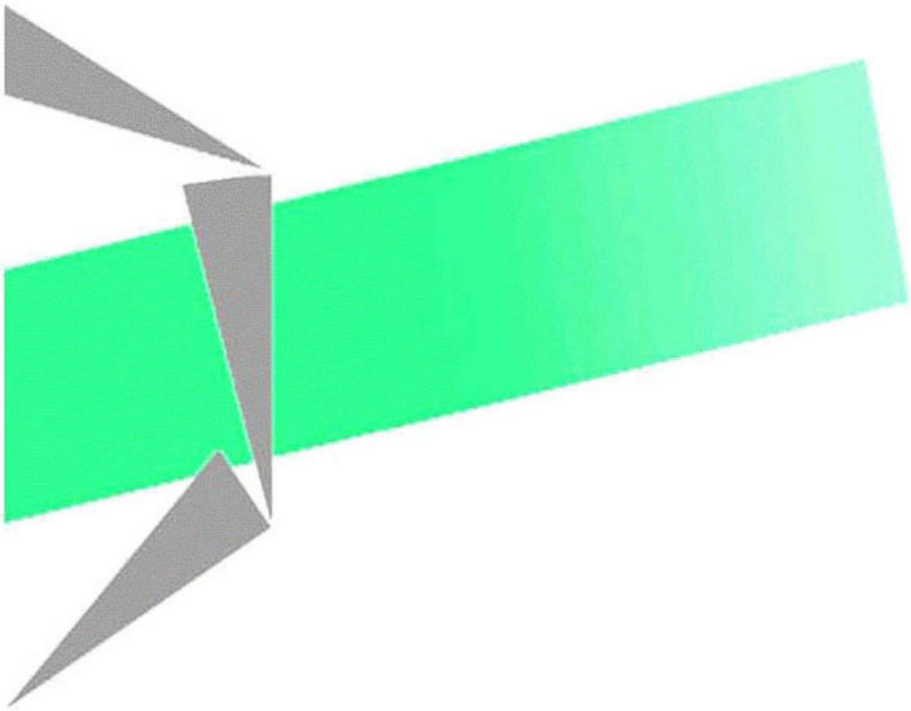
Les cahiers Leibniz



Informaticiens et didacticiens peuvent-ils travailler ensemble ?

Une étape du développement de l'EIAH APLUSIX : la mise en place de patrons d'exercices et d'une carte de tests.

Denis Bouhineau, Alain Bronner, Hamid Chaachoua
et Jean-François Nicaud

Laboratoire Leibniz-IMAG, 46 av. Félix Viallet, 38000 GRENOBLE, France -
ISSN : 1298-020X

n° 122
Mai 2005

Site internet : http://www-leibniz.imag.fr

Informaticiens et didacticiens peuvent-ils travailler ensemble ?

Une étape du développement de l'EIAH APLUSIX : la mise en place de patrons d'exercices et d'une carte de tests.

Denis Bouhineau*, Alain Bronner, Hamid Chaachoua*** et Jean-François Nicaud*.**

** Université Joseph Fourier - LEIBNIZ - MeTAH*
46 Av F. Viallet
38031 Grenoble Cedex
Prénom.Nom@imag.fr
*** LIRDEF - IUFM de Montpellier*
2 Place Godechot,
BP 4152, 34092 Montpellier Cedex
Prénom.Nom@montpellier.iufm.fr
**** IUFM – LEIBNIZ - MeTAH*
46 Av F. Viallet
38031 Grenoble Cedex
Prénom.Nom@imag.fr

RÉSUMÉ. *Au cours du développement de l'EIAH APLUSIX, il y a eu de nombreuses occasions d'un travail partagé entre informaticiens et didacticiens. Quelques-unes sont décrites succinctement, illustrant différentes formes de collaborations selon des schémas producteur-consommateur. Lors de la mise en place de patrons d'exercices et d'une carte de tests, une coopération plus approfondie a été possible, tout en respectant les disciplines de chacun. Cette mise en place de patrons d'exercices et d'une carte de tests est décrite longuement, elle est donnée comme exemple de cas où informaticiens et didacticiens ont pu travailler ensemble.*

MOTS-CLÉS : *algèbre, contrainte, domaine, hiérarchie, paramètre, pluridisciplinaire.*

Introduction

Au cours du développement de l'EIAH APLUSIX, et plus largement au sein de l'équipe bi-disciplinaire (didactique et informatique) où s'effectue ce développement, il y a eu de nombreuses occasions d'un travail partagé entre informaticiens et didacticiens. Pourtant, certaines des étapes de ce développement, et parmi elles, certaines primordiales, comme la définition du noyau de base (le micromonde d'édition et de raisonnement algébrique), se sont déroulées entre informaticiens, sans échange avec des didacticiens. D'autres étapes, comme la construction de conceptions d'élèves, se sont déroulées sur une chaîne de montage selon un schéma producteur-consommateur avec les informaticiens comme producteurs. D'autres étapes encore, par exemple la mise au point de diagnostic de production d'élèves, se sont opérées selon ce même schéma producteur-consommateur mais avec des rôles inversés, avec les didacticiens comme producteurs.

