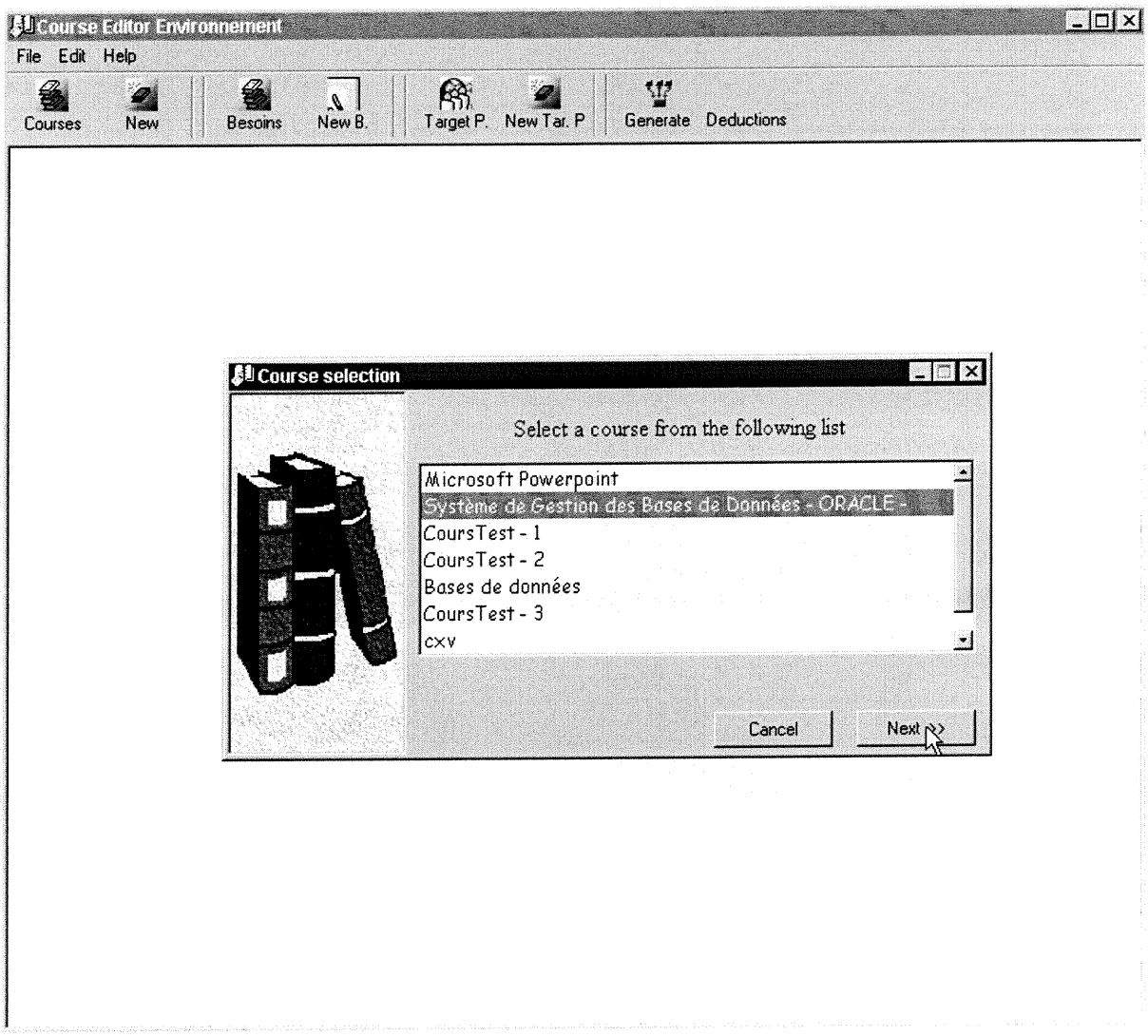


Figure 10 : Environnement de développement de cours

4.3.1 Gestion des unités cognitives

Le module de la gestion des unités cognitives (Figure 11) se base sur une interface simple et intuitive pour le développement de toutes les propriétés relatives à l'unité cognitive qui est un élément central dans la base globale de connaissance. Une organisation hiérarchique des unités définies est respectée au fur et à mesure de l'enrichissement de la base. Ainsi, un cours est tout simplement un réseau d'unités cognitives. On définit le cours C comme suit :

$$C = (\cup (UC_1), \cup (UC_2), \cup \ldots\ldots \cup (UC_n))$$

Avec UC_i = unité cognitive racine d'ordre i dans la hiérarchie du cours C.

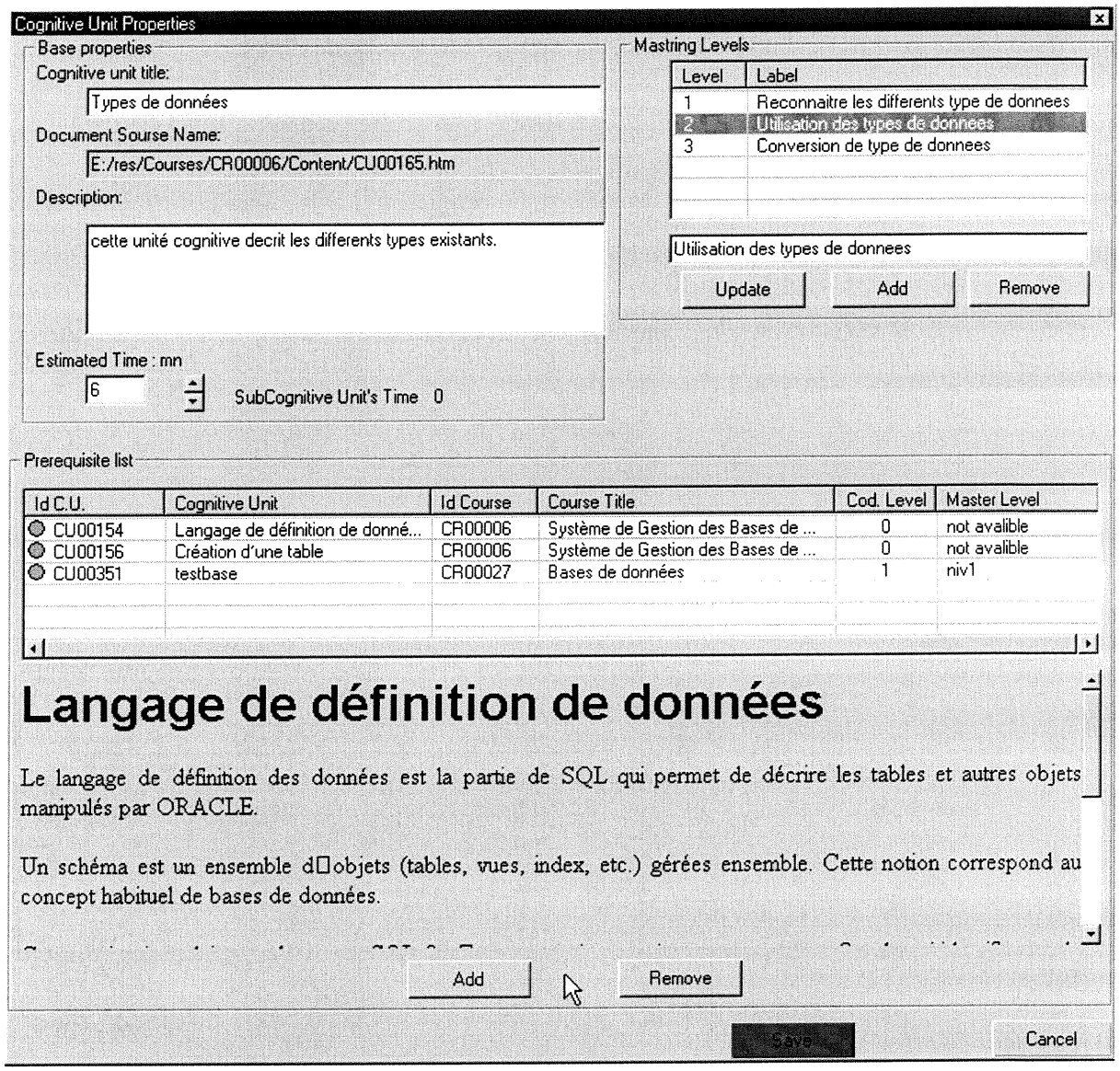


Figure 11 : Gestion des propriétés de l'UC

Algorithme de chargement d'un Cours (C) de la base :

DÉBUT

Soit EUCR l'ensemble des Unités Cognitives Racines pour le Cours C
Pour tout UC dans EUCR faire{

// On appelle une fonction récursive ajouter_uc_tostructure qui prend une unité cognitive comme argument.

Ajouter_UC_ToStructure(UC) ;

}

FIN

Ajouter_UC_ToStructure(Unité Cognitive UC) {

Ajouter l'unité cognitive UC dans la structure arborescente du cours.

Charger les niveaux de maîtrise relatifs à l'unité cognitive UC.

Ajouter les sous unités cognitives de UC en appelant la fonction par l'appel suivant :

Ajouter_Sous_UC(UC)

}

Ajouter_Sous_UC(UC){

Soit ESUC l'Eensemble des Sous Unités Cognitives de UC
Pour tout SUC dans ESUC faire{

// appel de la fonction Ajouter_UC_ToStructure

Ajouter_UC_ToStructure(SUC)

}

}

L'opération d'ajout d'une unité cognitive dans un curriculum fait appel à deux sources de données : soit on fait juste une réutilisation d'une connaissance déjà existante dans la

base, soit la création de toute une nouvelle unité cognitive et auquel cas, il faut développer la ressource didactique de celle ci.

4.3.2 Gestion des concepts

Dans l'objectif de faciliter la maîtrise par l'apprenant de l'unité cognitive, l'auteur de cours est appelé à définir un réseau de concepts au sein de chaque UC. Les concepts définis doivent figurer dans la ressource didactique liée à l'unité cognitive. Chaque concept est un objet défini par un titre, un identifiant unique dans la base de connaissance et la fréquence d'apparition dans la ressource. Pour chaque concept le concepteur définit un groupe d'explications qui sera exploité par l'agent explicatif lors d'une session d'apprentissage. Un concept est à la propriété d'une seule unité cognitive, cependant, il peut être référencé par n'importe quelle autre unité cognitive dans la base de connaissance.

La gestion des concepts au sein d'une ressource didactique est fait par une simple opération de " **Drag & Drop** " entre celle ci et l'UC. La figure 12 montre un exemple de définition des concepts SQL et ANSI. Le module de la gestion des concepts permet réutiliser la même définition aux différentes occurrences du même concept dans la même ressource didactique.

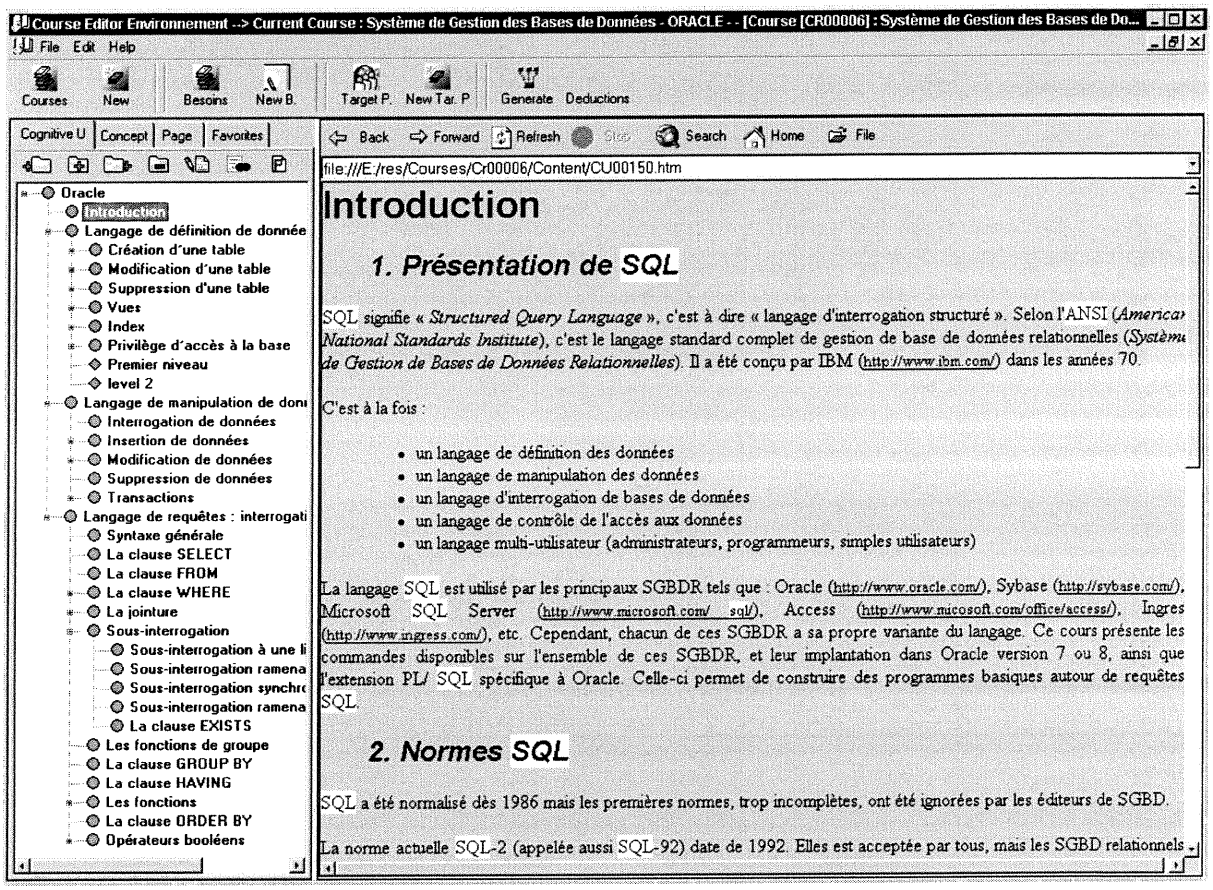


Figure 12 : Gestion des Concepts SQL et ANSI

4.4 Processus de génération automatique de cours adapté

Un cours adapté est un réseau d'unités cognitives, formant une suite d'activités qui permet l'acquisition et la maîtrise d'un ensemble de connaissances. Il répond à un besoin de formation spécifique, pour un apprenant (ou groupe d'apprenants) spécifique.

