Utilisation de l'abstraction et de la complexité dans la conception de systèmes éducatifs intelligents

Ruddy Lelouche

Utilisation de l'abstraction et de la complexité dans la conception de systèmes éducatifs intelligents

Ruddy Lelouche

Département d'informatique, UNIVERSITÉ LAVAL, Québec G1K 7P4 CANADA

Tel.: (418) 656-2131, poste 2597 Fax: (418) 656-2324
e-mail: lelouche@ift.ulaval.ca

Résumé

Dans les domaines faisant intervenir de la résolution de problèmes, tels que les disciplines du génie et la plupart des disciplines relevant des «sciences exactes», la connaissance que doit acquérir l'étudiant est double: la connaissance du domaine lui-même, mais aussi et surtout la connaissance nécessaire pour résoudre des problèmes dans ce domaine. Il s'ensuit qu'un système éducatif dans un tel domaine doit comprendre trois types de connaissances: la connaissance du domaine et la connaissance de résolution de problèmes, qui constituent les connaissances que doit acquérir et maîtriser l'étudiant, et la connaissance tutorielle, utilisée par le système pour faciliter le processus d'apprentissage de l'étudiant. Cet article a deux buts: d'une part présenter et modéliser ces trois types de connaissances, d'autre part, pour chacun de ces types, montrer comment apparaissent les niveaux d'abstraction et de complexité et comment il est possible de les prendre en compte. Nous montrons ainsi comment ces différents niveaux peuvent jeter un éclairage uniforme sur l'opération du système et le rendre plus convivial. Nous espérons ainsi apporter une modeste contribution au problème important et plus global de la définition, sinon d'une architecture générique, au moins d'une approche globale à la conception de systèmes éducatifs intelligents.

Mots-clés: niveau d'abstraction, de complexité, but tutoriel, conception de système, adaptation, raffinement, modélisation, résolution de problèmes, tutorat.

Abstract

In problem-solving domains, like engineering and most "exact science" disciplines, the knowledge to be acquired by the student is twofold: the domain knowledge itself, but also the knowledge used to solve problems in that domain. Thus an educational system in such a domain must encompass three types of knowledge: domain knowledge and problem-solving knowledge, i.e. the knowledge to be acquired and mastered by the student, and tutoring knowledge, used by the system to facilitate the student's learning process. This paper has two goals: first to present and to model these three knowledge types, and then to show how absttraction and complexity levels appear for each type and how they can be accounted for. We thus show how these various levels can shed a uniform light on the system operation and make it more user-friendly. We thus hope to bring a modest contribution to the important problem of defining at least a general approach to the design of intelligent educational systems.

Keywords: abstraction level, complexity level, tutoring goal, system design, adaptation, refinement, modelling, problem solving, tutoring.

Introduction

Presque tous les domaines enseignables varient en complexité, depuis les principes de base jusqu'à la résolution de problèmes ou la prise en compte de situations relativement complexes. Ainsi, un apprenant doit apprendre les bases d'un tel domaine avant de pouvoir s'attaquer à des notions plus larges ou plus approfondies. Et lorsqu'un tuteur humain détecte des erreurs ou des malentendus, il attire habituellement l'attention de l'étudiant sur un petit sous-ensemble des connaissances impliquées, de manière à corriger efficacement ces erreurs ou ces malentendus. Une telle interaction est centrée soit sur certains éléments de la connaissance du domaine, soit sur le niveau de la connaissance de résolution impliquée par un problème donné.

De plus, l'enseignement est un processus complexe en soi. Les activités et les stratégies d'enseignement varient considérablement selon leur complexité intrinsèque, le rôle et l'autonomie qu'elles laissent à l'apprenant, les types d'interactions qu'elles provoquent avec l'apprenant, les évaluations qu'elles permettent, les liens qu'elles font entre la théorie et la pratique, etc. De ce dernier point de vue, il est possible de classer les domaines «enseignables» en trois catégories selon le type de conaissance à acquérir par l'apprenant: le savoir-savoir, le savoir-faire, et le savoir-être. Des exemples de tels types de domaines sont respectivement: l'anatomie ou la grammaire d'un langage, l'aptitude à résoudre un problème mathématique ou médical, et la capacité de s'adapter à son environnement ou de traiter des relations interpersonnelles délicates. Nous nous intéressons plus particulièrement aux domaines du deuxième type.

Dans un domaine de résolution de problèmes ou de savoir-faire, que nous appellerons désormais *domaine-RP*, la connaissance à acquérir par l'étudiant est double:

d'une part la connaissance du domaine lui-même, mais aussi et surtout la connaissance nécessaire pour résoudre des problèmes dans ce domaine. Il s'ensuit qu'un logiciel éducatif dans un tel domaine, que nous appellerons similairement *logiciel-RP*, doit comprendre trois types de connaissances: la connaissance du domaine et la connaissance en résolution de problèmes, qui constituent les connaissances que doit acquérir et maîtriser l'étudiant, et la connaissance tutorielle, utilisée par le système pour faciliter le processus d'apprentissage de l'étudiant. Tout cela n'est certes pas nouveau.

Cet article a deux buts:

- présenter chacun des trois types de connaissance mis en œuvre dans un logiciel-RP;
- pour chaque type de connaissance, montrer comment apparaissent les niveaux d'abstraction et de complexité et comment il est possible de s'en servir.

Pour cela, nous présentons dans la section 1 notre modélisation de la connaissance du domaine et prenons quelques domaines comme exemples. La section 2 est consacrée aux avantages qu'il y a, dans un logiciel-RP, à séparer la connaissance du domaine de la connaissance en résolution de problèmes, et à la présentation de quelques activités de résolution de problèmes dans divers domaines-RP. Dans la section 3, nous décrivons brièvement quelques principes de modélisation de la connaissance tutorielle dans un logiciel-RP. Dans chacune de ces sections, nous montrons informellement comment apparaissent les niveaux d'abstraction et de complexité dans le type de connaissances décrites, et indiquons comment les utiliser dans les différents domaines-RP pris comme exemples. Enfin, la section 4 définit de manière plus formelle les niveaux d'abstarction et de complexité et, s'appuyant au besoin sur les sections précédentes, présente les avantages pédagogiques de l'utilisation de ces niveaux d'abstraction et de complexité dans la modélisation des trois types de connaissances mis en œuvre dans un logiciel-RP.

1 Connaissance du domaine

Pour décrire la connaissance du domaine, nous présentons d'abord ses caractéristiques dans un logiciel-RP général (section 1.1). Nous montrons ensuite comment on peut la modéliser dans quelques domaines-RP (section 1.2), et comment cette approche nous permet d'introduire les notions de niveaux d'abstraction et de complexité (section 1.3).