La dichotomie pourra paraître plus grande encore si l'on considère que les objets maniés lors de ces étapes par les didacticiens n'ont jamais réellement correspondu à ceux manipulés par les informaticiens. Au niveau souvent plus technique de l'informaticien se trouvent des objets computationnels, les objets du didacticien, même s'ils parlent de la même chose, sont d'un autre ordre, beaucoup plus abstraits et ils s'expriment de manière plus discursive et littéraire. Pour ces situations où il n'y a pas de véritable objet commun, du fait du niveau d'abstraction ou de la forme, des implicites véhiculés, etc., nous parlerons d'un schéma de travail producteur-consommateur hétérogène.

Pourtant, il y a eu travail en commun, synergie.

Une étape a été franchie lors de la construction de patrons d'exercices pour l'EIAH APLUSIX. Après une phase de travail selon le schéma producteur-consommateur hétérogène, le travail a pu s'effectuer en parallèle, sur des objets communs. La synergie est devenue symbiose.

Des questions se posent :

- ce travail aurait-il pu se faire sans didacticien ou sans informaticien ou selon un schéma producteur-consommateur hétérogène ?
- il n'y a pas eu d'objectif de travail ensemble où didacticien et informaticiens se seraient confondus ; chacun a gardé sa spécificité, son domaine de compétence. Est-ce que la « confusion » aurait été bénéfique ?
- plus généralement, l'EIAH est un domaine qualifié de pluridisciplinaire. Qu'est-ce que cela signifie ? Le travail présenté est-il paradigmatique ou constitue-t-il un objectif à atteindre et dépasser, ou encore est-il anecdotique, sans rapport avec la pluridisciplinarité ?

L'article, après un bref exposé des moments du travail commun informaticiens-didacticiens évoqués précédemment, décrira cette construction commune de patrons d'exercices, en montrant, d'une part, les évolutions qui se sont opérées en accord de

part et d'autre pour converger vers un objet commun de travail et, d'autre part, les constructions qui ont été développées, par la suite, indépendamment de part et d'autre, selon des directions orthogonales, celles de chaque discipline. Au moment de la conclusion, nous reviendrons sur les questions évoquées précédemment.

1. Premiers développements du projet APLUSIX

Le renouveau* du projet APLUSIX a commencé en Juin 2000 à Nantes, dans un laboratoire d'informatique, au sein d'une équipe ne comportant pas de didacticien. Le cœur du système et les concepts qui régissent l'environnement sont donc le fruit du travail d'informaticiens seulement. L'objectif était de produire un micromonde d'apprentissage de l'algèbre avec une interface ergonomique de manipulation d'expressions et de raisonnements algébriques fournissant des rétroactions épistémiques et pouvant vraiment aller en classe. Les rétroactions étaient définies à trois niveaux : au niveau syntaxique, l'environnement favoriser l'écriture d'expressions algébriques bien formées ; au niveau sémantique, l'élève était informé en permanence de la validité de ses calculs ; au niveau stratégique, des indicateurs marquaient la progression du travail. Pour une description plus complète, se référer à [Nic-2004]

En deux temps, Septembre 2001, et Septembre 2002, le projet APLUSIX s'est déplacé à Grenoble et a rejoint un laboratoire d'informatique, dans une équipe comportant des didacticiens. Les premiers travaux communs, informatique-didactique ont eu lieu en 2003 avec la description de règles de calculs erronés utilisées par les élèves. L'objectif en vue à plus long terme était la caractérisation de conceptions d'apprenants en algèbre. Ce travail s'est effectué sur la base de protocoles récoltés par l'environnement APLUSIX lors d'utilisations en classe, et visualisés dans l'environnement APLUSIX sous forme de films passés au magnétoscope. L'apport des informaticiens était le logiciel et ses fonctionnalités (enregistrement et visualisation de protocoles), les didacticiens faisant l'analyse et produisant les règles de transformations erronées. Un ensemble de plus de 600 heures d'utilisations APLUSIX a été récupéré lors d'une expérimentation à l'automne 2002. Un ensemble de 157 règles de transformations correctes et 67 règles de transformations erronées a pu être ainsi construit dans le domaine des mouvements dans les équations et inéquations linéaires.

Un second travail a succédé à la construction de cet ensemble de règles de transformation. Il concerne la mise au point d'un algorithme de diagnostic des productions des élèves. Ce travail a été entrepris côté informatique. Il a nécessité la définition de règles plausibles de transformations. L'ensemble précédent a été utilisé. Ainsi, l'apport des didacticiens a été l'ensemble des règles de transformations utilisées par les élèves.