La génération automatique de cours est un processus qui, en fonction d'un besoin de formation donné et en fonction du public ciblé par cette formation, génère la suite d'unités cognitives dont la réalisation permettra de satisfaire ces besoins. Ce processus se compose des étapes successives suivantes : la spécification des besoins en terme de formation, la spécification du public cible, la génération de la structure du cours.

4.4.1 Spécification des besoins en terme de formation

Le but de toute formation est l'acquisition d'un ensemble de compétences par un apprenant (ou groupe d'apprenants) donné.

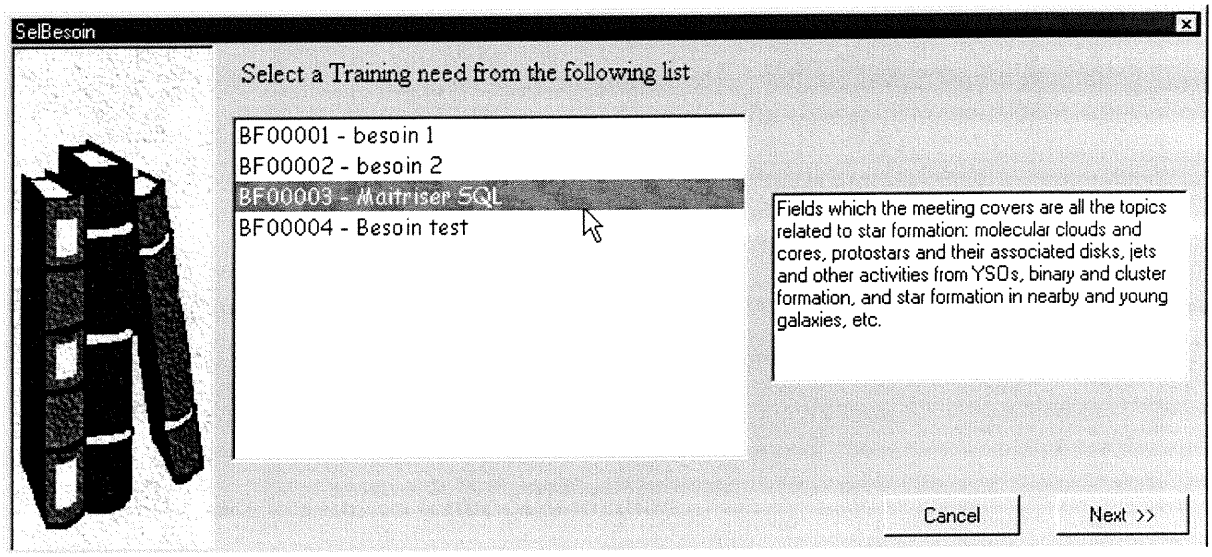


Figure 13 : Sélection d'un besoin déjà défini

La spécification des besoins de formation est l'opération qui permet de désigner les objectifs à atteindre par un public bien ciblé. Le processus de spécification est définit par les deux étapes suivantes :

Première Etape :

Du réseau des unités cognitives formant le cours initial complet défini au départ par un concepteur (éventuellement par un groupe de concepteurs), on choisit les unités cognitives ciblées par les objectifs de la formation en question. La suite des unités cognitives spécifiées hérite la même structure et les mêmes liens définis au niveau du cours de départ. Bien évidement l'utilisateur a accès aux différentes fonctions de mise à jour de la suite sélectionnée.

Deuxième Etape :

- Cette étape consiste à définir le niveau de maîtrise à atteindre pour chaque unité cognitive supportant un contrôle. Le niveau choisi par défaut est le niveau inférieur. On peut toujours redéfinir les niveaux de maîtrise en accédant en mode modification au contenu de la formation en question.

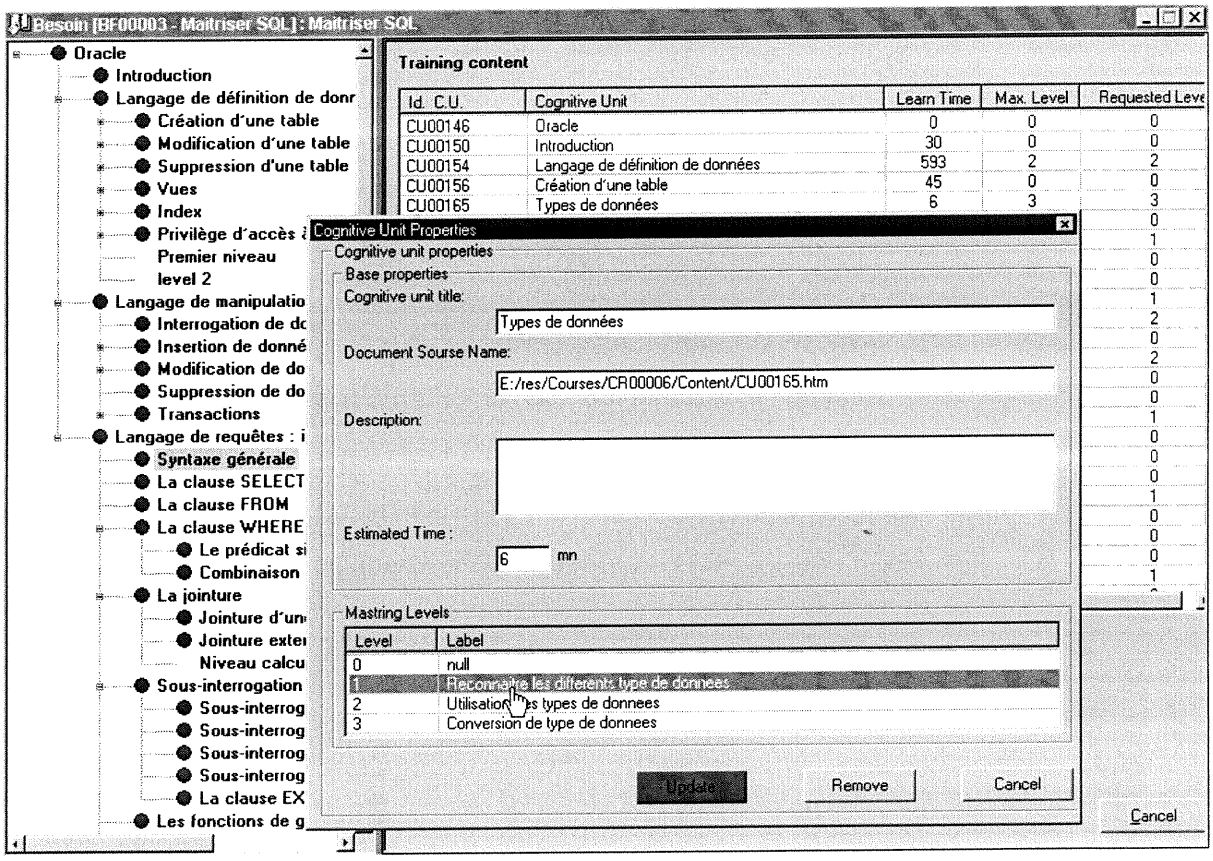


Figure 14 : Accès en modification aux niveaux de maîtrise

Le résultat de ce processus est un ensemble **BF** (voir notation ci-après) de couples bien structurés.

$$\mathbf{BF = \{ (UC_i, N_i) \in C \times [0..N_{\iota max}] \mid \quad 0 <= Ni < N_{max} , 0 < i < N_{maxuc} \}}$$

Avec :

$\mathbf{N_{max}}$: le niveau maximal de maîtrise définit au niveau de l'unité cognitive $\mathbf{UC_i}$.

$\mathbf{N_{maxuc}}$: Le nombre d'unités cognitives composant le cours **C**.

4.4.2 Spécification du public cible

Au moment de la préparation d'un cours adapté, on ne connaît pas, dans la plupart des cas, le niveau exact de connaissances que possède le public ciblé par cette formation. De plus, un cours est généralement destiné à un groupe d'apprenants dont les niveaux de connaissance propres à chacun, peuvent varier considérablement.

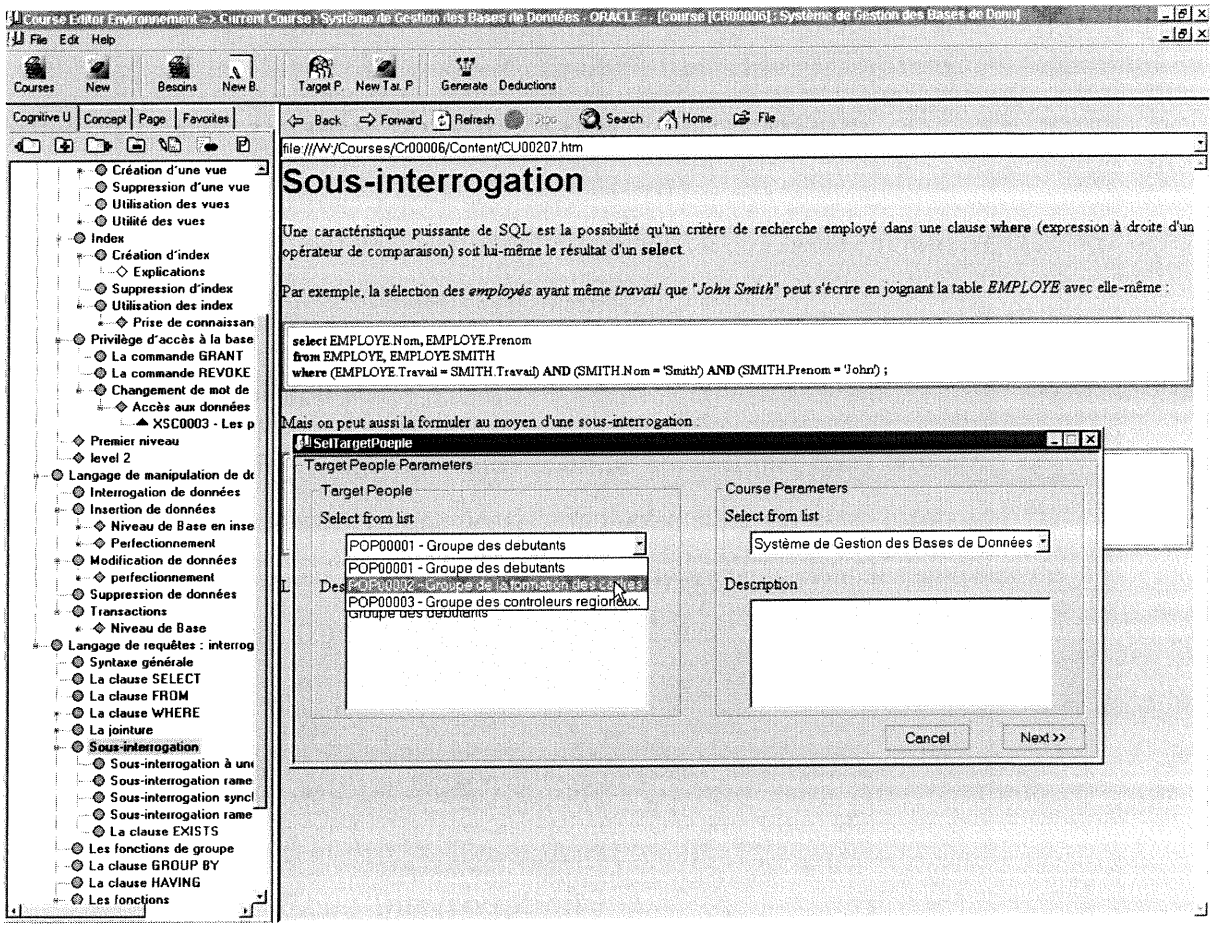


Figure 15 : spécification du public cible

Pour ces raisons, on procède à l'opération de spécification du public cible, qui nous permet de dégager un profil cognitif modèle des futurs apprenants pour ce cours. Il s'agit d'une spécification du niveau moyen de connaissance qui consiste aux opérations suivantes :

- Désigner les unités cognitives du cours, dont le public cible peut avoir une certaine maîtrise.
- Détermination du niveau moyen de maîtrise pour chaque unité cognitive impliquée.