1.1 Généralités

La *connaissance du domaine* (DK pour *domain knowledge*) contient tous les aspects théoriques et factuels de la connaissance à enseigner à l'étudiant. C'est le premier type de connaissance mis en œuvre dans un logiciel éducatif intelligent. Bien que sa structure spécifique varie, cette connaissance peut typiquement inclure des concepts, des entités et des relations décrivant le domaine [Brodie & al., 1984], des classes et des instances d'objets [Kim & Lochovsky, 1989], éventuellement des conditions ou contraintes d'utilisations, des faits, des règles [Kowalski, 1979; Clocksin & Mellish, 1981], des réseaux sémantiques ou associatifs [Findler, 1979; Sowa, 1984], etc.

Les activités principales du système centrées sur ce type de connaissance consistent à:

- fournir des présentations et des explications théoriques sur les divers éléments de connaisance et leurs relations dans le domaine d'enseignement;
- fournir aux autres modules du système, c'est-à-dire ceux liés à la résolution de problèmes et au tutorat, le support de base en connaissance du domaine dont ils ont besoin.

1.2 Application à quelques domaines

Dans le domaine particulier du **génie économique**, Lelouche et Morin [1998; Morin, 1998] représentent ce type de connaissance par des concepts, des relations, et certaines relations modélisées comme des concepts, les facteurs.

Les *concepts* sont des entités telles que l'investissement, l'intérêt, la durée de l'investissement, les valeurs présente et future, la composition de l'intérêt, la période de composition, le taux d'intérêt, les annuités ou versements périodiques, etc.

Les concepts sont liés les uns aux autres par différents types de *relations*. Celles-ci sont soit les relations habituelles utilisées en représentation des connaissances, comme *sous-classe de, élément de, sorte de*, etc., soit des relations numériques représentées par des formules. Une telle formule est:

$$F = P \times (1 + i)^n$$

qui, à partir de la valeur présente P d'un investissement de n périodes au taux i, calcule la valeur future correspondante F de cet investissement.

Une formule telle que (1) peut se réécrire:

$$F = P \times \Phi_{PF,i,n} \text{ où } \Phi_{PF,i,n} = (1 + i)^n$$

ou encore $P = F \times \Phi_{FP,i,n}$ où $\Phi_{FP,i,n} = (1 + i)^{-n}$

ce qui introduit ainsi les *facteurs* $\Phi_{PF,i,n}$ et $\Phi_{FP,i,n}$. Les facteurs nous permettent de séparer leur définition (égalités de droite ci-dessus, aspect quantitatif) de leurs utilisations possibles dans le domaine d'application (égalités de gauche, aspect qualitatif).

De manière similaire, un autre facteur $\Phi_{AP,i,n}$ convertit une série de montants annuels identiques A en une seule valeur présente P:

$$P = A \times \Phi_{AP,i,n} \text{ avec } \Phi_{AP,i,n} = \frac{(1+i)^n - 1}{i\,(1+i)^n}$$

En fait, $\Phi_{AP,i,n}$ est une somme de facteurs Φ_{FP}. Le facteur $\Phi_{PA,i,n}$ effectue le processus inverse:

$$A = P \times \Phi_{PA,i,n} \text{ avec } \Phi_{PA,i,n} = \frac{i\,(1+i)^n}{(1+i)^n - 1}$$

Il existe encore d'autres facteurs, par exemple pour convertir une série arithmétique ou géométrique de montants périodiques en une valeur présente ou en une valeur future; ces facteurs se calculent également comme une somme de facteurs Φ_{FP} et/ou Φ_{PF}.

En **geometrie**, les *concepts* sont des éléments tels que le point, la ligne, le segment, et plus tard l'angle. Des exemples de *relations* entre concepts sont l'adjacence (de segments ou d'angles), le parallélisme (de lignes ou de segments), la complémentarité (d'angles), etc. Des concepts de plus haut niveau sont le triangle, le rectangle, le carré, le polygone (généralisation du nombre de côtés), le cercle, l'ellipse (passage à l'infini du nombre de côtés), le solide, le polyhèdre (passage à la troisième dimension), etc. Ces concepts plus abstraits ou plus complexes sont définis en utilisant les concepts de plus bas niveaux et les relations qui les lient (ainsi, un triangle est un ensemble de trois segments adjacents deux à deux).

En **physique mécanique**, on introduit de même les *concepts* de temps, de distance, de vitesse, d'accélération, de masse, de force, et plus tard de vitesse angulaire, d'accélération angulaire, de moment d'inertie, de couple, etc. On introduit aussi des *relations*, comme celle définissant la vitesse comme la variation de distance par unité de temps, ou celle spécifiant que l'accélération a est proportionnelle à la force appliquée F. En généralisant le mouvement linéaire au mouvement angulaire, une autre relation définit la vitesse angulaire comme la variation d'angle par unité de temps, et une autre spécifie que l'accélération angulaire α d'un corps solide est proportionnelle au couple τ qui lui est appliqué. Plus précisément, nous avons: pour un mouvement linéaire:

$$F = M \times a \quad \text{avec } M = \text{masse totale du corps} \qquad (6)$$

et pour un mouvement circulaire:

$$\tau = I \times \alpha \quad \text{avec } I = \sum (m \times r^2) \qquad (7)$$

L'équation (6) exprime la seconde loi de Newton. Dans l'équation (7), I est le moment d'inertie, exprimé en fonction de la masse m de chacune de ses particules et de leur distance r à l'axe de rotation. Il est clair que M dans l'équation (6) et I dans l'équation (7) jouent un rôle de facteurs comme en génie économique.

Les formules telles que (2–7) reliées aux facteurs portent essentiellement sur des aspects quantitatifs, mais les similarités et les différences entre eux, et les circonstances qui président à l'utilisation de l'un quelconque d'entre eux, sont de nature profondément qualitative. En **génie économique**, si la *valeur* d'un facteur se calcule effectivement à partir de deux ou trois paramètres numériques, le *contexte* dans lequel ils sont utilisés varie selon que l'on désire déplacer dans le temps un montant unique ou une série de montants, identiques ou non, ou au contraire calculer un montant annuel équivalent, etc. En fait, ce contexte correspond au type de conditions qui gouvernent l'investissement, ou *type de conditions de l'investissement*, indépendamment des montants et des durées impliqués, et est donc essentiellement qualitatif. De même en **physique mécanique**, la proportionnalité entre la force et l'accélération linéaire, ou entre le couple et l'accélération angulaire, exprime une relation qualitative. C'est seulement en cas de besoin que le coefficient de proportionnalité sera explicité en utlisant la masse effective M dans la formule (6) ou le résultat du calcul du moment d'inertie I dans la formule (7), qui dans le cas général implique le calcul d'une intégrale double. D'ailleurs, le raisonnement qualitatif n'a-t-il pas son origine dans la physique qualitative?