---

*Avant 2000, le projet APLUSIX a produit des tuteurs pour les factorisations, cf. [Nicaud 1990]. En 2000, s'est opéré un changement radical par adoption d'une approche micromonde.

D'autres exemples pourraient être donnés, ils sont basés sur un schéma producteur-consommateur qui s'instancie indifféremment avec pour les producteurs les informaticiens ou les didacticiens, idem pour les consommateurs. A noter que si producteur et consommateur n'appartiennent pas au même milieu, en général, la matière apportée par le producteur nécessite une adaptation pour être utilisée par le consommateur, producteur et consommateur ne travaillent pas sur les mêmes objets. Les deux cas présentés précédemment ne font pas exceptions.

2. Génération automatique d'exercices

2.1. Planification du projet de génération automatique d'exercices et de tests

Différentes tâches ont été réparties au sein de l'équipe. Le tableau 1 donne les grandes étapes de ce projet avec les responsabilités entre les disciplines.

Tâches	Réalisation
Exploration du projet	En commun
Définition du langage de patrons d'exercices	Informaticiens
Implantation des primitives pour les patrons	Informaticiens
Développement du générateur d'exercices	Informaticiens
Construction des patrons	Didacticiens
Conception de la carte des tests	En commun
Interface générateur -carte des tests	Informaticiens
Remplissage de la carte des tests	Didacticiens
Notation des exercices	Informaticiens
Prise en compte du temps	En commun
Internationalisation de la carte des tests	En commun
Débogage	En commun

Tableau 1 : Planification des tâches

2.2. Principes de la génération des exercices : Notions de patron

Le but est de construire automatiquement des listes d'exercices pour l'entraînement, le diagnostic et, plus généralement, l'apprentissage relativement aux différents types de tâches algébriques (calculer, développer, factoriser, résoudre). Ces exercices tiennent compte, dans une certaine mesure, du curriculum français, sans y être attaché strictement, de manière à pouvoir être utilisé dans divers contextes scolaires, et sont de différents niveaux de difficulté en mettant en jeu différentes règles de transformation (distributivité, identités remarquables, etc.). Leur construction s'appuie essentiellement sur des concepts mathématiques.

Les exercices sont engendrés par des patrons d'exercices contenant des variables ou paramètres (comme la nature des nombres en jeu ou la taille des coefficients) et un tirage au hasard des valeurs de ces variables. Le patron lui-même est une expression algébrique, par exemple <<a(bx+c)+d(-ex+f)=gx+h>> ou <<x²-px+1=0>>. Les variables d'un patron sont associées à un domaine D défini par des contraintes restreignant le tirage aléatoire.

Le projet doit donc articuler une dimension didactique pour la détermination des familles d'exercices, des patrons, des variables et des domaines, et une dimension informatique pour la programmation et l'implémentation des patrons, des listes et des tirages au sort.

La notion de patron relève à la fois de la didactique des mathématiques et de l'informatique. Une hiérarchie emboîtée de patrons est à construire au niveau didactique et informatique. On dira que le patron reste abstrait quand il définit une famille encore paramétrée par des variables ; quand il conduit à des exercices effectifs par instanciation des variables dans le domaine considéré, nous parlerons de patron effectif.

2.3. La construction des patrons : la dimension didactique

Les patrons sont tout d'abord des formes algébriques et pourraient être considérés a priori indépendamment d'une sémantique (autrement dit regardés comme listes de symboles régis par une syntaxe) dans la mesure où leur nature est une expression algébrique (polynomiale ou égalité). Mais ils sont bien rattachés à une sémantique dans la mesure où ils vont être construits relativement à des problèmes algébriques de type : calculer, développer ou factoriser un polynôme, ou encore, résoudre une équation, une inéquation ou un système d'équations.

La structure commence avec les 5 grands types de problèmes qui correspondent au premier rang de la hiérarchie des patrons, cf figure 1.

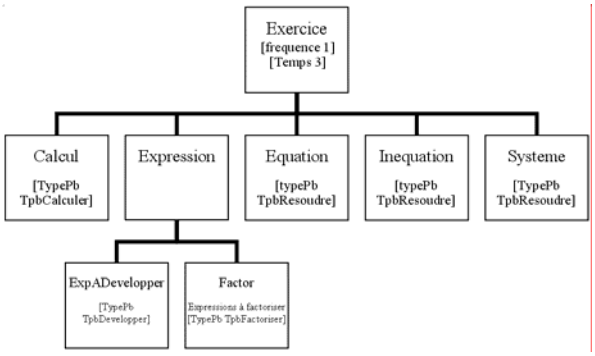


Figure 1. *Le haut de la hiérarchie des patrons.*

Développons l'exemple des patrons de type « Factor ». Ce type contient les patrons fabriqués en vue du genre de tâches : « Factoriser un polynôme P(x) d'une variable ». Chaque genre donne naissance à des patrons, encore abstraits, qui correspondent à des formes sur lesquelles les exercices seront construits. Autrement dit, ces patrons sont, dans un premier temps, élaborés en fonction de la forme qui est donc la première variable prise en compte. Toutes les formes ou expressions sont envisageables. Mais le but étant d'engendrer des exercices effectifs, pouvant être effectivement proposés à des élèves du contexte scolaire français ou étranger, nous avons restreint les formes en fonction de certaines sous-variables ou d'instances de