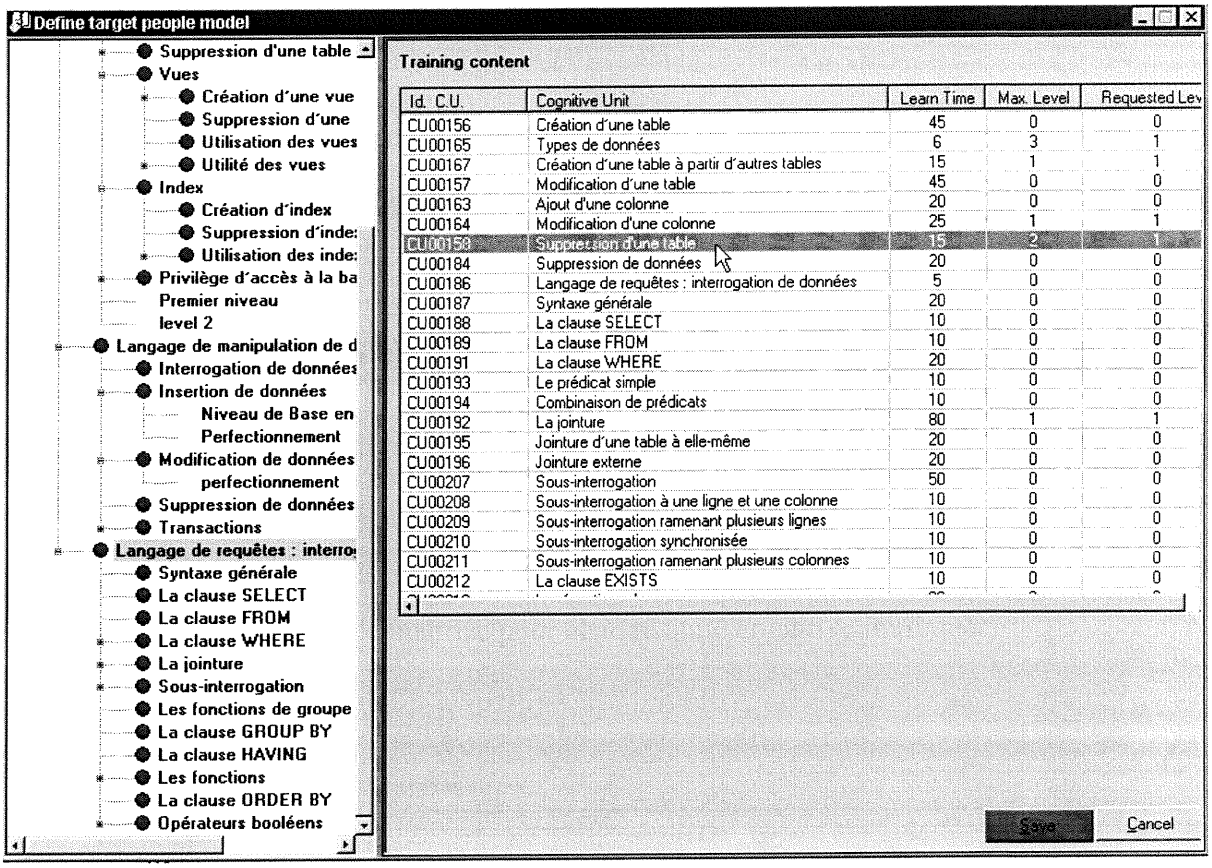


Figure 16 : Définition du modèle type du public cible

Cette spécification du public cible n'est pas tout a fait réaliste. Elle pénalise les apprenants ayant un niveau plus bas, en même temps elle favorise ceux ayant un niveau plus avancé. Pour palier à cette lacune et pour pouvoir individualiser le contenu du cours généré dans notre environnement, on a ajouté un mécanisme d'adaptation et d'individualisation, expliqué dans ce qui suit, du cours en permettant à l'apprenant qui

vient de se connecter pour la première fois au système d'indiquer son modèle cognitif exacte par rapport au contenu de la formation en question.

4.4.3 Génération du cours résultat

En partant de la spécification des besoins en terme de formation, de la définition du public cible et des connaissances du curriculum, le processus de génération du graphe de cours tente de générer et d'ordonnancer l'ensemble des unités cognitives du cours ciblé. La maîtrise et la réussite des activités déterminées permettent à un membre du public cible, d'atteindre les buts spécifiés au départ dans les besoins de formation.

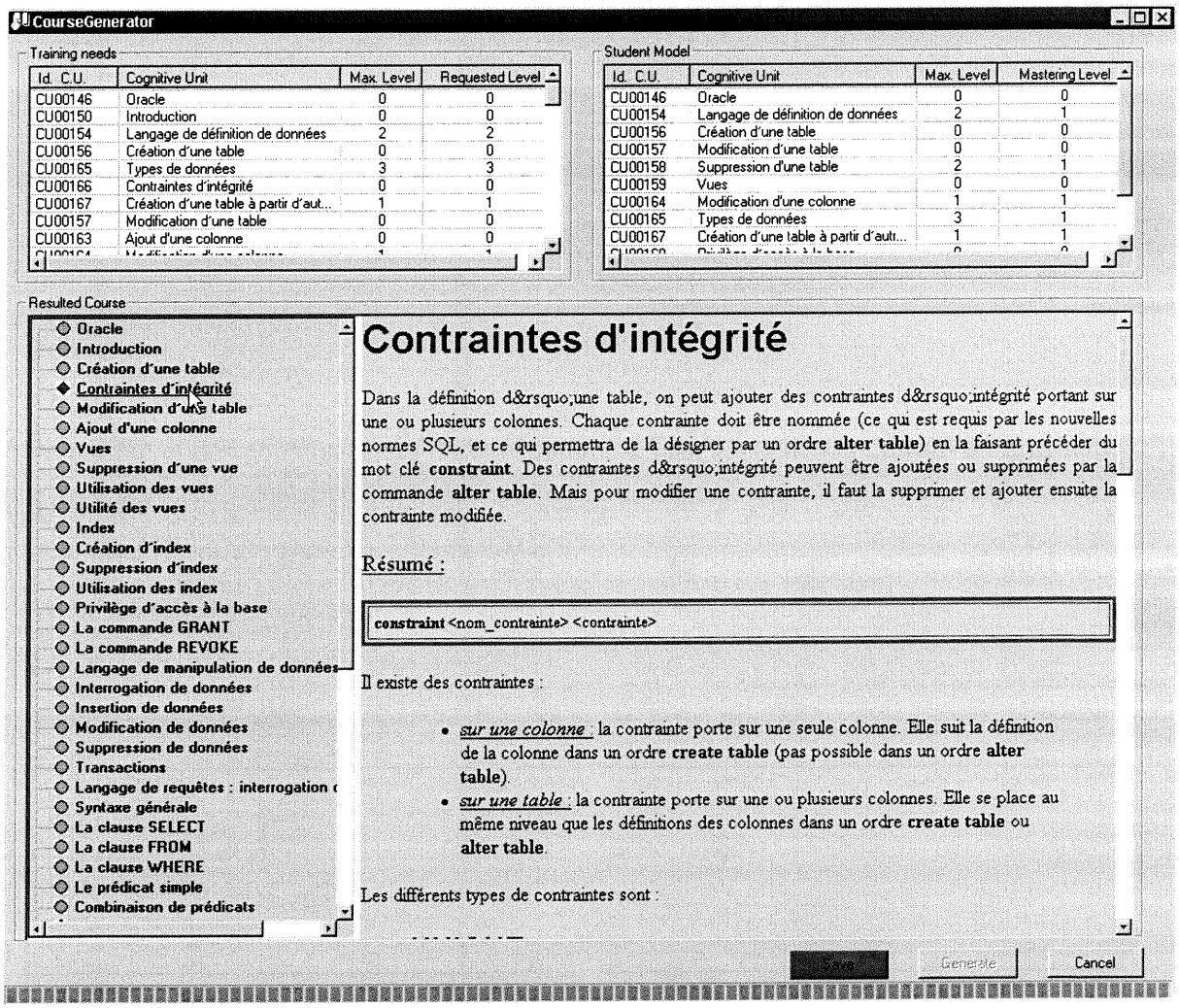


Figure 17 : Génération automatique du cours adapté

Soit G cette fonction de génération de graphe de cours. On note :

$\mathbf{G\ (B, P, C_e) = C_r}$

Avec :

- B : Besoins spécifiés en terme de formation. $\mathbf{B = (UC_i, N_i)}$ avec UC_i est l'unité cognitive a maîtriser et N_i est le niveau souhaité.
- P : Public cible. P est l'ensemble des couples (Unité Cognitive, Niveau de maîtrise) qui donne le niveau de connaissance du public cible.
- C_e : Cours en entré. C'est le cours choisi pour la génération du cours adapté.
- C_r : Cours résultat . C'est le cours résultat du processus de génération. C'est la valeur retournée par le fonction G.

Principe de la fonction de génération de cours G :

Les besoins de formation sont spécifiés en terme de niveaux de maîtrise à atteindre. Le principe du processus de génération automatique du cours est de considérer chaque élément de cette spécification pour le comparer avec le profil cognitif de l'apprenant cible. Le résultat de cette comparaison permet de dégager les deux cas de figures suivantes :

1- Le niveau de l'apprenant dépasse le niveau déterminé dans la spécification. Dans ce cas on considère l'unité cognitive en question comme acquise et elle ne sera pas prise en charge dans la planification.
2- L'apprenant ne maîtrise pas l'unité cognitive au niveau désiré. Dans ce deuxième cas, le système inclut l'unité en question dans le cours généré et on procède aux traitements des prérequis de l'unité cognitive considérée.

4.5 Conclusion

L'environnement de développement de cours est basé sur une modélisation flexible, orienté multi-concepteurs et adaptative :

► Une modélisation flexible, car les connaissances du curriculum (unités cognitives, concepts, ressources didactiques, explications et exercices) sont représentées sous une forme déclarative.

► Une modélisation adaptative, car elle peut être adaptée aux besoins d'un apprenant particulier en supportant la génération du cours individualisé et les activités d'apprentissage selon les connaissances, les besoins et les préférences de ce dernier.

Notre environnement permet l'exploitation d'un curriculum par un STI pour la génération de cours, la planification de l'enseignement et supporte le déroulement d'un cours complet. Il répond à deux caractéristiques fondamentales dans la théorie du design pédagogique: la représentation du contenu de la matière à enseigner et l'organisation de ce contenu.

► Une modélisation orientée multi-concepteurs, car on a la possibilité d'avoir des environnements de conception distribués. En se basant sur les capacités des systèmes de gestion de base de données dans le domaine des bases de données distribuées, notre environnement de développement de cours peut servir comme application distribuée. Le système de gestion de base de données utilisé fera abstraction à la distance. Ainsi, des équipes de conception géographiquement éloignées peuvent collaborer à la réalisation d'un même cours (ou ensemble de cours) en accédant et partageant la même base universelle de connaissances.

CHAPITRE V

ENVIRONNEMENT D'ENSEIGNEMENT

5.1- Les Environnements d'Apprentissage

Le concept d'environnement d'apprentissage permet de dépasser l'opposition simpliste et manichéenne entre les défenseurs de l'apprentissage par induction (résultant des seules activités exploratoires du sujet) et les partisans des tutoriels inspirés (de près ou de loin) par l'enseignement programmé (caricature de l'apprentissage par enseignement). Ces systèmes doivent être capables de favoriser l'acquisition de concepts et de procédures associés à un domaine de connaissance. Il s'agit ni plus ni moins de construire ce que l'on pourrait appeler "une station de travail thématique" que chaque élève (ou chaque maître) pourrait adapter à son goût et à ses besoins.

Voici quelques définition d'un environnement d'apprentissage :

- Un environnement d'apprentissage est un système qui réalise la synthèse entre, d'une part, les avantages de l'exploration libre et de la construction progressive des objets de connaissance et, d'autre part, l'intérêt du guidage propre aux systèmes tutoriels. L'idée centrale est de permettre à l'apprenant de transformer rapidement et efficacement ses expériences en connaissances organisées.
- Un environnement d'apprentissage est aussi un système qui privilégie l'idée que la meilleure façon d'apprendre c'est de se retrouver dans une situation (quasi) réelle de

conception et de travail. Plutôt que de construire des logiciels orientés sur l'explicitation formelle de connaissances scolaires, nous pensons qu'il est maintenant possible de concevoir des outils et des environnements qui assisteraient l'élève efficacement dans les problèmes auxquels il doit faire face dans sa carrière d'apprenant (Brown, 1989). Il apparaît que l'apprentissage incident de l'ensemble des connaissances nécessaires à la résolution d'une tâche pose moins de problème de motivation ou d'attention, si l'intérêt pour la tâche est assuré à un niveau élevé.