1.3 Vers les notions de niveaux d'abstraction et de complexité

Dans la plupart des domaines-RP, l'abstraction apparaît de manière évidente dans la *définition des concepts* eux-mêmes, comme nous l'avons montré pour les trois domaines ci-dessus.

Si le domaine utilise aussi des *facteurs*, il apparaît que tout facteur introduit un *niveau d'abstraction intermédiaire* entre les concepts intervenant dans les équations qui le définissent. Par exemple, dans la formule (1), ou de manière équivalente dans les formules (2) et (3) en **génie économique**, ou dans les formules (6) et (7) en **physique**, nous avons (voir figure 1):

- au bas de la hiérarchie, des concepts de base qui «explicitent des technicalités» si nécessaire: le taux d'intérêt et le nombre de périodes en génie économique, la distribution des masses à l'intérieur du corps en physique;
- au haut de la hiérarchie, des concepts plus fondamentalement reliés au problème en cours de résolution: en

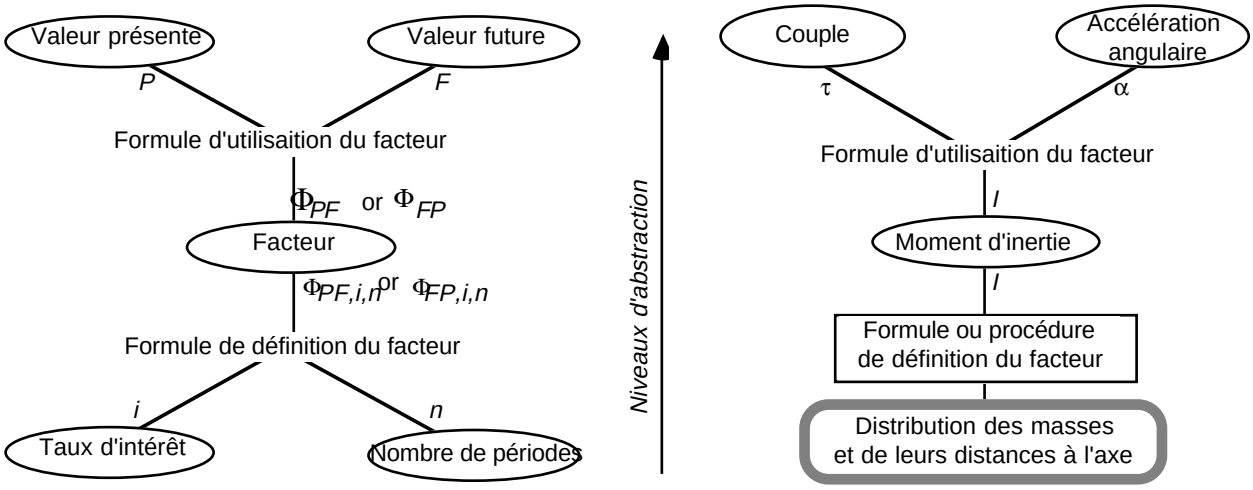

Figure 1 — Représentation d'un facteur comme concept.

génie économique les valeurs présente et future de l'investissement concerné, et en physique la force et l'accélération, ou le couple et l'accélération angulaire;
- entre ces deux niveaux, un niveau intermédiaire introduit par le facteur: Φ_{FP} ou Φ_{PF} en génie économique, ou *M* ou *I* en physique mécanique.

Alors que le facteur était au départ simplement une variable de calcul intermédiaire [voir les formules (2) et (3), ou (6) et (7)], ce statut le fait apparaître comme un *concept à but pédagogique*, qui établit une distinction claire entre:
- l'aspect quantitatif, calculatoire de la définition d'un facteur,
- et l'aspect pratique, qualitatif de l'utilisation d'un facteur dans un problème du domaine.

Cela corrobore la théorie plus générale [Lenat & al., 1979; Malec, 1989] selon laquelle l'utilisation de multiples *niveaux d'abstraction* facilite le processus de modélisation et simplifie les inférences qu'on peut faire sur les concepts d'un domaine.

Notre approche peut être généralisée de manière très intéressante, du moins dans certains domaines où on peut utiliser des facteurs de plus haut niveau définis à partir des premiers. En effet, en **génie économique**, au-dessus de Φ_{FP} et Φ_{PF}, les facteurs exprimant les valeurs présente et future d'une série de montants identiques (et vice-versa) sont une première généralisation de cette hiérarchie de concepts. Par exemple, le facteur Φ_{AP} défini en (4) est en fait une somme de facteurs Φ_{FP} :

$$\Phi_{AP,i,n} = \sum_{k=1}^{n} \Phi_{FP,i,k} = \sum_{k=1}^{n} (1+i)^{-k} = \frac{(1+i)^n - 1}{i\,(1+i)^n}$$

où la dernière expression résulte du calcul de la série géométrique indiquée. Cet exemple constitue une preuve de (4), mais aussi et surtout montre que le facteur Φ_{AP} se trouve à un plus haut niveau que Φ_{FP}. Il est important de noter qu'il s'agit ici d'un *niveau de complexité* plutôt que d'abstraction, puisque ce niveau traduit la manière de définir et de calculer le facteur Φ_{AP}. De manière similaire, en **physique**, le moment d'inertie d'un corps complexe peut être (et est souvent) calculé comme la somme de moments d'inertie élémentaires, et se situe donc à un plus haut niveau de complexité.

2 Connaissance en résolution de problèmes

Pour décrire la connaissance en résolution de problèmes (RP), nous présentons maintenant les caractéristiques générales de la modélisation de cette connaissance dans un logiciel-RP (section 2.1). Comme nous l'avons fait pour la section 1, nous présentons ensuite notre modèle dans les domaines du génie économique et de la physique (section 2.2), et nous montrons sur des exemples comment l'abstraction et la complexité sont utilisées en résolution de problèmes (section 2.3).

2.1 Généralités

Le second type de connaissance est spécifique aux domaines-RP [Ganeshan & al., 2000; Gertner & VanLehn, 2000], et donc aux logiciels-RP. Nous l'appelons *connaissance en résolution de problèmes* (PSK pour *problem-solving knowledge*). Elle contient tous les processus utilisés pour résoudre un problème résultant de l'instantiation d'une situation pratique basée sur la connaissance du domaine [Kowalski, 1979; Patel & Kinshuk, 1997]. En d'autres termes, pour pouvoir résoudre un problème, la connaissance en RP doit s'appuyer sur un bagage théorique préalable, qui se trouve dans la connaissance du domaine. Les processus stockés dans PSK peuvent se représenter de diverses façons, telles que la logique [Kowalski, 1979], les réseaux procéduraux [Brown & Burton, 1978], les réseaux sémantiques avec attachements procéduraux, les réseaux de transition (augmentés), les règles de production [Goldstein, 1979; Anderson & Reiser, 1985], etc.