ces variables. L'idée est d'engendrer différentes complexités pour les expressions proposées. Il ne s'agit pas seulement d'une complexité intrinsèque liée à la forme, mais aussi liée à la difficulté de la factorisation. Par exemple (2x +9)(5x+1) + (3x - 4)(5x+1) + (7x +5)(5x+1) est moins complexe à factoriser que $\frac{2}{9}x^2 - 3$ alors que la forme de la dernière ne contient que deux termes. Les résultats de recherche en didactique de l'algèbre, l'étude des programmes de l'enseignement secondaire et les difficultés repérées par les enseignants conduisent à prendre en compte comme premières variables didactiques potentielles [Brousseau 1981, 1997] (c'est-à-dire des variables qui influencent la nature des stratégies et des règles utilisées) pour le genre de tâches : « Factoriser un polynôme P(x) d'une variable » :

- la diversité des règles à utiliser et les formes associées ;
- le nombre de transformations à mettre en œuvre ;
- la présence de transformations intermédiaire pour certains termes ou facteurs ;
- la profondeur de la factorisation ;
- le degré des polynômes ;
- le degré et la forme du facteur commun ;
- la présence ou non d'un facteur –1 ;
- la présence ou non d'un facteur 1 implicite.

La plupart de ces variables peuvent être reprises pour les autres grands types de problèmes algébriques, mais certaines sont spécifiques du genre « factoriser », comme la dernière de la liste précédente. La hiérarchie et la combinaison de ces variables conduisent à une nouvelle typologie emboîtée. En complexifiant les formes par un jeu raisonné des variables précédentes, nous avons obtenu 5 grands types abstraits dans le type Factor, se répartissant en 14 sous-catégories :

1. Type Factor 1 : Polynôme somme de termes constants et de monômes avec un facteur commun, autrement dit polynômes de forme générale AB+AC ou BA+CA avec A, B et C constants ou monômes de degré 1 ou plus.
   - Sous type 1 : A constant.
   - Sous type 2 : A monôme.

2. Type Factor 2 : Polynôme somme de produits formés d'un facteur commun de degré 1 et de facteurs constants ou monômes, autrement dit polynômes de forme générale AB+AC avec A du premier degré, B et C constants ou monômes de degré 1.

3. Type Factor 3 : Polynôme somme de produits du premier degré, autrement dit polynômes de forme générale AB + AC avec A du premier degré, B et C constant ou de degré 1.
   - Sous type 1 : B et C de degré 1 et de la forme ax+b.
   - Sous type 2 : B de degré 1 et de la forme ax+b ; C constant ou monôme.
   - Sous type 3 : B de degré 1 et de la forme ax+b ; C = 1 (Patrons de polynômes de forme générale AB + A avec A du premier degré).
   - Sous type 4 : B de degré 1 et de la forme ax+b ; C = -1 (Patrons de polynômes de forme générale AB - A avec A du premier degré).

- Sous type 5 : Le facteur commun A fait partie d'un carré A^2 (Patrons de polynômes de forme générale $aA^2 + AC$ avec A du premier degré).

- Sous type 6 : Le facteur commun A est du premier degré et nécessite une transformation intermédiaire pour être apparent.

4. Type Factor 4 : Polynômes développés liés aux identités remarquables

- Sous type 1 : Polynôme de la forme $A^2 + 2AB + B^2$ ou $A^2 - 2AB + B^2$ avec A et B constant ou du premier degré.
- Sous type 2 : Polynôme de la forme $A^2 – B^2$ avec A et B constant ou du premier degré.

5. Type Factor 5 : Polynôme complexe, somme de polynômes de types Factor 3 ou Factor 4. Le facteur commun n'est pas toujours apparent, une transformation intermédiaire et/ou une factorisation du type correspondant aux formes Factor 4 (liées aux identités remarquables) sont nécessaires. Le facteur commun est de la forme ax+b.

- Sous type 1 : Présence d'une forme $A^2 + 2AB + B^2$.
- Sous type 2 : Forme se ramenant à $A'^2 – (A^2 + 2AB + B^2)$.
- Sous type 3 : Présence d'une forme $A^2 – B^2$ à factoriser.