- Un environnement d'apprentissage est enfin un logiciel qui permet d'entraîner un certain nombre de procédures de "calcul" pour acquérir les automatismes nécessaires à la résolution de problèmes récurrents (orthographe, opérations arithmétiques, décisions statistiques, etc). Il doit aussi prendre en compte les champs conceptuels du domaine et les définitions de termes pour donner à l'apprenant la possibilité de communiquer efficacement avec les spécialistes d'un champ de compétence.

5.2 Architecture orientée WEB

En dépit de l'effet bénéfique bien connu et significatif des cours particuliers individuels comparé à ceux donnés en classe standard [Bloom,1984], faire doter chaque apprenant d'un logiciel autonome qui fonctionne sur une machine simple est presque impossible.
Le développement d'un STI est une opération très coûteuse aussi bien en terme de budget que de temps.

La diffusion des applications liés aux STI à de grandes assistances est logistiquement difficile. Cependant, le World Wide WEB procure une occasion sans précédent aussi bien pour les concepteurs de cours que pour les apprenants. Avec la prolifération de l'accès au WEB, nous avons maintenant des conditions appropriées pour apporter un STI traditionnel sur un environnement distribué et bien hétérogène [Alpert, Ksingley et Peterf, 2000].

Pour développer notre solution WEB on a trois pistes à explorer :

1- Client indépendant : le tuteur (le module avec le quel réagit l'apprenant) est intégré complètement dans une applet java qui s'exécutera après sur le poste client (machine apprenant). Toute la logique du tuteur est géré en local.
2- Architecture HTML-CGI : L'utilisateur interagit avec des pages HTML dans un browser ; les informations fournies par l 'apprenant sont transmis au serveur qui appelle le programme CGI qui répond alors avec de nouvelles pages HTML. Toutes la logique réside sur le serveur.
3- Architecture distribuée Client Serveur : Une applet java (légère) contient la partie d'interaction d'utilisateur et communique directement avec des serveurs en utilisant le protocole RMI (Remote Method Invocation) qui sera détaillé dans la prochaine section. La logique du système est partagée entre le client et le serveur.

Nous avons opté pour le choix de cette dernière option pour des raisons d'efficacité et de flexibilité.

Notre architecture rapporte un autre avantage principal qui est spécifique à ITS: le modèle de l'apprenant pour tous les utilisateurs réside dans un emplacement simple, sur le serveur centralisé, par opposition à avoir des modèles d'apprenants pour différents individus sur plusieurs différentes machines (dans un laboratoire d'ordinateur d'école ou dans toute une école, par exemple). Ainsi, bien que les apprenants puissent entrer de n'importe où, le système (via notre agent pédagogique) est toujours capable d'identifier le niveau de connaissance actuelle de chaque apprenant et le place dans la bonne position dans le cours en question.

5.4 Le choix de Java

Étant donné la décision pour déployer notre système sur l'Internet, on pourrait choisir une mise en place impliquant une architecture HTML-CGI. Le CGI représente une architecture généralement utilisée dans le WEB. Dans l'approche de CGI, chaque fois qu'un serveur de HTTP reçoit l'information d'un browser qui est destiné à une application particulière de CGI, ces données sont expédiées au programme de CGI qui s'exécute sur le serveur. Beaucoup de serveurs de HTTP relancent le programme de CGI chaque fois qu'ils reçoivent des données pour ce programme. Le module CGI répond aux utilisateurs en envoyant de nouvelles pages de HTML au serveur WEB qui transmet alternativement le HTML au browser de client. Dans tous les cas, il n'y a aucune communication directe entre le client et l'application CGI; l'information est conduite par le serveur WEB.
Il faut noter qu'il y a un certain nombre de contraintes associées au déploiement de notre système sur le WEB. Premièrement, et c'est peut-être la plus importante, nous voulons donner à l'apprenant une expérience fortement interactive avec l'environnement de formation chose qui est très limitée par ce que HTML pourrait fournir.

Nous souhaitons incorporer les dispositifs d'interface qui seraient impossibles ou fortement impraticables dans le HTML, tels que la représentation progressive et conviviale du profil de l'apprenant, le feed-back immédiat par l'intermédiaire des médias multiples pour chaque activité réalisée dans le cours objet de la formation et bien d'autres composantes graphiques interactives qui répondent immédiatement aux actions d'utilisateur. Pour mettre en application ces dernières fonctionnalités et d'autres capacités dans l'interface usager, nous avons développé une solution qui permet de faire fonctionner une partie du code sur le poste apprenant d'une façon complètement indépendante de la partie résidante sur le serveur.

5.5 Les objets distribués : Remote Method Invocation (RMI)

L'invocation d'une méthode à distance sur un objet distant (RMI) est un modèle d'objet distribué qui garde la sémantique du modèle objet Java. Le RMI permet d'utiliser et d'implémenter d'une manière efficace des objets distribués.
Le système RMI est spécialement conçu pour opérer dans l'environnement Java. CORBA présume un environnement hétérogène et multi-langage, le RMI est son pendant dans un environnement homogène Java (machine virtuelle de Sun Microsystems).
Les objectifs du RMI sont:
- de supporter l'invocation à distance d'un objet dans différentes machines virtuelles,
- de supporter le callback des serveurs vers les applets,
- d'intégrer le modèle objet distribué en java sans changer de sémantique,
- de faire disparaître les différences entre les modèles d'objets distribués et le modèle local,
- de rendre simple la liaison entre des applications distribuées,
- de préserver la sécurité de l'environnement d'exécution java (run time).

Le système RMI gère un ensemble de services liés à la gestion des objets distribués :
- le garbage des objets distants,
- la réplication des serveurs et des objets,
- l'activation des objets persistants.

5.5.1. Modèle objet distribué.

Le RMI est l'action d'invoquer une méthode d'une interface distante sur un objet distant.
Ils existent malgré tout plusieurs différences entre le modèle distribué et le modèle non distribué :

- les clients des objets distants interagissent avec des interfaces distantes, jamais avec l'implémentation des classes de ces interfaces,

- les arguments sont passés par valeur (copie) plutôt que par références,
  objet distant est passé par référence, non par copie de son implémentation distante,

- un objet distant est passé par référence, non par copie de son implémentation distante,
- la sémantique de quelques unes des méthodes de la classe Objet sont spécialisées pour les objets distants,
- des exceptions spécialisées gèrent les erreurs.

Dans le modèle d'objet distribué, le client interagit avec un objet stub (représentant local qui gère l'accès à l'objet distant) qui a exactement le même ensemble d'interface que la classe de l'objet distant. Les stubs sont générés par le compilateur rmic.

5.5.2. Architecture.

L'architecture du système RMI est une architecture à trois niveaux (voir figure 17). Le niveau supérieur Stub/Skeleton est l'interface entre le niveau application et le reste du système RMI. Ce niveau transmet les données au niveau inférieur Remote Reference Layer qui gère les références distantes des objets.

Le stub est le proxy client d'un objet distant. Il permet l'appel, la mise en forme des arguments (qui utilise la sérialisation), l'invocation d'une méthode et la fermeture de l'appel. Le Skeleton est le serveur sur la machine distante qui gère l'objet distribué et dont la fonction est de distribuer les appels à l'implémentation effectif de l'objet distant. Les classes stub et skeleton appropriées sont déterminées au run time et chargées dynamiquement si besoin.

Le niveau intermédiaire Remote Reference Layer traite avec le niveau inférieur de transport. Il est responsable de la gestion des protocoles spécifiques qui sont indépendants des stub et skeleton. Par exemple : invocation point à point, invocation à un groupe d'objets répliqués.

Le niveau Transport Layer gère les espaces d'adresse et de machine virtuelle, gère un canal (channel) entre les adresses d'objets, la connexion et le transport.

Nous tenons à signaler que c'est sur la technologie RMI qu'on a basé le développement de notre agent pédagogique qu'on va détailler après.