Les activités principales du système centrées sur ce type de connaissance consistent à:
- fournir les outils calculatoires et inférentiels pour résoudre un problème, que cette résolution soit effectuée par le système ou par l'apprenant;
- fournir les outils inférentiels pour suivre et guider un étudiant dans une session de RP.

L'avantage majeur de la séparation entre DK et PSK — connu depuis longtemps — est de distinguer le domaine lui-même des habiletés utilisées pour résoudre un problème pratique dans ce domaine, ce qui simplifie ainsi le processus d'apprentissage. Cette séparation de la connaissance entre DK et PSK est, par nature, commune à tous les domaines-RP. C'est pourquoi nous pensons qu'un logiciel-RP, qui vise à aider l'étudiant à apprendre comment résoudre des problèmes, devrait également la mettre en œuvre (ce qui n'est pas toujurs le cas).

De plus, en suivant la trace de Lelouche et Morin [1997], nous pouvons mettre à profit — nous pensons de manière nouvelle — cette séparation entre DK et PSK pour définir, dans un logiciel-tuteur-RP, quatre *modes d'opération génériques*, fondés sur le type de connaissance impliqué (DK ou PSK) et sur qui la «produit» ou l'«engendre» (le système ou l'étudiant).
- Dans le *mode de présentation du domaine*, l'étudiant demande au système de le renseigner sur un élément théorique du domaine, et le système réagit en transférant à l'étudiant l'information ou la connaissance demandée. La connaissance mise en œuvre dans ce mode est toujours dans DK, et est produite par le système.
- Dans le *mode démonstration*, l'étudiant demande au système de résoudre un problème pratique ou de le guider dans la résolution d'un tel problème. Dans le premier cas, le problème en question a sans doute été posé par l'étudiant, tandis que dans le second c'est sans doute le système qui l'a posé. Mais dans les deux cas, la connaissance mise en œuvre provient esentiellement de PSK, et est produite par le système.
- Dans le *mode question de cours*, le système demande à l'étudiant de développer un élément du domaine, et

l'étudiant exprime dans ses termes la compréhension qu'il a de cet élément. En cas de besoin, le système peut intervenir pour corriger cette compréhension. La connaissance mise en œuvre est empruntée à DK, et est produite par l'étudiant.

- Enfin, dans le *mode exerciseur*, le système demande à l'étudiant de résoudre un problème pratique. L'étudiant le résoud étape par étape, montrant ce qu'il comprend de la connaissance en RP impliquée et de la connaissance du domaine associée. Si nécessaire, le système peut décider d'intervenir pour l'aider à atteindre son but ou pour le corriger. La connaissance mise en œuvre est essentiellement dans PSK, et est produite par l'étudiant.

2.2 Application à quelques domaines

Plusieurs activités de RP sont indépendantes du domaine, par exemple:

1. identifier et instancier les données du problème;
2. identifier et instancier le résultat attendu;
3. appliquer une formule;
4. appliquer un théorème.

Tout domaine-RP possède également ses propres activités de résolution de problèmes, donc dépendant du domaine en question. Par exemple en **génie économique**, de telles activités sont:

5. tracer un diagramme temporel pour représenter les instants significatifs du problème;
6. comparer des montants situés à la même date;
7. comparer des montants situés à des dates différentes;
8. additionner des montants situés à la même date;
9. additionner des montants situés à des dates différentes;
10. choisir une date de référence;
11. déplacer un montant d'une date à une autre;
12. agréger une série de montants périodiques en un montant unique équivalent;
13. distribuer («exploser») un montant en une série de montants périodiques équivalents.

De même, des activités du sous-ensemble ci-dessus de la **physique mécanique** sont:

14. calculer un couple;
15. calculer une accélération angulaire;
16. calculer un moment d'inertie.

2.3 Utilisation de l'abstraction et de la complexité

Dans de nombreux cas, une activité de RP peut se reformuler en une activité différente, de *niveau d'abstraction* plus bas, parce que les termes ou les concepts utilisés sont plus immédiats, plus terre-à-terre, plus proches du problème à résoudre. Par exemple, en physique mécanique, en supposant que le couple et le moment d'inertie d'un solide donné sont connus (soit donnés, soit déjà calculés), l'activité «calculer l'accélération angulaire» (activité 15) se reformulerait en «appliquer la formule (7)», qui est une instance de l'activité 3, de plus bas niveau. Un système tuteur en résolution de problèmes de physique est décrit dans [Gertner & VanLehn, 2000].

Parfois, une tâche de RP peut se diviser en tâches plus petites, ce qui nous permet d'introduire la notion de *niveaux de complexité* dans les tâches aussi. Par exemple, en **génie économique**, la comparaison de deux montants situés à des dates différentes demande:

- d'abord, de choisir une *date de référence* à laquelle effectuer la comparaison;
- ensuite, de déplacer un des montants (ou les deux) de sa date actuelle à la date de référence;
- enfin, de comparer les montants, maintenant situés tous les deux à la même date de référence.

Ces sous-tâches (de types 10, 11, et 6 respectivement dans la liste ci-dessus) apparaissent donc comme étant de complexité plus basse que la tâche initiale (de type 7). Cependant, il est bon de noter que, bien que la tâche 7 soit plus complexe que la tâche 6 (celle-ci est une partie de celle-là), ces deux tâches s'énoncent en utilisant le même *niveau d'abstraction.*

Il se peut aussi que certaines tâches de bas niveau ne puissent apparaître *que* comme sous-tâches d'une tâche de plus haut niveau. Par exemple, toujours en **génie économique**, la tâche «tracer un diagramme temporel» (type 5 ci-dessus) demande d'exécuter les sous-tâches suivantes, qui ne peuvent être accomplies que comme parties de cette tâche (d'où leur identification dans cet article de 5a à 5d plutôt qu'en utilisant de nouveaux numéros):

5a. tracer un axe des temps englobant tous les instants impliqués par les données du problème;
5b. tracer des flèches représentant les montants mis en œuvre dans les données du problème;
5c. si nécessaire, fractionner un montant (ou chaque montant d'une série) pour simplifier les calculs;
5d. tracer une flèche qualitative spéciale représentant le résultat attendu des calculs à effectuer.

Dans ce cas, l'activité 5 est à la fois d'un plus haut niveau de complexité et d'un plus haut niveau d'abstraction que n'importe laquelle de ses sous-activités.