Les patrons se présentent ainsi comme des formes qui ouvrent encore des possibilités trop importantes ou pas assez réalistes dans certains contextes scolaires. Par exemple, en début d'apprentissage, proposer de factoriser $\frac{123456789}{987654321}x + \frac{123456789}{987654321}$ peut paraître incongru pour l'élève comme pour l'enseignant, même si cela pourrait être intéressant du point de vue didactique. Plus généralement les expressions doivent respecter la coutume didactique [Balacheff 1988] de certains niveaux de classe : Les nombres doivent être simples et les calculs simplifiables en général. Nous avons ainsi essayé de prendre en compte certaines clauses de ces coutumes ou de contrats didactiques plus spécifiques [Brousseau 1990]. De plus, à certaines phases d'apprentissage, les factorisations sur les coefficients est un enjeu. Pour obtenir des patrons effectifs, il s'avère nécessaire d'injecter d'autres variables pour définir le domaine D des coefficients, comme :

- le nombre de termes ;
- la nature des coefficients et leurs types d'écriture (entière, décimale, fractionnaire, etc.) ;
- les propriétés arithmétiques intrinsèques ou réciproques de certains coefficients (diviseurs, multiples, facteur commun, etc.).

Des questions se sont alors posées dans l'équipe. S'il était clair pour chacun d'entre-nous qu'il fallait prendre en compte la nature des coefficients, les choix des écritures mathématiques et les instructions du langage informatique ont fait l'objet de débats dont nous restituons quelques questions : Est-ce préférable de faire deux patrons ou de programmer directement un opérateur [+ ou –] pour avoir des termes a + b ou a – b ? Suffit-il de dire que b est un nombre relatif ? Engendre-t-on directement des rationnels en les déclarant comme tels, ou bien les engendre-t-on comme fractions à partir des entiers ? Par exemple un même patron, au niveau d'une

forme pour un développement <<(ax+b)(cx+d)>>, peut engendrer des patrons effectifs différents selon le domaine de coefficients D, celui-ci pouvant prendre les valeurs D = Z, D, Q.

Comment faire pour avoir des entiers relatifs « stricts » (éléments de Z-N), des rationnels stricts (éléments de Q-D), ou des décimaux éventuellement entiers, des rationnels éventuellement entiers ou décimaux stricts ? Comment obtenir indifféremment ces différents types de nombres ? Des problèmes mathématiques et informatiques s'entrecroisent ici.

Le choix de programmation du domaine est mixte en jouant à la fois sur la forme des coefficients eux-mêmes et sur une liste de conditions sur les coefficients. Par exemple, le premier sous-type élémentaire « Factor 1 » du type « Factor » correspond aux patrons de polynômes de forme générale : $ax^3 + bx^2 + cx + d$, certains coefficients pouvant être nuls. En jouant sur certaines variables précédentes, nous sommes tout d'abord parvenus à différents patrons rassemblés en deux sous-types, selon que le facteur commun est une constante ou contient un facteur x ou x^2 (Eval(ab) veut dire que le nombre ab sera écrit sous la forme évaluée et canonique. De plus, le domaine en question est celui des coefficients a, b, c et non celui de la variable x), cf tableau 2.

Type Factor 1 Sous type 1 : A Constant	**Type Factor 1 Sous type 2 :** A monôme
ax+ Eval(ab) sur le domaine Z	x^2+bx sur le domaine Z
Eval(ab)x-a sur le domaine N	ax^2+bx sur le domaine Z
Eval(ac)x+ Eval(bc) sur le domaine Z	Eval(ac)x^2+ Eval(bc)x sur le domaine Z
Eval(bc)+Eval(ac)x sur le domaine Z	ax^3+bx^2+cx sur le domaine Z
	Eval(ac)x^3+ Eval(bc)x^2 sur le domaine Z

Tableau 2. Sous patrons du type Factor

Ces deux types sont élémentaires et sont conformes au programme de la classe de 4ème en France.

La génération d'exercices effectifs de chacun de ces patrons demande encore une restriction du domaine des coefficients en prenant en compte la taille des nombres coefficients. Il est difficile de donner des critères de taille dans l'absolu. Nous nous sommes référés aux usages, notamment ceux issus des manuels. Par exemple, les exercices des manuels où l'on peut considérer les coefficients comme « petit » en classe de quatrième pour un entier relatif donnent un nombre relatif en valeur absolue inférieur ou égal à 12. Définir petit sur D n'est pas simple non plus, et une proposition serait : une partie entière en valeur absolue inférieure ou égale à 9 et une ou deux décimales. Définir petit sur Q pourrait être numérateur entier relatif en valeur absolue inférieure ou égale à 20 et dénominateur positif inférieur ou égale à 10. Ces questions ont fait l'objet d'échanges dans le groupe pour arriver à un compromis entre informaticiens et didacticiens.

Finalement, la prise en compte de toutes ces variables et les contraintes du langage de programmation choisi nous ont conduits à des patrons effectifs. Par

exemple, la forme finale pour le patron de type « Factor 1 Sous type 2» et de forme « Eval(ac)x²+ Eval(bc)x » sur le domaine D = Z est donné figure 2.

```
{[nom FactorDistSTD23Z]
 [sorteDe FactorDistSTD]
 [patron <<ax^2+bx>>]
 [domaine ((c entier+ petit)(d entier+ petit)(e entier* petit))]
 [avec ((<> c 1) (:= a (* d c)) (:= b (* e c)))]}
```

Figure 2. Un patron Factor 1 Sous type 2

Le langage utilisé traduit ici les relations : a = d.c et b = e.c avec c différent de 1. Un tirage aléatoire sur le patron précédent de nom FactorDistSTD23Z factor produit les exercices effectifs donnés figure 3.