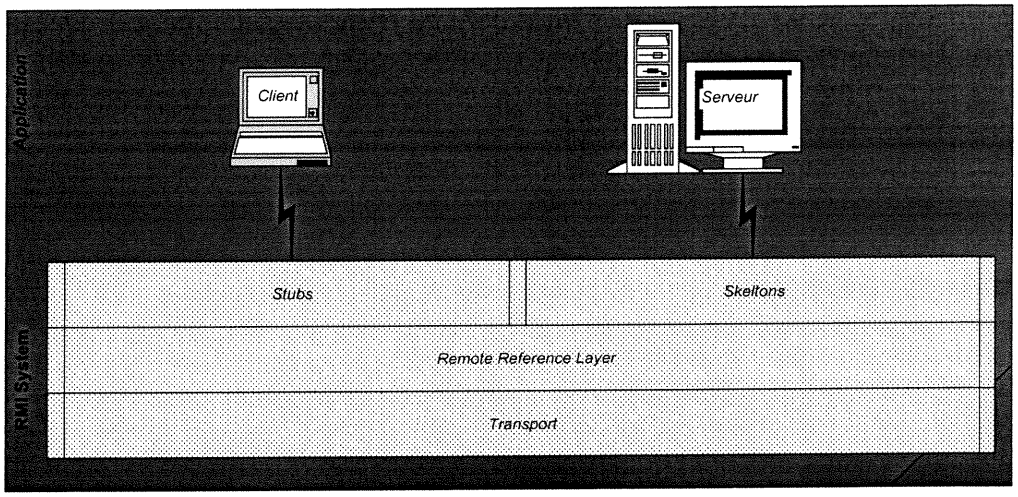


Figure 18 : Architecture RMI

5.5.3. Scénario d'utilisation.

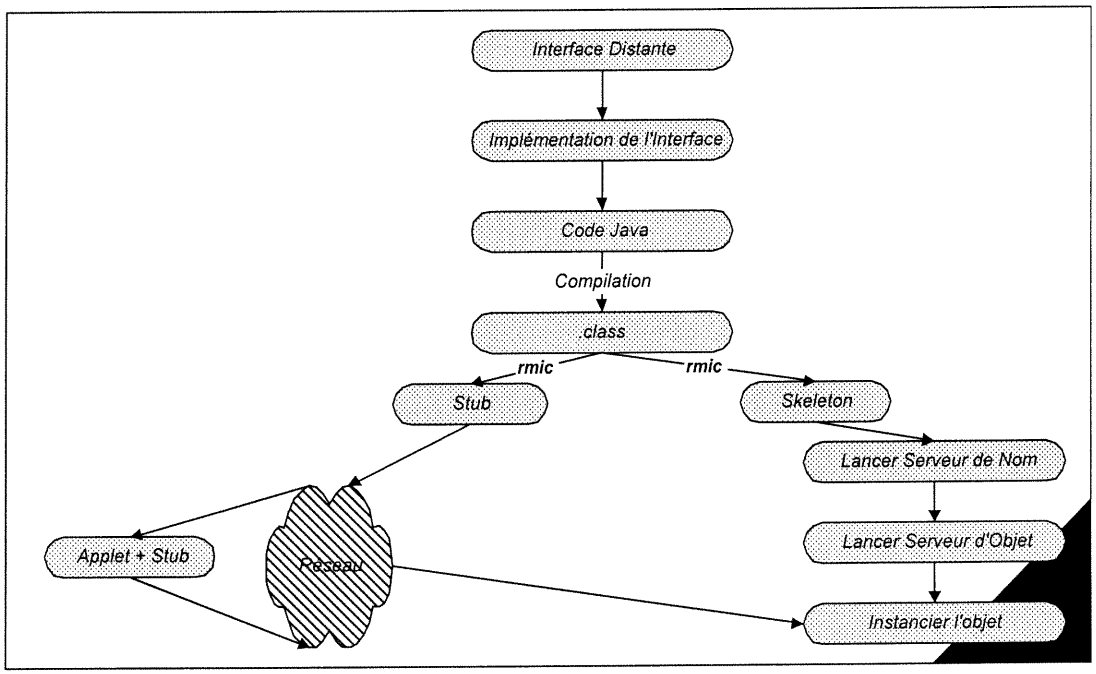


Figure 19 : Utilisation RMI

5.6 Environnement du tuteur

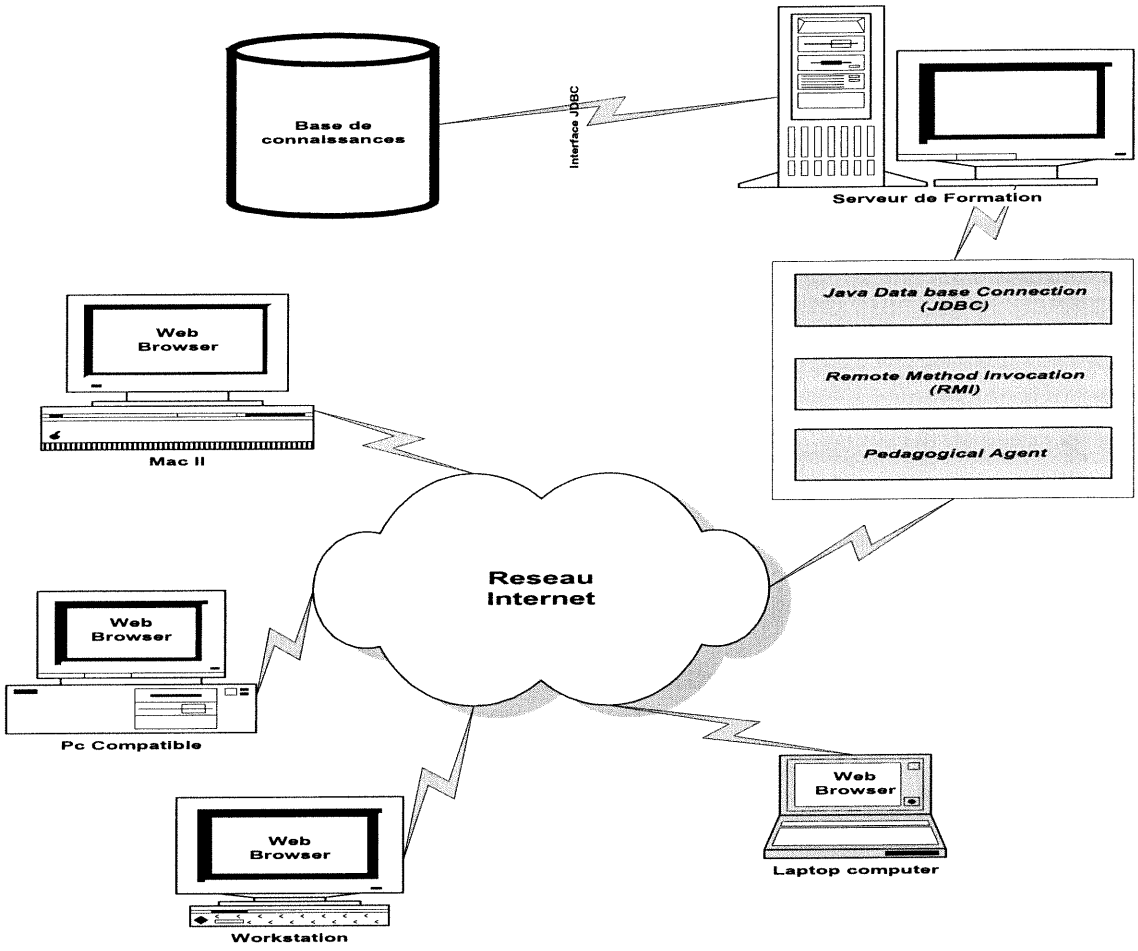


Figure 20 : Environnement opérationnel du tuteur

Exclusivement orienté WEB, l'environnement du tuteur n'a besoin que d'un browser (Netscape Communicator, Internet Explorer ou autres) pour se dérouler correctement sur le poste client. Le système vérifie automatiquement si tous les plug-in nécessaires pour supporter les composantes SWING de Java. Au cas contraire, le système les installe d'une manière transparente à l'apprenant. Notre choix des composantes SWING était justifié par le souci de donner une plus grande convivialité à l'interface utilisateur.

La première étape consiste en un processus d'identification de l'apprenant (voir Figure 21), cela permettra à l'agent pédagogique de savoir s'il s'agit d'un nouveau ou d'un ancien étudiant.

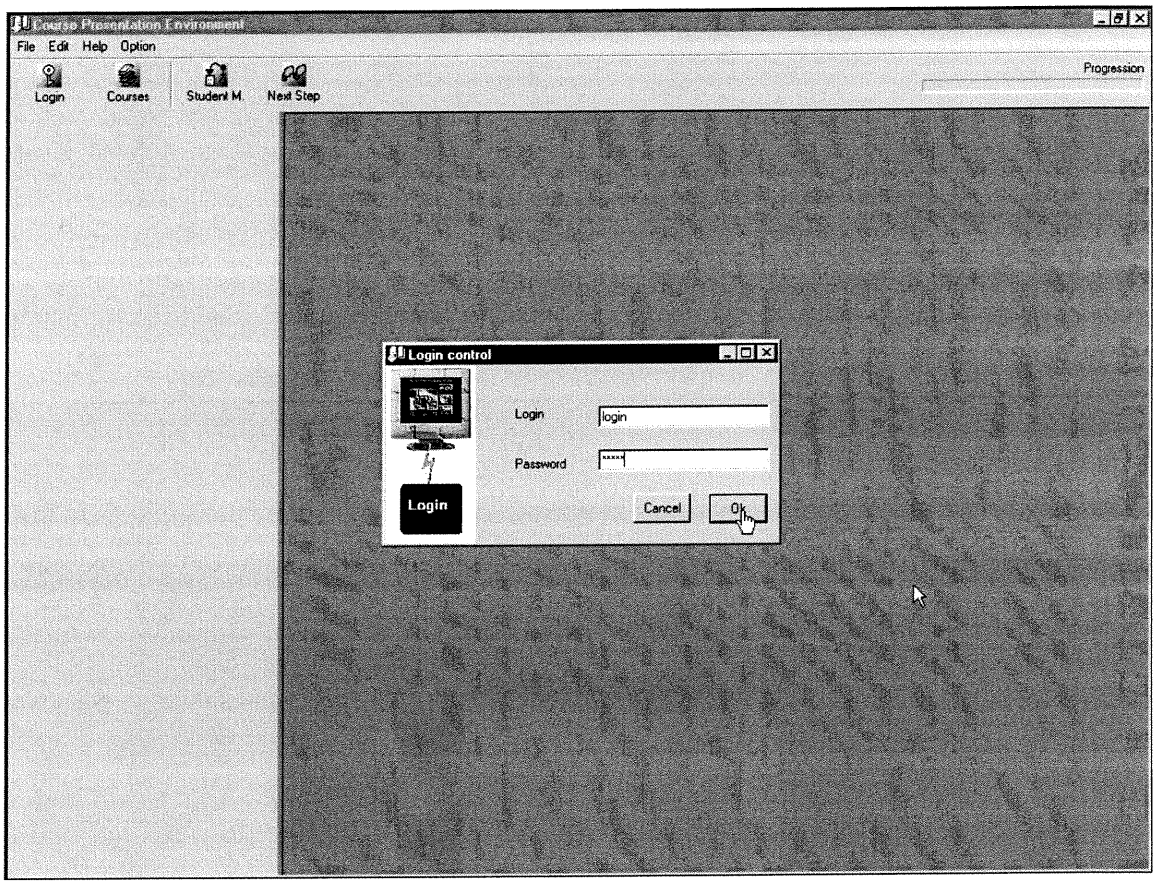


Figure 21 : interface avec serveur de formation

Dans les deux cas, le tuteur donne l'accès aux différents sujets disponibles sur le serveur de formation (voir Figure 22).

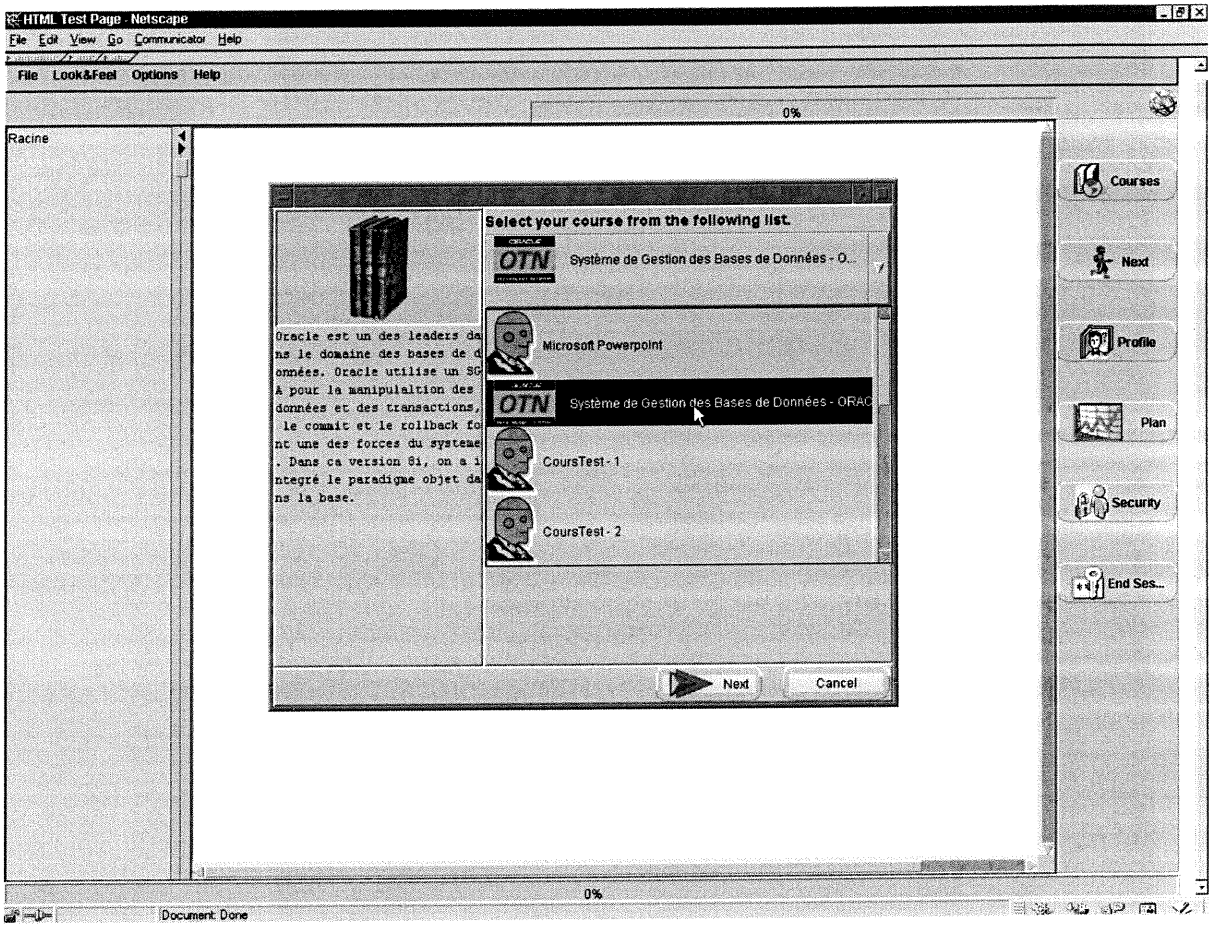


Figure 22 : Accés aux ressources de formation

5.7 Profil de l'apprenant

Le module de la gestion du modèle de l'apprenant est une partie intégrante de l'agent pédagogique. Ce dernier tient à jour le profil sur le serveur de formation après chaque session de formation. On distingue deux cas de figures :

Le premier cas concerne le nouvel étudiant qui vient de s'inscrire dans notre système. Après le chargement du contenu de la formation, l'agent pédagogique lui présente le curriculum de toutes les unités cognitives pour qu'il déclare son niveau de connaissance par rapport à chaque élément constituant le contenu de la formation préparé pour cette session(voir Figure 23).