3 Connaissance tutorielle

Nous présentons brièvement ici la connaissance tutorielle pour aider le lecteur à mieux cerner les liens qu'a cette connaissance avec DK et PSK. Comme pour DK et PSK, nous rappelons d'abord ce qu'est la connaissance tutorielle (section 3.1), et ensuite comment l'abstraction est utilisée dans ce contexte (section 3.2). Cependant, comme cette connaissance est normalement indépendante du domaine, nous ne donnons pas d'exemple de domaine d'application.

3.1 Généralités

La *connaissance tutorielle* (TK pour *tutoring knowledge*) contient tous les processus tutoriels existant dans le système. Elle n'est pas directement reliée au domaine ou à la résolution de problèmes, mais est là pour aider l'étudiant à comprendre, assimiler, et maîtriser

la connaissance contenue dans DK et dans PSK [Gagné & al., 1992; Gagné & Trudel, 1996].

Les activités principales du système qui utilisent TK sont:

- ordonner et formatter les éléments à présenter à l'étudiant (conception pédagogique);
- monitorer une session tutorielle, c'est-à-dire déclencher les différents processus tutoriels en fonction du but tutoriel du système et des actions de l'étudiant; ce monitoring peut nécessiter de donner des explications, de poser des questions, de changer de type d'interaction, etc.;
- dans un domaine-RP, pendant que l'étudiant résout un exercice, monitorer les activités de RP de l'étudiant: comprendre et évaluer ces activités, donner des conseils pour les corriger ou les optimiser, mettre sur la voie ou résoudre partiellement l'exercice en question (lorsque cette activité est requise par l'étudiant ou par le module tutoriel), etc.;
- analyser en permanence la progression de l'étudiant pour entretenir sa motivation et améliorer l'efficacité des interventions tutorielles.

L'avantage de séparer la connaissance tutorielle de la connaissance à apprendre, et donc à enseigner, a été souligné il y a longtemps [Goldstein, 1977; Sleeman & Brown, 1982; Clancey, 1986; Wenger, 1987], et réside dans la réutilisabilité de TK dans divers domaines. Dans le cas des domaines-RP, le domaine à enseigner comprend évidemment DK et PSK; en effet, le terme «connaissance du domaine» s'applique à DK s'il désigne le type de connaissance, et à DK + PSK s'il désigne la connaissance à acquérir par l'étudiant. Ainsi, comme indiqué dans l'introduction, la connaissance du système dans un logiciel-RP se retrouve séparée en trois catégories plutôt qu'en deux.

3.2 Utilisation de l'abstraction

Nous croyons que les processus tutoriels sont déclenchés par des *buts tutoriels* qui dépendent de l'environnement éducatif et du contexte d'apprentissage en cours. Le rôle des buts tutoriels a été discuté dans plusieurs articles, certains parmi les plus récents traitant d'ontologies de tâches et de processus éducatifs [Mizoguchi & al., 1996; Mizoguchi, 1999]. Dans l'état actuel de notre recherche, notre postulat est le suivant: la (ou les) hiérarchie(s) sous-jacente(s) qui guident la manière dont les processus tutoriels interagissent entre eux n'est pas reliée aux processus *eux-mêmes*, mais plutôt au *but courant à atteindre* au moment où ils sont invoqués. Ce but courant varie au cours de la session, selon les actions effectuées ou difficulés rencontrées par l'étudiant, en suivant une hiérarchie construite dynamiquement et fondée sur l'abstraction. Si notre postulat se vérifie, alors la *structure dynamique des buts et sous-buts éducatifs* — qui dépend elle-même des désirs ou choix de l'étudiant, de l'objectif sous-jacent principal du système, des habiletés générales de l'étudiant, de son état (p. ex. de fatigue) et de sa performance au moment courant, etc. — déterminera la succession des processus tutoriels qui seront activés et des interactions tutorielles qui se dérouleront.

À notre connaissance, l'utilisation des niveaux d'abstraction pour induire une hiérarchie dynamique de buts tutoriels est nouvelle, de même que la supposition qu'une telle hiérarchie jouera un rôle majeur dans l'activation des divers processus tutoriels et des interactions système–apprenant. Les buts tutoriels ont été utilisés par Towle [2000], mais uniquement pour des simulations éducatives, pas pour des processus tutoriels en général (dont les simulations ne sont qu'un exemple parmi d'autres).

4 Intérêts éducatifs des niveaux d'abstraction et de complexité

Dans les sections précédentes, nous avons décrit les trois types de connaissance mis en œuvre dans un logiciel-RP, en montrant «à l'occasion», de manière «impressionniste», comment les notions d'abstraction et de complexité appartaissent dans la desfription de ces connaissances. une approche basée sur les niveaux d'abstraction et de complexité pour faciliter la modélisation des trois types de connaissance mis en œuvre dans un logiciel-RP. Dans cette section-ci, après avoir clarifié de manière plus systématique les niveaux d'abstraction et de complexité dans la section 4.1, nous présentons les intérêts éducatifs de notre modèle. Plus spécifiquement, les sections 4.2 à 4.4 reprennent les trois types de connaissance présentés aux sections 1 à 3, en mettant davantage en lumière, en cas de besoin, l'abstraction, la complexité, et/ou leur organisation en niveaux. Enfin, la section 4.5 récapitule cette discussion en soulignant quelques intérêts pédagogiques supplémentaires des niveaux d'abstraction et de complexité.

4.1 Définition des niveaux d'abstraction et de complexité

Dans les trois premières sections, nous avons seulement fait référence aux niveaux d'abstraction et de complexité. Ici, nous essayons de définir ces notions de manière plus précise et plus généralement applicable. Les deux notions s'appuient sur l'idée commune de *raffinement*, mais elles diffèrent dans la manière d'effectuer ce raffinement: de manière générale, l'abstraction est basée sur, ou fait référence à, l'*expressivité* ou l'*étendue*, tandis que la complexité fait référence à, ou est basée sur, le *nombre de composants* mis en œuvre. Voyons cela pour chacun des trois types de connaissances.