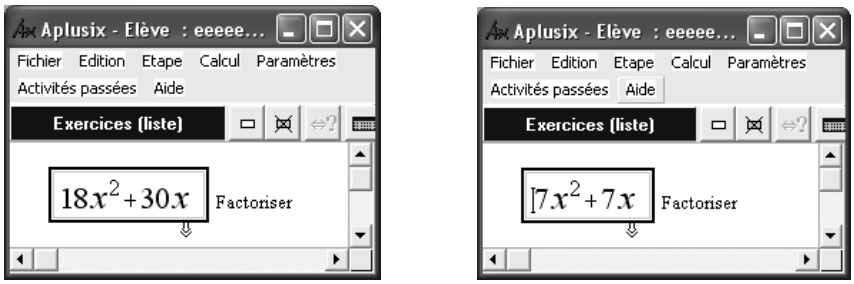


Figure 3. Tirage aléatoire sur le type Factor 1 Sous type 2Factor 1

Au final, une structure hiérarchique avec 436 patrons effectifs a été obtenue.

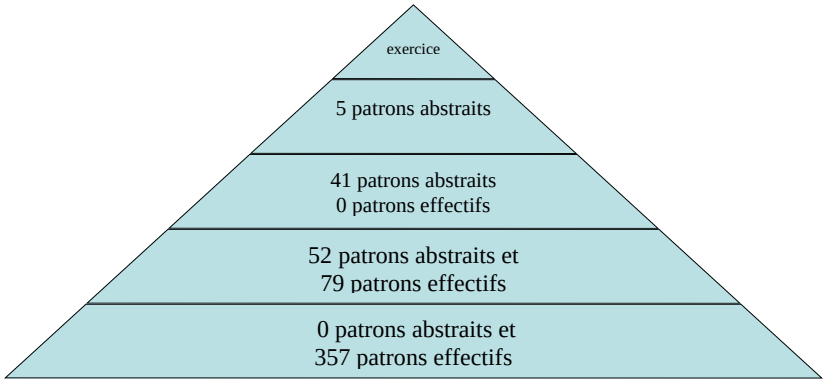


Figure 4. **Nombres de patrons par niveau hiérarchique.**

2.4. La construction des patrons d'exercices : la dimension informatique.

La figure 2 donne un exemple de patron. La structure utilisée comporte une dizaine de champs. Les deux premiers (Nom, SorteDe) permettent de définir une

hiérarchie de patrons et d'hériter d'attributs. Parmi les autres champs visibles dans l'exemple 2, le lecteur reconnaîtra le patron lui-même, la définition du domaine pour chaque variable et les contraintes à respecter. Parmi les champs non-visibles (hérités d'un patron plus près de la racine de la hiérarchie), notons le type de problème associé au patron, le temps moyen pour résoudre le problème, des commentaires définissant le patron pour fournir des explications, une pseudo fréquence d'apparition du patron (par rapport à une fréquence étalon).

Voici un autre exemple de patron. Il est lié au type de problèmes « résoudre une équation du premier degré » et permet d'exhiber d'autres contraintes à respecter.

```
{[nom PatronEqDg1Groupement17]
 [sorteDe PatronEqDg1Groupement1]
 [patron <<[a/b]x+[c/e]=d/e>>]
 [domaine ((a parmi (-5 -4 -3 -2 -1 1 2 3 4 5))(b parmi (1 2 3 4 5 6))
           (c  parmi  (-5  -4  -3  -2  -1  1  2  3  4  5  6  7  8  9))
           (d  parmi  (-5  -4  -3  -2  -1  1  2  3  4  5  6  7  8  9))
           (f parmi (2 3)))]
  [avec ((NEstPasDivisiblePar a b)(NEstPasDivisiblePar c b)
        (NEstPasDivisiblePar d b) (<> d c)(:= e (* f b)) (<> d e))]}
```

Figure 5. Un patron associé au genre « Résoudre »

Pour instancier un patron, nous utilisons un mécanisme de tirage aléatoire non trivial qui assure le respect des contraintes, en invalidant certains patrons et provoque une erreur interne si le patron ne peut être instancié après un nombre raisonnable d'essais. Il va de soi que l'erreur interne est récupérée, c'est un mécanisme transparent pour l'utilisateur. Par rapport au projet Wims [Perrin-Riou 2004], ou à la spécification d'IMS-QTI [IMS-QTI 2004], c'est une originalité. Ces conditions de refus d'un patron permettent des vérifications globales et facilitent l'expression de certaines propriétés. De plus, après instanciation des variables, des calculs supplémentaires sont effectués : suppressions des éléments neutres, utilisation des éléments absorbants. Ainsi, <<nax=ma>> pour n=-1, m=1 et a=3 fournit –3x=3, et non (-1)3x=1*3.