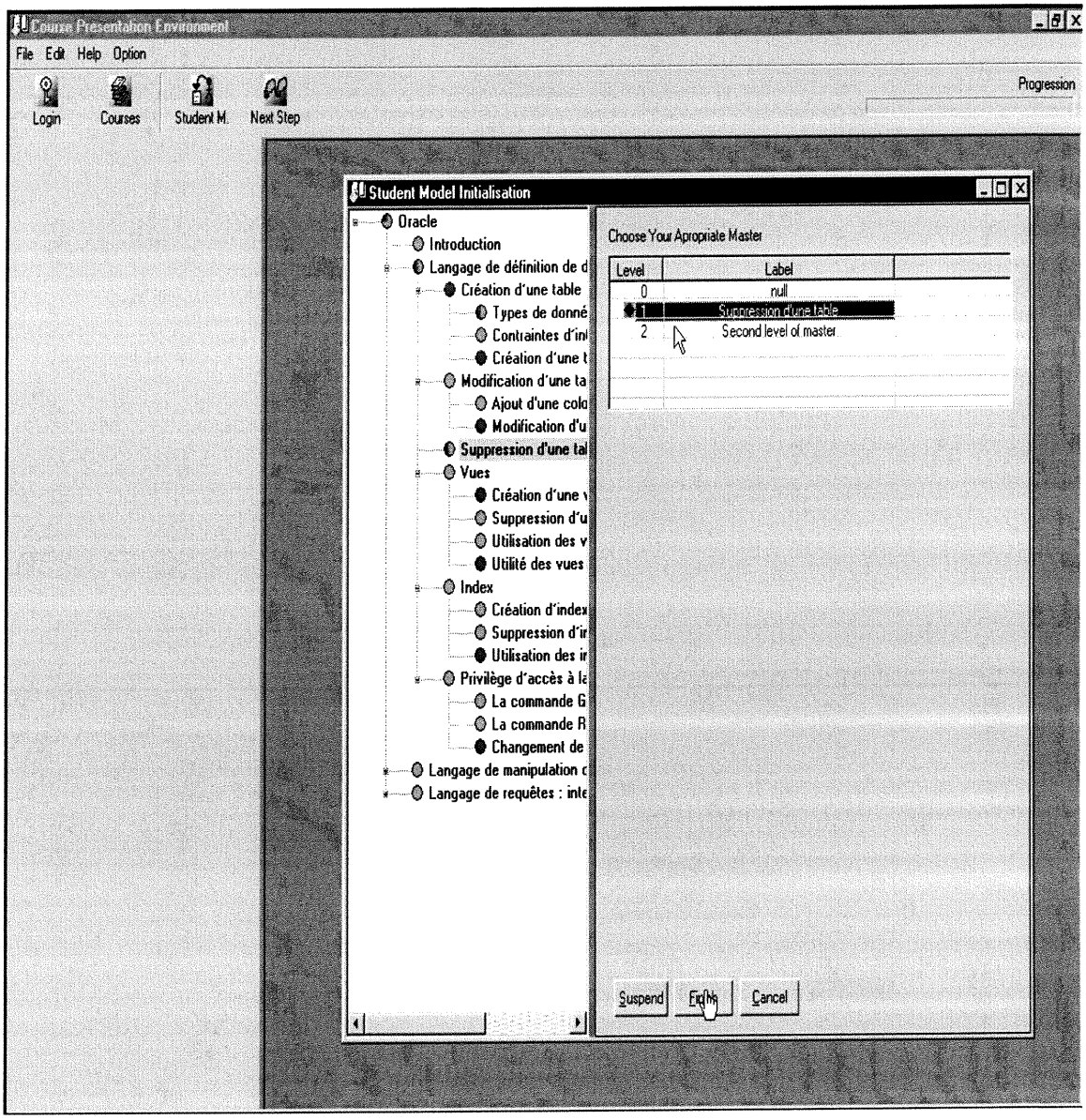
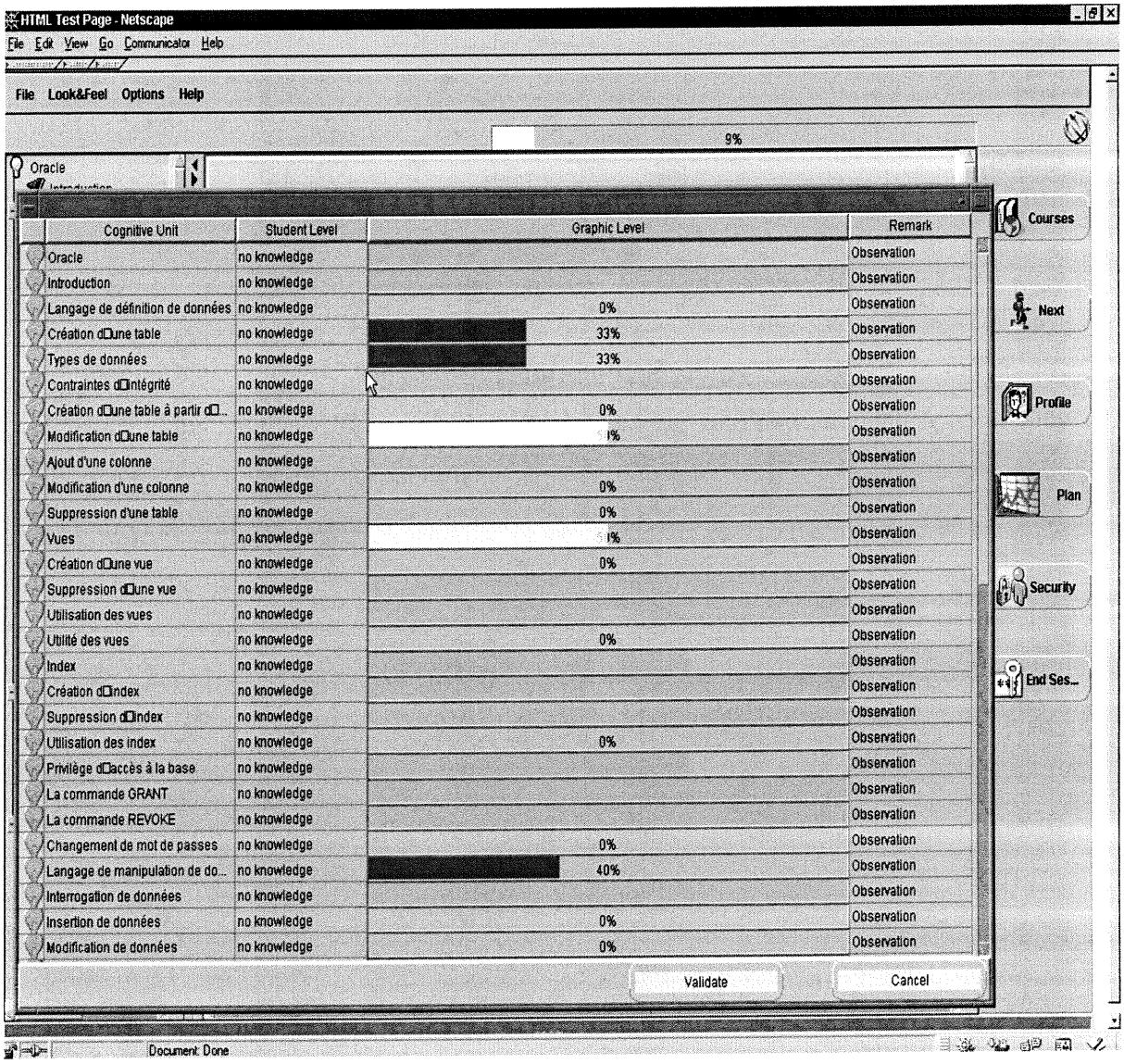


Figure 23 : Initialisation du Profil Apprenant

Le deuxième cas concerne les étudiants qui ont déjà utilisé le système au moins une seule fois, dans ce cas, l'agent pédagogique charge automatiquement le profil de l'apprenant en question. L'apprenant ne peut acceder cette fois ci qu'en mode consultation au contenu de son modele (voir figure 24).



Cognitive Unit	Student Level	Graphic Level	Remark
Oracle	no knowledge		Observation
Introduction	no knowledge		Observation
Langage de définition de données	no knowledge	0%	Observation
Création dUne table	no knowledge	33%	Observation
Types de données	no knowledge	33%	Observation
Contraintes dIntégrité	no knowledge		Observation
Création dUne table à partir dU...	no knowledge	0%	Observation
Modification dUne table	no knowledge	1%	Observation
Ajout d'une colonne	no knowledge		Observation
Modification d'une colonne	no knowledge	0%	Observation
Suppression d'une table	no knowledge	0%	Observation
Vues	no knowledge	1%	Observation
Création dUne vue	no knowledge	0%	Observation
Suppression dUne vue	no knowledge		Observation
Utilisation des vues	no knowledge		Observation
Utilité des vues	no knowledge	0%	Observation
Index	no knowledge		Observation
Création dIndex	no knowledge		Observation
Suppression dIndex	no knowledge		Observation
Utilisation des index	no knowledge	0%	Observation
Privilège dAccès à la base	no knowledge		Observation
La commande GRANT	no knowledge		Observation
La commande REVOKE	no knowledge		Observation
Changement de mot de passes	no knowledge	0%	Observation
Langage de manipulation de do...	no knowledge	40%	Observation
Interrogation de données	no knowledge		Observation
Insertion de données	no knowledge	0%	Observation
Modification de données	no knowledge	0%	Observation

Figure 24 : Profil de l'apprenant

5.8 L'agent pédagogique (AP)

Basé sur la technologie RMI (Remote Method Invocation), l'agent pédagogique est doté d'une architecture (voir Figure 27) lui permettant de gérer d'une manière efficace la session d'apprentissage de l'apprenant. Il contrôle entre autres, la progression de l'apprenant dans le cours en question, la gestion du profil de l'apprenant (mise à jour en temps réel du profil sur le serveur), la planification des activités à présenter en tenant compte du modèle de l'apprenant.

5.8.1 La structure de donnée utilisée

La partie donnée de l'AP est centrée principalement sur la classe java ''**Activity**'' schématisé comme suit :

La constante entière **présentation_et_évaluation** = 0;

La constante entière **présentation** = 1;

La constante entière **évaluation** = 2;

La constante entière **exercice** = 3;

La constante entière **planification_des_exercices_d'un_niveau_donné** = 4;

La constante entière **presenter_un_exercice** = 5;

Un Objet de type Unité Cognitive ;

Un entier « codeActiv » qui représente le code de l'activité;

Un Objet «ObjectOfActivity » qui peut être un contenu, un exercice,...

Pour gérer bien la fonction de planification, on a procédé à une codification de toutes les activités possibles. Ainsi, on distingue six cas de figures :

1- **présentation_et_évaluation = 0** : c'est l'état initial affecté au départ (durant le processus de chargement de cours) à chaque activité. On considère que l'activité comporte deux volets a savoir présentation du contenu et évaluation de l'apprenant.

2- **présentation = 1** : dans ce cas, l'agent sait qu'il s'agit d'une présentation du contenu de l'unité cognitive de l'activité courante. Il envoie une requête à l'environnement du tuteur qui demande le chargement de la ressource liée à l'unité cognitive en question, c'est une requête envoyée au serveur **HTTP**.

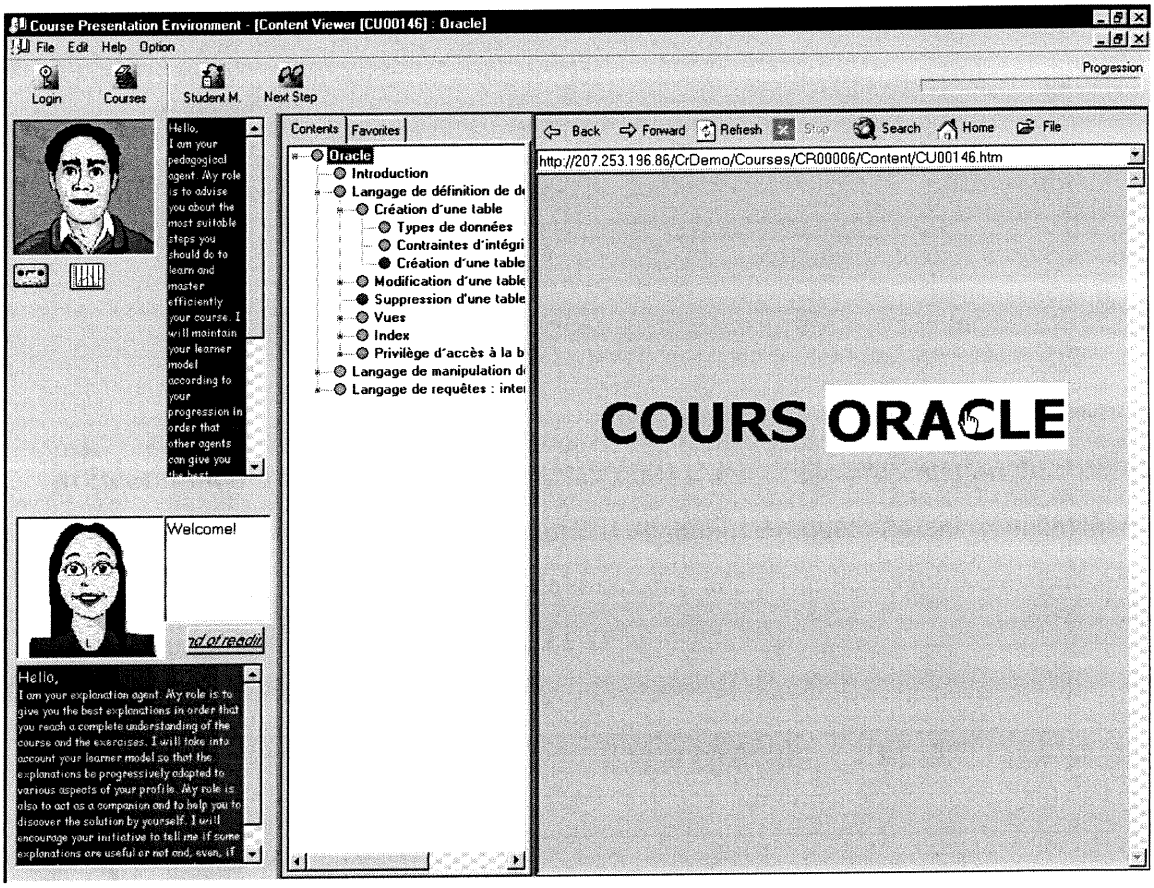


Figure 25 : Présentation du contenu

3- **évaluation = 2** : dans ce cas, il s'agit de procéder à l'évaluation de l'apprenant par rapport à l'unité cognitive en cours. Ce processus consiste à planifier un certain nombre d'activités de type exercices. Chaque activité planifiée aura **exercice** (valeur = 3) comme code. On planifie également, les activités liées aux sous unités cognitives de celle en cours avec **présentation_et_Evaluation** (valeur = 0) comme code.