Pour les **concepts du domaine**, en prenant la *géométrie* comme exemple, un polygone a un plus haut niveau d'abstraction qu'un triangle ou qu'un carré, parce que le nombre de côtés d'un polygone n'est pas précisé, mais un plus bas niveau d'abstraction qu'un ensemble de segments, parce que les segments constituant un polygone doivent en outre être adjacents deux à deux. Dans le même domaine, un carré a un plus haut niveau de complexité qu'un triangle, parce qu'il a plus de côtés, et aussi parce que des contraintes (de taille et d'angles) affectent ces côtés. Si nous prenons maintenant le *génie économique* comme autre exemple de domaine, nous avons vu que les facteurs Φ_{FP} et Φ_{AP} s'expriment avec

le même niveau d'abstraction, mais que Φ_{AP} a un plus haut niveau de complexité, à cause de la manière de le définir et de le calculer. En restant dans le domaine proprement dit, une distinction analogue entre les niveaux d'abstraction et les niveaux de complexité affecte les **relations entre concepts**. Par exemple, en *géometrie*, l'adjacence entre angles est une relation de plus bas niveau d'abstraction que la similarité entre triangles. La similarité de polygones, pour sa part, a le même niveau d'abstraction que la similarité entre triangles, mais un plus haut niveau de complexité.

Pour les **activités de résolution de problèmes**, nous avons des distinctions similaires, comme la section 2.2 l'a montré en s'appuyant sur plusieurs exemples.

Enfin, cela reste vrai pour les **processus tutoriels**, les plus perceptibles étant les interactions entre système et étudiant. Par exemple, la tâche–système (ou humaine...) consistant à guider un étudiant en train de résoudre un problème se révélera être d'un niveau de complexité plus élevé si l'étudiant rencontre des difficultés, bien que le niveau d'abstraction de ce processus ne dépende pas de l'étudiant lui-même ou du problème en cours de résolution. D'un autre côté, la tâche conmsistant à réagir à une requête d'étudiant pour un conseil ou une explication est d'un niveau d'abstraction plus bas que ce guidage général. Toutefois, là encore, la complexité de cette tâche dépendra naturellement de la requête spécifique de l'étudiant (certaines questions simples à formuler demandent parfois des réponses très complexes!), mais aussi (surtout?) de la clarté de la requête et de la manière dont l'étudiant est ou n'est pas satisfait de la réponse initiale du système.

De telles distinctions fondées sur les niveaux ont également été faites, par exemple, par Mizoguchi et ses collègues [Ikeda & Mizoguchi, 1994; Mizoguchi, 1999]. Il est important de souligner que, même si la proposition «A a un plus haut niveau d'abstraction que B» est claire et peut être vraie, nous pensons que le *nombre* de niveaux d'abstraction entre A et B est par essence non défini, ce nombre dépendant de la modélisation effectuée. A fortiori, il serait encore plus illusoire d'essayer d'affecter une *valeur numérique* à ces niveaux: en effet, de telles valeurs seraient tout à fait arbitraires.

4.2 Modélisation du domaine

La définition des concepts du plus simple au plus complexe induit de manière évidente et connue un ordre de présentation des éléments de DK. De même et en outre, la hiérarchie des facteurs décrite à la section 1 pour le génie économique ou la mécanique nous permet d'en déduire un *ordre pour la présentation des facteurs* à l'étudiant, du niveau le plus bas (le plus simple) vers le plus haut, c'est-à-dire par complexité de compréhension croissante. Cela ne signifie pas qu'un tel ordre soit unique, ni même le meilleur (p. ex. les intérêts personnels de l'étudiant pourraient rendre un autre ordre plus motivant pour lui), mais il est justifié par notre modèle. Cet ordre de présentation peut lui-même induire un *ordre possible des prérequis*. Par exemple, si un étudiant éprouve des difficultés à utiliser adéquatement Φ_{AP}, a-t-il bien maîtrisé Φ_{FP}, facteur conceptuellement plus simple? S'il ne comprend pas ce qu'est un couple, a-t-il bien compris ce qu'est une force?

De plus, les niveaux d'abstraction induits par les facteurs permettront au système tuteur de mettre en œuvre une *modélisation plus fine des erreurs*. Par exemple, la source d'une erreur de compréhension à propos de l'une des deux relations dans l'équation (2) ou (3) ou (7) (voir aussi la figure 1) est relativement facile à identifier, en utilisant le facteur correspondant, soit comme une erreur de définition, soit comme une erreur d'utilisation; au contraire, une erreur à propos de l'équation globale (1), où les relations de définition et d'application ne sont pas explicites, ne peut pas être diagnostiquée aussi finement. De même, une erreur faisant intervenir un facteur Φ_{AP} pourra être identifiée comme erreur de définition ou comme erreur d'utilisation, mais pourra *aussi* être éventuellement diagnostiquée comme résultant d'une maîtrise insuffisante du facteur plus simple Φ_{FP} comme concept (qui pourra à son tour être diagnostiquée comme liée soit à la définition, soit à l'utilisation, de ce concept). De manière analogue, en physique, si l'étudiant trébuche sur des concepts tels que l'accélération angulaire ou le moment d'inertie, a-t-il maîtrisé les concepts plus simples (quoique similaires) d'accélération ou de masse?

Les niveaux d'abstraction et de complexité sur les éléments du domaine (concepts et relations, éventuellement incluant les facteurs) peuvent ensuite être utilisés pour introduire différents niveaux d'abstraction dans les *explications*. De telles explications ont alors des chances d'être plus personnalisées, plus adaptées aux questions de l'étudiant (car exprimées au bon niveau de détail), et aussi de déclencher les rappels éventuellement nécessaires à l'étudiant (eux aussi au bon niveau).

4.3 Modélisation de la résolution de problèmes

Les activités de RP brièvement présentées à la section 2 introduisent de manière naturelle les niveaux d'abstraction et de complexité. En effet, la résolution d'un problème peut en général se décomposer, éventuellement de plus d'une manière, en un certain nombre d'étapes principales, pouvant à leur tour être divisées en sous-étapes plus simples. Comme expliqué à la section 4.1, chaque sous-activité dans ce cas peut être soit plus simple (niveau de complexité plus bas), soit plus concrète (niveau d'abstraction plus bas), soit à la fois plus simple et plus concrète, que l'activité dont elle fait partie.

Dans une première étape de développement, ces hiérarchies d'abstraction et de complexité, mises en place à la fois pour les éléments du domaine (DK) et pour les problèmes à résoudre (PSK), faciliteront la *définition des types d'exercices* à implanter dans notre système tuteur, et faciliteront le *choix du type d'exercice* à administrer à l'étudiant (tâche du module tuteur). Plus tard, lorsque le système de base sera opérationnel, les mêmes hiérarchies nous aideront à développer un *générateur d'exercices* automatisé mettant en œuvre les différents éléments du domaine ou de la RP que l'étudiant sera censé maîtriser. Cette approche l'aidera alors à acquérir un esprit plus

critique sur les importances relatives du *savoir-faire* (c'est-à-dire la résolution de problèmes, ou PSK) et du *savoir-savoir* (c'est-à-dire la connaissance du domaine, ou DK).