Au fur et à mesure de l'avancement du travail nous avons été obligés d'améliorer la robustesse de l'analyseur syntaxique et lexical que nous utilisions, ses reprises sur erreurs et les messages d'erreur qu'il fournissait, de telle sorte que les utilisateurs (didacticiens) gagnent de l'autonomie dans l'écriture des patrons.

3. Mise au point d'une carte de test

Un test pour l'élève consiste à choisir une famille d'exercices relevant du même type de problèmes (calculer, développer, factoriser, résoudre) en choisissant un élément dans diverses sous catégories en fonction de la difficulté et des notions ou règles en jeu. Une carte de tests est un ensemble de tests.

La mise au point de la carte des tests s'est effectuée comme celle des patrons, avec une première phase de définition du langage de description d'un test et d'une carte de test, puis une seconde phase dans laquelle les didacticiens ont instancié les

tests et la carte des tests, pendant que les informaticiens mettaient en place effectivement le langage de descriptions des tests et de la carte des tests ainsi que les algorithmes de tirage aléatoire et d'interface nécessaires.

Le langage de description d'un test est simple : il comporte essentiellement des informations permettant la construction d'une hiérarchie des tests, et des commentaires. La partie utile d'un objet « test » est la liste « exos » qui comporte les noms des patrons associés au test.

Les listes d'exercices sont localement mélangées pour garder une progression (s'il y en a une de prévue dans le test) mais aussi pour éviter des effets mémoire. Si l'on veut que les instances ne soient pas équiprobables, on installe des fréquences dans les patrons (cf. 3.4). Le nombre de patrons dans un test est déterminé par le temps. Arbitrairement un test dure 30min, chaque patron a défini un temps moyen de résolution, le tirage de la liste des exercices du test se fait tant qu'il reste du temps, en bouclant sur la liste des patrons associée au test.

Rappelons que dans APLUSIX, l'élève demande des exercices, soit en activité « Exercice » pour s'entraîner en ayant la vérification des calculs, soit en activité « test ». Il nous fallait construire une interface assez ergonomique permettant à l'élève de faire son choix pour obtenir une liste d'exercices, soit en mode exercice, soit en mode test. Nous avons mis en place une carte des tests (Figure 7) sous la forme d'un tableau de dimension 2 avec des noms sur les lignes et les colonnes ayant un sens dans l'organisation des concepts (mais peut-être pas immédiat pour l'élève), tel que chaque case de ce tableau corresponde à un test. A chaque case active du tableau est associée une liste d'exercices à tirer au sort. Quand on clique sur une case, on obtient une description synthétique de cette liste.

```
{[nom TestCalcul]
 [sorteDe CarteDesTests]
 [SuperFamille CalculNumerique]
 [NoLigne 1]}

; Calculs sur les entiers
{[nom TestCalcul1]
 [sorteDe TestCalcul]
 [description CalNumEntier]
 [exos ( PatronCalculCE1 PatronCalculCE2 PatronCalculCE3 )]
 [NoColonne 1]}

; Calculs sur les décimaux
{[nom TestCalcul2]
 [sorteDe TestCalcul]
 [description CalNumDecimaux]
 [exos ( PatronCalculCD1 PatronCalculCD2 PatronCalculCD3 )]
 [NoColonne 2]}
```

Figure 6. Définition de tests.

Cette carte sert :
- à voir les tests possibles (comme indiqué ci-dessus),
- à choisir un test pour le passer dans l'activité « Test » ou « Exercice »,
- à voir les scores de l'élève sur les tests passés.

Les exercices à gauche de la carte contiennent en général les tests les plus simples et ceux de droite les plus compliqués. En affichant la carte avec les meilleurs scores de l'élève dans les cases, il a une vue synthétique sur les endroits où il est passé et sur ce qu'il a fait.

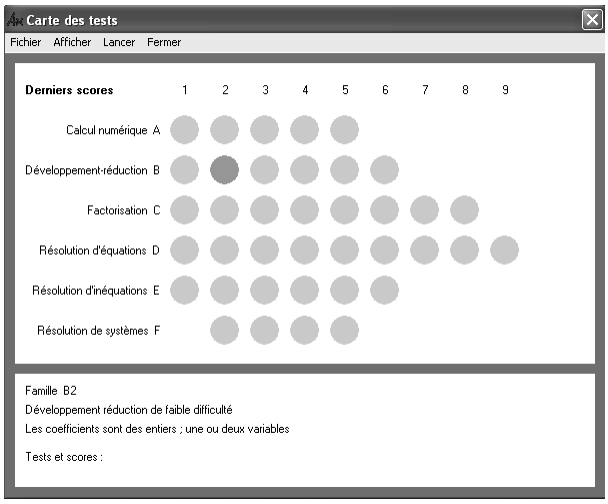


Figure 7. Carte des tests.