4- **exercice = 3** : c'est le cas ou l'agent pédagogique décide de présenter les exercices attachés aux différents niveaux de maîtrises de l'unité cognitive en cours. L'Agent

planifie toutes les activités liées aux différents niveaux de maîtrise non encore acquis pour l'apprenant en question. On planifie, pour chaque niveau, une activité avec le code **planification_des_exercices_d'un_niveau_donné** (valeur = 4) et avec un objet de type niveau de maîtrise.

5- **planification_des_exercices_d'un_niveau_donné = 4:** ce code d'activité déclenche l'action de planification de toutes les exercices liés à chaque niveau de maîtrise. L'agent pédagogique planifie, pour chaque exercice, une activité de code **presenter_un_exercice** (valeur = 5). L'objet de l'activité dans ce cas est un exercice.

6- **presenter_un_exercice = 5 :** c'est à ce stade la ou l'agent pédagogique présente l'exercice dans l'environnement de formation.

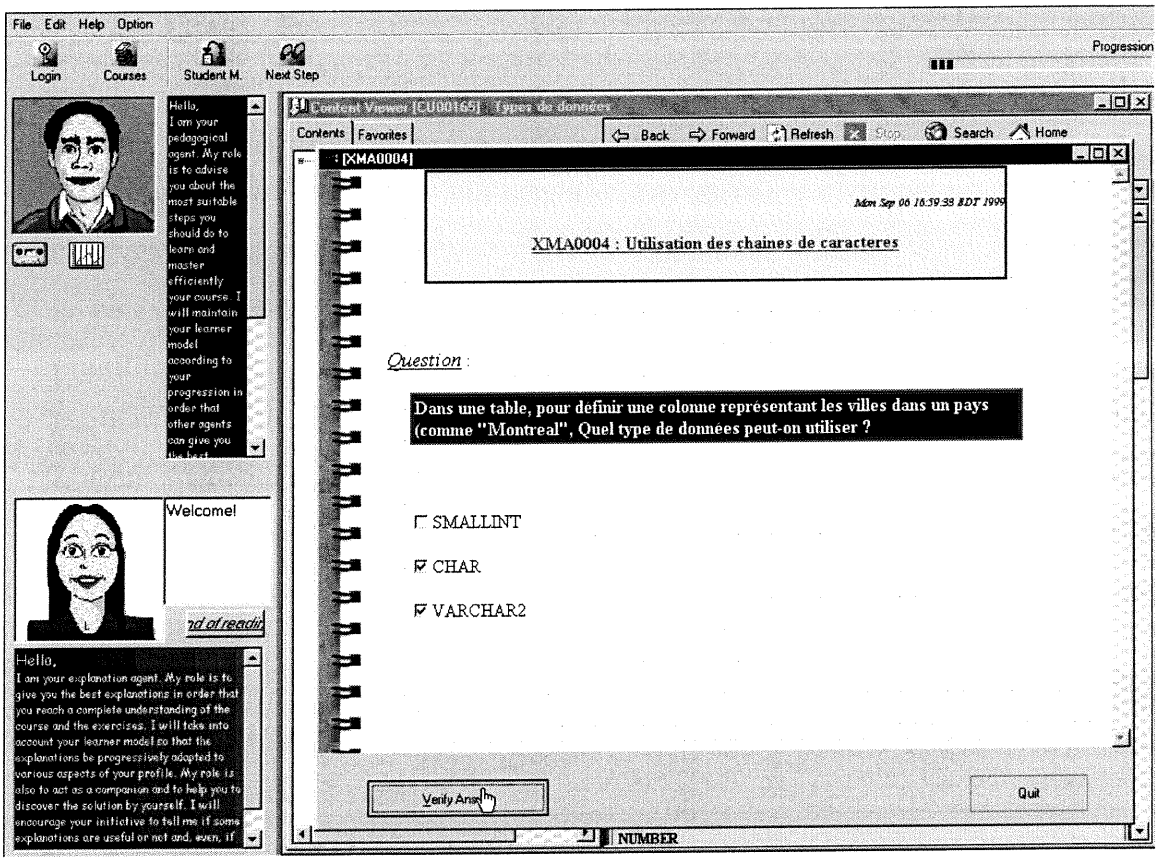


Figure 26 : Présentation d'une activité de type exercice

5.8.2 La logique de l'agent pédagogique

En plus de la partie des données, l'agent pédagogique est basé sur une logique lui permettant de gérer la session de formation. Le suivant constitue les principales fonctions implémentées par l'agent :

- **Fonction d'initialisation:**

Cette fonction permet à l'agent pédagogique d'initialiser son environnement par les informations de l'apprenant utilisant le système, du cours cible de la formation à distance, de la structure des unités cognitives constituant le cours en question et du profil de l'apprenant. Elle est exécutée à chaque connexion d'un nouvel utilisateur au tuteur.

- **Fonction de planification :**

C'est par le biais de cette fonction que l'agent pédagogique tente de déterminer quelle est la prochaine activité à présenter à l'apprenant. En tenant compte du profil de l'apprenant, l'agent pédagogique envisage les trois cas de figures suivants :

- présentation de contenu

  C'est le cas ou l'objet de l'activité à présenter est une ressource didactique représentant le contenu de l'unité cognitive en question. La ressource didactique est chargée par une requête formulée au niveau d'un serveur HTTP (serveur Apache dans notre cas).

- présentation de concepts

  A ce niveau, l'agent pédagogique présente la liste des concepts définie au niveau l'unité cognitive courante. Chaque concept met l'accent sur un aspect jugé très important par le concepteur de cours.

- Presentation d'exercices

  Dans ce processus d'évaluation, l'agent pédagogique présente une ressource de type exercice. La réussite de l'exercice permet à l'apprenant de changer son niveau de maîtrise par rapport à l'unité cognitive en question, en passant du niveau (n) au niveau (n+1).

Pour des questions d'optimisation et de performances, les ressources de type concepts, explications et exercices sont chargés juste à la demande.

- **Fonction d'évaluation :**

Cette fonction est appelée par l'agent pédagogique si l'apprenant achève une activité de type exercice, on passe l'objet exercice, le mode de terminaison de l'exercice et le score obtenu par l'apprenant. Les modes de terminaison d'un exercice sont codifiés comme suit :

* réalisation = 0 : c'est la cas d'une terminaison normale.

* suspension = 1 : c'est le cas ou l'apprenant abandonne l'exercice

Soit ''SeuilEchec'' le seuil d'échec d'un exercice fixé au départ par le concepteur;

Soit ''SeuilReussite'' le seuil de réussite de l'exercice en cours fixé au départ par le concepteur;

Soit **N** le niveau de maîtrise sollicité par l'apprenant.

DEBUT

en cas de suspension de l'exercice

    mettre a jour le profil de l'apprenant ;

    planifier l'activité suspendue;

    présenter encore une fois le contenu de la ressource didactique liée à l'unité cognitive courante ;

}

en cas de réalisation de l'exercice

    // c'est le cas d'une terminaison normale, auquel cas on examine le score obtenu

    Si (Score ≥ SeuilReussite) alors {

```
            // on change le niveau de l'apprenant au niveau supérieur
            modifier le profil de l'apprenant ;
            enlever l'activité courante de la pile des activités planifiées ;
            planifier la prochaine activité ;
        }
        Sinon {
            Si (Score ≥ SeuilEchec ) alors {
                // le score obtenu n'est pas tellement convaincant
                // On présente un exercice du même niveau de maîtrise
                Si ( existe un exercice du même niveau de maîtrise ) alors {
                    planifier la prochaine activité ;
                }
                Sinon  traiter le niveau précédent ;
            }
            Sinon {
                // c'est le cas ou Score ≤ SeuilEchec, c'est un échec total
                Si (N ≥ 2) alors {
                    rétrograder le niveau de l'apprenant ;
                    mettre à jour le profil ;
                    présenter le contenu de l'unité cognitive en question ;
                    planifier les exercices du niveau N-1 ;
                }
                Sinon {
                    planifier les unités cognitives pré-requis de celle en
            question ;
                }
            }
            planifier la prochaine activité ;
        }

FIN
```

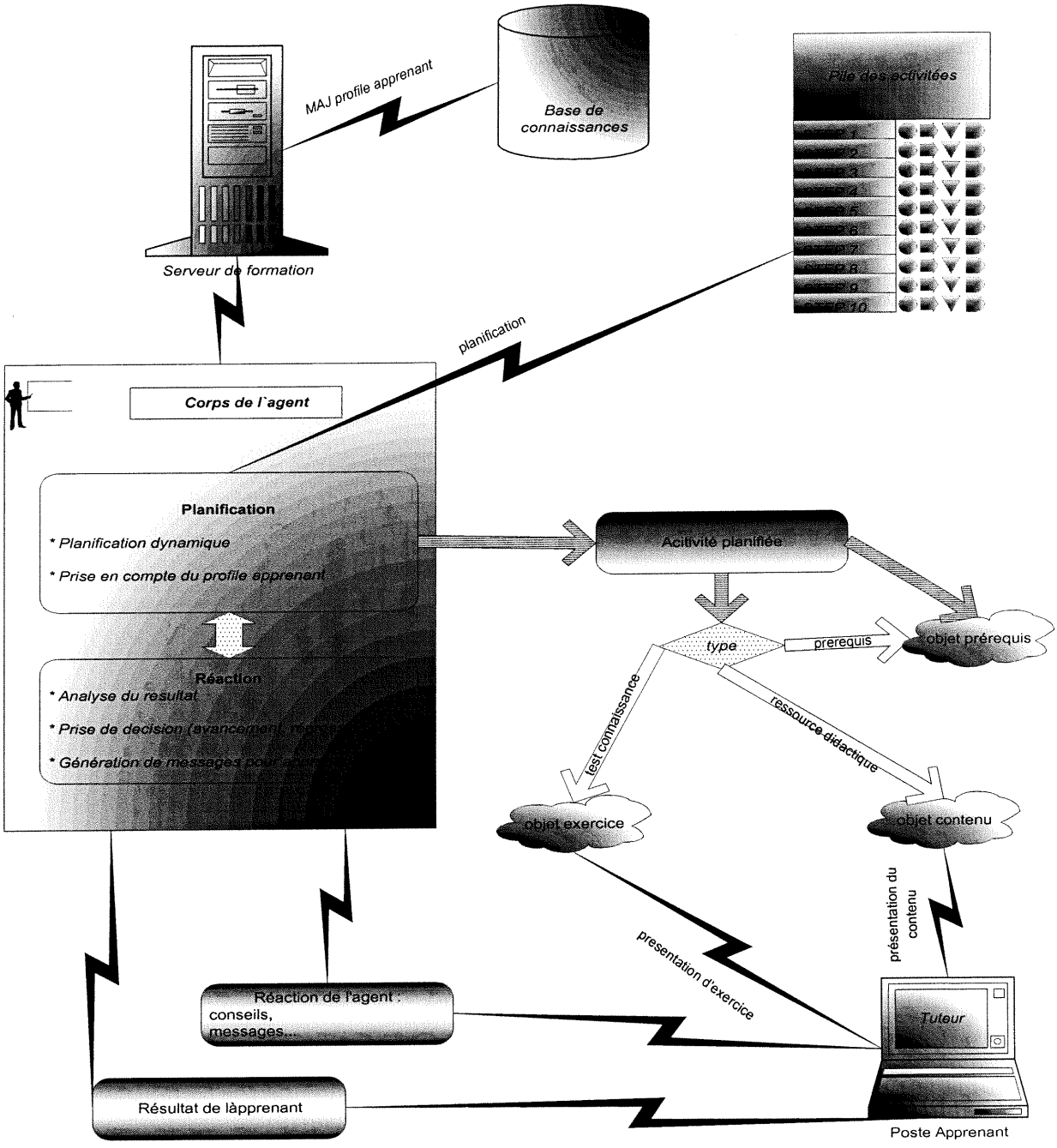


Figure 27 : Architecture de l'agent pédagogique

5.9 Expérimentation

Notre environnement de développement de cours a été expérimenté dans trois cours qui ont été développés dans la compagnie Virtuel Age Inc. par différentes personnes; le premier était relatif au système de gestion de bases de données (ORACLE), le deuxième traitait de Microsoft PowerPoint et le troisième concernait un logiciel de la réalité virtuele (EON). Dans cette entreprise, la contribution de nos outils a été de deux types : une contribution directe et une contribution indirecte.

Dans la contribution directe, nos outils ont été directement utilisés par les concepteurs de ces cours. Les outils de modélisation de la connaissance et les outils de génération automatique de cours ont permis aux concepteurs de représenter la connaissance du domaine et de générer des cours pour différents publics cibles. Des ressources pédagogiques sous format HTML ont été créées par des outils externes (Microsoft FrontPage et Netscape Composer) et associées aux différentes unités cognitives dans notre environnement.