Comme pour les éléments du domaine, de tels niveaux d'abstraction et de complexité nous permettront d'introduire de *multiples niveaux d'explications* à propos du problème à résoudre, variant à la fois en abstraction (niveau de focalisation, termes utilisés, références effectuées aux éléments du domaine) et en complexité (quantité de détails, références éventuelles aux sous-étapes de résolution).

Enfin, une telle approche amènera l'étudiant à se concentrer spécifiquement sur les *activités* pour lesquelles il a le plus besoin de soutien, avec les niveaux d'abstraction et de complexité appropriés à sa situation et à son cas particuliers.

<table>
<tr><td colspan="2">Mode de fonctionnement</td><td colspan="2">Mode de présentation du domaine</td><td>Mode démonstration</td><td colspan="2">Mode question de cours</td><td>Mode exerciseur</td></tr>
<tr><td colspan="2">Type principal de connaissance impliqué</td><td colspan="2">Connaissance du domaine</td><td>Connaissance en résolution de problèmes</td><td colspan="2">Connaissance du domaine</td><td>Connaissance en résolution de problèmes</td></tr>
<tr><td colspan="2">But principal de l'étudiant</td><td colspan="3">Apprendre</td><td colspan="3">Évaluer et vérifier son apprentissage</td></tr>
<tr><td colspan="2">Direction du transfert de la connaissance</td><td colspan="3">Système → Étudiant</td><td colspan="3">Étudiant → Système</td></tr>
<tr><td rowspan="4">Interaction typique</td><td>Déclencheur (démarrage)</td><td colspan="3">L'étudiant demande au système...
des renseignements sur un élément théorique du domaine / de résoudre un problème pratique ou de le guider dans sa résolution</td><td colspan="3">Le système demande à l'étudiant...
de développer un élément du domaine / de résoudre un problème pratique</td></tr>
<tr><td>Transfert de connaissance</td><td colspan="3">Le système présente...
l'élément demandé / une solution possible au problème demandé</td><td colspan="3">L'étudiant présente sa vision de...
l'élément demandé / une solution possible au problème demandé</td></tr>
<tr><td>Résultat (fermeture)</td><td colspan="3">L'étudiant exprime sa compréhension...
de l'élément / de la solution du problème</td><td colspan="3">Le système évalue les réponses de l'étudiant, et éventuellement les corrige</td></tr>
</table>

Figure 2 — Caractéristiques des quatre modes de fonctionnement typiques d'un logiciel-RP.

4.4 Modélisation du tutorat

Comme indiqué à la section 2.1, la distinction entre DK et PSK conduit naturellement à définir quatre modes de fonctionnement génériques. La figure 2 rappelle leurs caractéristiques essentielles.

Les buts tutoriels visés par le système (voir section 3) se traduiront en général en une *chaîne d'invocations successives* de processus tutoriels, et donc d'interactions tutorielles. En effet, le système peut être amené à changer temporairement de type d'interaction, par exemple pour réagir aux actions ou aux requêtes de l'étudiant, éventuellement les affiner ou les préciser, voire les modifier. La place limitée de cet article ne permet pas de donner d'exemple, mais [Morin & Lelouche, 1998], par exemple, contient un exemple détaillé de dialogue tutoriel en génie économique. Cette chaîne d'invocations peut même être *récursive*, de manière directe ou non, selon les types de processus à enchaîner. Cependant, la complexité potentielle de cette chaîne est seulement apparente. En effet, à cause de la hiérarchie d'abstraction des buts tutoriels [Morin & Lelouche, 1998], chaque processus nouvellement activé le sera nécessairement avec une *étendue plus étroite* etéou une *complexité moindre* que celle du processus en cours (donc avec un niveau d'abstraction moindre), ce qui élimine naturellement le risque d'«oublier» le but tutoriel initial ou de déclencher une boucle infinie d'interventions.

Plus généralement, le tutorat de l'étudiant peut prendre la forme d'explication(s), de guidage, de mise sur la voie, ou même de résolution partielle de l'exercice sur lequel l'étudiant est en train de travailler. Les niveaux d'abstraction et de complexité du processus mis en œuvre dépendront du but tutoriel courant (voir section 3), et seront donc appropriés au cas particulier en cours. Nous pensons que cette approche est proche de celle de VanLehn et ses collègues [2000], bien qu'ils concentrent leur attention sur l'affaiblissement et l'approfondissement (résultats particuliers des interactions tutorielles) plutôt que sur le but pédagogique courant (la cause de ces interactions).

4.5 Intérêts généraux de ces niveaux d'abstraction et de complexité

Les niveaux d'abstraction ne sont certes pas nouveaux. Ce qui est nouveau, pensons-nous, est leur utilisation systématique, d'abord pour jeter un éclairage uniforme sur la conception et l'opération d'un système éducatif «intelligent» , et ensuite pour le rendre plus convivial une fois implanté.

D'abord, ces niveaux permettent de donner une

meilleure personnalisation aux interventions tutorielles du système, pour satisfaire à la fois les besoins de l'étudiant et les buts tutoriels du système, améliorant ainsi sa convivialité et son efficacité.

Ensuite, toutes les fonctionnalités présentées ci-dessus devraient conduire à des *interactions plus souples*, plus «naturelles», plus humaines avec l'étudiant. Cette capacité de mieux reproduire le comportement d'un bon enseignant humain contribue aussi à rendre le système plus convivial, et à augmenter ses chances d'être effectivement utilisé par l'étudiant.

Enfin, bien que cet aspect sorte du cadre de notre article, le raffinement des trois types de connaissances comme indiqué dans les sections 1 à 3 peut, à terme, conduire à la conception et à l'implantation d'un *modèle structuré des erreurs*, adoptant également l'abstraction comme ligne directrice, et finalement d'un *modèle hiérarchisé de l'étudiant* construit avec la même approche.

Conclusion

Cette présentation d'une structure possible de la connaissance dans les domaines-RP, qui met l'accent sur la séparation entre la connaissance du domaine et la connaissance en résolution de problèmes, rappelle comment il est naturel d'en déduire une théorie générale de fonctionnement d'un système-RP, à savoir les quatre modes de fonctionnement décrits aux sections 2.1 et 4.4.

En outre, les niveaux d'abstraction et de complexité mis en évidence dans l'ensemble de cet article peuvent servir de ligne directrice commune pour aider à trouver une représentation appropriée de chaque type de connaissance. À terme, ils pourront servir à créer des systèmes éducatifs réagissant avec des niveaux appropriés d'abstraction et de complexité aux activités courantes de l'éudiant, et donc à créer des systèmes éducatifs plus efficaces. Plus généralement, cette ligne directrice peut éclairer d'une manière unificatrice la conception du système, bien qu'elle n'ait jamais été utilisée de manière systématique dans la conception ou l'implantation d'un système tuteur.

Nous espérons ainsi apporter une modeste contribution au problème important et plus global de la définition d'une architecture générique pour les systèmes éducatifs intelligents.

Références

Anderson John R. (1986) *Cognitive Modelling and Intelligent Tutoring*. National Science Foundation (Washington, DC).

Anderson, John R. & B. J. Reiser (1985) "The LISP Tutor". *Byte* **10**, no. 4, p. 159-175.

Brodie, Michael L., John Mylopoulos & Joachim W. Schmidt, eds. (1984) *On Conceptual Modelling, Perspectives from Artificial Intelligence, Databases, and Programming Languages*. Springer Verlag (New York).

Brown, John S. & Richard R. Burton (1978) "Diagnostic models for procedural bugs in mathematical skills". *Cognitive Science* **2**, p. 155-192.

Clancey, William J. (1986) "Qualitative student models". *Annual Review of Computer Science* **1**, p. 381-450.

Clocksin, William F. & Christopher S. Mellish (1981) *Programming in Prolog*. Springer-Verlag (Berlin).

Findler, Nicholas V., ed. (1979) *Associative Networks*. Academic Press (Orlando, FL).

Frasson, Claude, Gilles Gauthier & Alan Lesgold, eds. (1996) *Intelligent Tutoring Systems*, Proceedings of the *Third International Conference, ITS'96*, Montréal (Canada), 12-14 June 1996. LNCS 1086, Springer (Berlin).

Gagné, Denis & André Trudel (1996) "A highly flexible student-driven architecture for computer-based instruction". [Frasson & al., 1996], p. 66-74.

Gagné, Robert M., Leslie J. Briggs & Walter W. Wager (1992) *Principles of Instructional Design*, fourth edition (1st edition published in 1974). Harcourt Brace Jovanovich (Orlando, FL).

Goldstein, Ira P. (1977) *The Computer as Coach: an Athletic Paradigm for Intellectual Education*. AI Memo 389, AI Laboratory, Massachusetts Institute of Technology (Boston, MA).

Ganeshan, R., W.L. Johnson, E. Shaw & B.P. Wood (2000) "Tutoring diagnostic problem solving". In [Gauthier & al., 2000], p. 33-42.

Gauthier, Gilles, Claude Frasson & Kurt VanLehn (eds.) (2000) *Intelligent Tutoring Systems.* Proceedings of *ITS 2000, the 5th International Conference on Intelligent Tutoring Systems*, Montreal (Canada), 19-23 June 2000. LNCS 1839, Springer (Berlin).

Gertner, A.S. & K. VanLehn (2000) "ANDES: a coached problem-solving environment for physics". In [Gauthier & al., 2000], p. 133-142.

Goldstein, Ira P. (1979) "The genetic epistemology of rule systems". *International Journal of Man-Machine Studies* **11**, no. 1, p. 51-77.

Ikeda, Mitsuru & Riichiro Mizoguchi (1994) "FITS: a Framework for ITS – A computational model of tutoring". *Journal of Artificial Intelligence in Education*, Vol. 5, no. 3, p. 319-348.

Kim, Won & Frederick H. Lochovsky, eds. (1989) *Object-Oriented Concepts, Databases, and Applications*. ACM Press, Addison-Wesley Publ. (Reading, MA).

Kowalski, R. (1979) *Logic for Problem Solving*. North-Holland (Berlin).

Lelouche, Ruddy & Jean-François Morin (1998) "Introduction of pedagogical concepts in domain modelling for an ITS". *European Journal of Engineering Education*, Vol. 23, No. 2 (June 1998), p. 255-271.

Lelouche, Ruddy & Jean-François Morin (1997) "Knowledge types and tutoring interactions in an ITS in a problem-solving domain". *FLAIRS 1997, Proc. of the 10th International FLorida Artificial*

Intelligence Research Symposium, Daytona Beach (Florida, U.S.A.), 10-14 May 1997, p. 62-66.

Lenat, D., Hayes-Roth, F., Klahr, P. (1979) *Cognitive Economy*. Working Paper HPP-79-15, Stanford Heuristic Programming Project. Stanford University (CA), June 1979, 46 pages.

Malec, J. (1989) " Knowledge elicitation during dynamic scene description ". *ACM–SIGART Newsletter : special issue on knowledge acquisition*, no. 108, April 1989, p. 162-163.

Mizoguchi, Riichiro (1999) "Ontology-awareness for intelligent instructional systems". Invited talk. *Advanced Research in Computers and Communications in Education, Volume 1* (G. Cumming, T. Okamoto & L. Gomez, eds.), Proceedings of the *7th International Conference on Computers in Education*, *ICCE 99*, Chiba (Japan), 4-7 November 1999. IOS Press (Amsterdam, Netherlands), p. 45-52.

Mizoguchi, Riichiro, Mitsuru Ikeda & Katherine Sinitsa (1996) "Task ontology design for intelligent educational/training systems". Proceedings of the ITS 1996 *Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs*, Montreal (Canada), p. 1-21.

Morin J.-F. (1998) *Conception of an intelligent tutoring system in cost engineering: knowledge representation, pedagogical interactions, and system operation.* Master Thesis, Département d'Informatique, Université Laval (Québec, Canada), 186 pages.

Morin J.-F., Lelouche R. (1998) "Agent-oriented tutoring knowledge modelling in a problem-solving ITS". Proc. of the ACM-SIGART *Workshop on Interaction Agents (IA 98)*, L'Aquila (Italy), 24 May 1998, p. 26-32.

Patel, Ashok & Kinshuk (1997) "Intelligent tutoring tools in a computer-integrated learning environment for introductory numeric disciplines". *Innovations in Education and Training International*, Vol. 34, No. 3, p. 200-207.

Sleeman, D. H. & John Seely Brown, eds. (1982) *Intelligent Tutoring Systems*. Academic Press (London).

Sowa, John F. (1984) *Conceptual Structures*. Addison-Wesley (Reading, MA).

Towle, B. (2000) "Using student task and learning goals to drive the construction of an authoring tool for educational simulations". In [Gauthier & al., 2000], p. 173-181.

VanLehn, K., R. Freedman, P. Jordan, C. Murray, R. Osan, M. Ringenberg, C. Rosé, K. Schulze, R. Shelby, D. Treacy, A. Weinstein & M. Wintersgill (2000) "Fading and deepening: the next steps for ANDES and other model-tracing tutors". In [Gauthier & al., 2000], p. 474-483.

Wenger, Étienne (1987) *Artificial Intelligence and Tutoring Systems*. Morgan Kaufmann (Los Altos, CA).