Les concepteurs des cours ont trouvé nos outils conviviaux et faciles à utiliser. Les interfaces graphiques leur ont été très utiles pour surmonter la complexité inhérente à la modélisation de ce type. La gestion par vues de la base universelle du domaine facilitait le travail en groupe des membres des équipes de conception (deux concepteurs dans le cours système de gestion de bases de données (ORACLE), un dans celui sur Microsoft PowerPoint et deux sur celui de la réalité virtuelle) et permettait à plusieurs concepteurs de travailler simultanément sur le même cours.

Dans la contribution indirecte, notre système été un élément indispensable au fonctionnement d'autres modules dans d'autres projets de maîtrise (comme l'agent explicatif, développement des ressources de type exercices dans les STI). Un prototype intégré de ces modules a été donc mis au point et a permis de simuler quelques cours et de donner une formation d'essai pour une classe de 20 étudiants portant sur ORACLE comme système de gestion de bases de données.

Nous signalons que la facilité, l'optimisation et la productivité des experts de la matière comme critères sont au centre de nos critères d'expérimentation et d'évaluation de nos recherche.

CHAPITRE IV

CONCLUSION

Le but de notre projet est la réalisation d'un environnement intégré de modélisation et de restructuration de cours d'une part et de rendre le système fonctionnel dans un contexte distribué orienté WEB. Contrairement au STI traditionnelle (STI opérant intégralement sur un poste client), notre STI profite d'une large exploitation, tirant ainsi profil de la prolifération des nouvelles technologies (Java, RMI,...), pour atteindre une large population géographiquement répartie.

Le module de modélisation de cours (module du curriculum) est au centre de l'architecture de ce projet. Il est chargé de fournir les expertises pédagogiques du domaine aux autres composants du STI. La modélisation et la représentation de connaissances dans le module du curriculum sont inspirées du modèle CREAM. Nous avons apporté des simplifications pour rendre l'utilisation de notre système plus facile dans un milieu pratique.

L'intérêt de notre modélisation vient du fait qu'elle possède plusieurs avantages :
- Prise en compte de l'expertise du domaine : exprimée dans le réseau d'unités cognitives en terme de contenu et de niveaux de maîtrises.
- Prise en compte de l'expertise pédagogique : exprimée dans la logique de l'agent pédagogique.

- Flexibilité de la représentation due à l'utilisation de structures essentiellement déclaratives faciles à gérer et à manipuler.

- Support des processus de génération automatique de cours avec prise en compte du modèle cognitif de l'apprenant et des besoins de formation.

Nous avons développé nos outils de modélisation de cours dans le but d'améliorer l'organisation globale de la connaissance et du processus de modélisation. Le principe de la base universelle de connaissances que nous avons introduit a permis de remédier aux lacunes de la majorité des STI qui ne considéraient la conception que dans un cadre mono-concepteur et mono-curriculum.
Ce principe a procuré plusieurs avantages à notre environnement :
- élimination de la redondance,
- diminution des coûts de réalisation,
- augmentation de la productivité des concepteurs,
- augmentation de la qualité générale des cours développés.
Il n'y a pas de mesures réelles mais constatation par rapport à des approches complexes comme celles de CREAM ou de Merrill par exemple.
Pour pouvoir rendre notre système opérationnel dans un contexte WEB, nous avons développé le module tuteur en se basant sur une architecture distribuée. Le module tuteur présente à l'apprenant toute la partie interactive dans le cadre d'une interface usager, plus souple et plus conviviale, basée sur les composants SWING de Java. La partie logique de l'agent pédagogique réside dans le serveur. Ce n'est qu'au moment de l'ouverture d'une nouvelle session de formation que l'agent pédagogique se présente à l'apprenant avec son interface pour gérer la progression de l'étudiant dans le cours.

Nous avons développé également le module de gestion du profil de l'apprenant qui réside sur le serveur de formation. On a introduit des algorithmes pour rendre l'exploitation de ce profil (en consultation et en mise à jour) plus rapide et plus efficace.

L'expérimentation de notre environnement de développement de cours et de d'exploitation des cours développés dans le cadre de formation à distance suggère plusieurs extensions futures. Bien que nos outils soient très faciles à utiliser, les concepteurs éprouvent parfois des difficultés, comme dans le choix des notions de modélisation à utiliser et du niveau de détail à avoir dans la conception. L'intégration d'un conseiller à l'environnement de développement de cours, peut apporter une aide importante aux concepteurs. Le conseiller suivra le travail du concepteur à la trace et lui fournira des explications et des exemples de cas pouvant l'aider dans sa tâche, soit à la demande du concepteur ou quand le conseiller le jugera nécessaire. Ceci, peut être réalisé par un mécanisme d'explication de connaissances que nous pouvons introduire.

Une autre direction très intéressante à explorer concerne l'intégration, dans le tuteur, d'un outil efficace pour supporter le travail collaboratif entre les différents apprenants, en particulier dans les activités de test (résolutions de problèmes, déroulement d'une simulation,...).

BIBLIOGRAPHIE

[Aïmeur 97]
Aïmeur, E., Frasson, C., Dufort, H., Leibu, D. "Some Justifications For The Learning by Disturbing Strategy", AI-ED 97, World Conference on Artificial Intelligence and Education, Japan, pp. 119-126, Août, 1997.

[Ayala 95]
Ayala, G. & Yano, Y. "GRACILE: A Framework for Collaborative Intelligent Learning Environments", Journal of the Japanese Society of Artificial Intelligence, Vol.10 No.6, pp. 156-170, 1995.

[Alpert, Singley et Peterf, 2000]
Alpert,R, Singley,K, Peter G. Fairweather : "Porting a standalone intelligent tutoring system to the Web", 5th International Conference, ITS 2000 Montréal, Canada, June 2000.

[Anderson 88]
Anderson, J.R. "The expert module", Polson & Richardson, pp. 21-53, 1988.

[Burns et Capps,1988]
Burns, H. L, Capps, C,G. Foundations of intelligent tutoring system : an introduction. Dans Polson, M.C. and Richardson, J.J (Eds) Foundation of Intelligent Tutoring Systems, pp. 21-53, Lawrence Erlbaum Associates Publishers, 1988.

[Brown et al., 1982]
Brown,J.S., Burton, R.R, Kleer, J.. Pedagogical, naturel language and knowledge engineering in sophie I, II and III in D. Sleeman and J.S Brown(Eds), intelligent tutoring systems. pp. 227-282, academic press 1982.

[Brown et Burton, 1978]
Brown, J.S., Burton, R.R. Diagnostic models for procedural bugs in basic mathematical skill. Cognitive science, vol. 2, pp. 379-426. 1978.

[Bloom, 1984]
The 2 sigma problem : The search for methods of group instruction as effective as one-to-one tutoring, Educational Researcher, 13, 3-16. 1984.

[Brown et VanLehn, 1980]
Brown, J.S, VanLehn, K.. Repair theory: a generate theory of bugs in procedural skills. Cognitive Science, vol. 4, 379-426, 1980.

[Brown et al., 75]
Brown, J.S., Burton, R.R., Bell, A.G. "SOPHIE : a step towards a reactive learning environment", Man-Machine studies, 1975.

[Burton 82]
Burton, R.B., Brown, J.S. "An investigation of computer coaching for informal learning activities", Intelligent Tutoring Systems, pp 79-98. Academic Press, New York, NY, 1982.

[Burton et Brown, 1976]
Burton, R.B, Brown, J.S. Tutoring and Student Modeling paradigm for gaming environements. SIGCSE Bulletin, vol.8, 236-246,1976.

[Barr et Davidson, 1981]
Barr, A., Davidson, J. Representation of knowledge. Barr, A, and Feigenbaum, E.A (Eds), the handbook of Artificial Intelligence vol. I., 141-222, William Kaufmann. 1981.

[Cabrol et al., 91]
Cabrol, D., Rabine, J.P., Ricard, D., Rouillard, M. "EXP'AIR : un logiciel Pour l'apprentissage de l'interprétation des données de spectroscopique Infrarouge", Quéré, M., "Système experts et enseignement assisté par ordinateur", pp 121-148, Ophrys, 1991.

[Carbonell 70]
Carbonell, R. "AI in CAI : an artificial intelligence approach to computer aided instruction", IEEE transactions on man – machine systems, vol.11, pp 190-202. 1970.

[Chan 90]
Chan, T.W., Baskin, A.B. "Learning companion systems", dans Frasson, C., Gauthier, G. (dir.), Intelligent tutoring systems : at the crossroad of AI and education. Ablex Publishing Corp., Norwood, NJ, 1990.

[Clancey 82]
Clancey, W. J. "GUIDON". Dans Barr & Feigenbaum (dir), The handbook of artificial intelligence, pp 267-278. William Kaufmann, Inc., Los Alto, CA, 1982.

[Carr et Goldstein 1977]
Carr, B., Goldstein, I.P. Overlay:A theory of modeling for computer aided instruction. In Artificial Intelligence Meno 406 (Logo Memo 40), Massachusetts Institute of Technology, 27-33. 1977.

[Carbonell, 1970b]
Carbonel, J.R. Mixed-initiative Man-Computer instructional Dialogues. Doctorat Dissertation. Massachisetts Institute of Technology. 1970.

[Clancey 1979]
Clancey, W. J. Tutoring rules for guiding a case method dialogue. International Journal of Man-Machine Studies, vol. 11, 24-49. 1979.

[Fisher 88]
Fisher, G., Morch, A. "CRACK : a critiquing approach to cooperative kitchen design", dans Frasson, C. & Gauthier, G. (dir.), Proceedings of first international conference on intelligent tutoring systems, pp 176-185, Montréal, QC, 1988.

[Frasson et La Passadiere 1990]
Frasson, C., La Passadiere, B. A student model based on learning context. In Proceedings of the International Conference on Advanced Research on Computer in Education, IFIP, Tokyo, Japan 231-336. 1990.

[Frasson et al., 97]
Frasson, C., Aïmeur, E., Mengelle, T., Leibu, D. "Processus d'instruction d'acteurs pédagogiques", Colloque Cinquièmes journées EIAO de Cachan, pp 251-262, Hermes, France, Mai 1997.

[Ferraris 84]
Ferraris, M., Modoro, V., Olimpo, G. "Petri Nets as a modeling tools in the development of CAL courseware", Computer in education, vol. 8, no. 1, pp 286-290, 1984.

[Gagné 92]
Gagné, R. M. Principles of instructional design. Harcourt Brase Jovanovich College, Publisher. Fort Worth, Texas,1992.

[Gagné 1976]
Gagné, R. M. Les principes fondamentaux de l'apprentissage : application à l'enseignement. Les editions HRW Ltée, Montreal, 4e edition. 1976.

[Goldstein 1982]
Goldstein, L. P. The generc graph : a representation for the evolution of procedural knowledge. In Sleeman, D.H and Brwon, J.S (Eds), Intelligent Tutoring Systems, 51-78, Academic press, London.1982.

[Guo 98]
Guo, R. "Visual manipulation for development of curriculum", NTICF, Rouen, 1998.

[Holt et al, 1994]
Holt, P., Dubs, S., Jones, M., Greer, J.. The State of Student Modelling. Dans Greer, J.E and McCalla, G.I (Eds) : Student Modelling : the key to individualized knowledge-based instruction, 1994.

[Halff 88]
Halff, H. "Curriculum and instruction in ITSs", Foundations of intelligent tutoring systems, pp 19-108, Lawrence Erlbaum Associates, Hillsdale, NJ, 1988.

[Lajoie 89]
Lajoie, S., Lesgold, A. & al. "A procedural guide to the avionics Troubleshooting tutor development process", Rapport technique, Learning research and development center, University of Pittsburgh, Pittsburgh, 1989.

[Lantz et al., 1983]
Lantz, B. S., Bergar, W. S., Farley, A. M. An intelligent CAI system for teaching equation solving. Journal of Computer-Based Instruction, vol. 10, 32-42, 1983.

[Nkambou 96]
Nkambou, R. "Modélisation des connaissances de la matière dans un système tutoriel intelligent : modèles, outils et applications", Thèse de Ph.D., Département d'informatique et de R.O., Université de Montréal, QC, 1996.

[Reiser 85]
Reiser, B.J., Anderson, J.R., Farrell, R.B. "Dynamic student modeling in an intelligent tutor for LISP programming ", Proceedings of the International Joint Conference on Artificial Intelligence-85, page 8-14, Los Altos, CA : Morgan Kaufmann, 1985.

[Reiser et al., 1985]
Reiser, B.J,Anderson, A, Farrel, R. Dynamic student modeling in an intelligent tutor for lisp programming. In Proceeding of the Ninth International Joitn Conference of Artificial Intelligence, Los Angeles, 8-1, 1985.

[Ragnemalm 1996]
Ragnemalm, E.L. Student diagnosis in practice--; bridging a gap. User Modeling and User-Adapted Interaction vol. 5, no. 2, 93-116, 1996.