4. Expérimentations

Il n'y a pas eu d'expérimentations spécifiques des patrons et de la carte de test pour diverses raisons : ce genre d'expérimentation est plus difficile à analyser car les exercices sont tirés aléatoirement. En général, les didacticiens préfèrent travailler sur des listes d'exercices connues à l'avance et définie précisément par eux. Cependant, depuis Mai 2004, ces patrons et cette carte de tests sont utilises en classe de manière informelle. Ces utilisations se font, par exemple, lors de phases de remédiation. Les retours des enseignants sont très positifs.

Un projet d'expérimentation de masse, orienté statistique, basé sur la carte des tests, est en préparation pour un avenir indéterminé. Cette expérimentation servira aussi à valider le travail entrepris sur la modélisation des conceptions d'élèves en algèbre.

5. Conclusion

Nous avions questionné au début de l'article ce fait naturalisé maintenant : l'EIAH est un domaine qualifié de pluridisciplinaire.

Notre projet est clairement pluridisciplinaire, comme on le voit dans les objectifs annoncés et dans la planification (cf. 3.1 et 3.2). D'une part, on peut se demander quelle signification réelle a cette pluridisciplinarité et, d'autre part, on peut

remarquer qu'une présence pluridisciplinaire ne garantit pas naturellement la réussite d'un projet et ne favorise pas nécessairement des interactions efficaces et pertinentes. Le cadre de travail et la méthodologie ont en fait permis d'établir une véritable coopération entre deux disciplines et des paradigmes autonomes pour atteindre un objectif commun.

Bien sûr les informaticiens auraient pu, comme ils l'avaient fait pour le début de développement du logiciel, travailler de manière isolée, notamment en raison de leurs expériences en mathématique, et produire patrons et tests automatiques. Un produit aurait pu être fourni « clé en main » aux consommateurs didacticiens et enseignants. On peut penser qu'alors, d'une part, le langage aurait été premier et que, d'autre part, les patrons n'auraient pas eu cette richesse qualitative et quantitative vis-à-vis de la structure et de la hiérarchie. Le cadre a en fait permis des boucles d'interaction entre les membres de notre communauté plurielle à partir des avancées de chacun. Même si une première structure langagière a été proposée par les informaticiens, les didacticiens ont commencé à développer l'organisation et le contenu de leurs patrons comme lors d'une analyse a priori en papier crayon pour couvrir l'ensemble des tâches algébriques et pour prendre en compte des variables didactiques et difficultés que rencontrent les élèves. Lors de l'écriture dans le langage informatique, les problèmes rencontrés déplaçaient diverses dimensions, d'une part épistémique avec la modification des patrons, voire de la structure, et, d'autre part, informatique en faisant bouger le langage lui-même. Nous n'aurions pas obtenu le même « produit » si les didacticiens avaient proposé directement une liste de patrons aux informaticiens, qui auraient fait directement une conversion informatique.

Si la recherche en EIAH est pluridisciplinaire, ce n'est pas une simple addition de points de vue disciplinaires sur un même objet. Cette construction interdisciplinaire a permis de prendre mieux appréhender l'articulation de la complexité de l'organisation des savoirs algébrique et son implémentation dans l'EIAH. Nous avons pleinement rencontré et vécu là les phénomènes et problèmes soulevés par l'inévitable transposition informatique [Balacheff 1994].

Un pas a pu être fait vers de la transdisciplinarité [Morin 1997], le didacticien devenant apprenti programmeur avec l'aide des informaticiens, super débogueurs, comme on l'a vécu par moment. Les informaticiens sont rentrés dans les questions didactiques pour comprendre les constructions didactiques et ont pu favoriser l'explicitation, voire la modification de propositions allant de soi parfois pour les didacticiens. Cela nous a conduits à être un peu plus polyglotte et à mieux se parler, mieux dialoguer, mieux comprendre les problèmes de l'autre discipline aussi.

En fait, le travail peut être aussi qualifié de co-disciplinaire [Blanchard 2000] dans la mesure où, malgré des frontières un peu repoussées, chacun est resté prioritairement dans son domaine. Cependant, plus qu'une interaction entre les disciplines présentes, il y a eu une co-construction à propos d'un même projet, même si les objets empiriques pour chacun pouvaient différer.

Ce type de travail co-disciplinaire ouvre la voie à d'autres compléments. En perspectives, côté informatique, la réalisation d'une interface conviviale de description de patrons pour l'enseignant, permettant de tester les patrons imaginés ; côté didactique, la construction de patron ayant un lien avec le curriculum de chaque pays et la mise au point d'un tableau qui indique pour chaque niveau scolaire les familles d'exercices du curriculum.

Bibliographie

[Balacheff 1988] Balacheff N. « Le contrat et la coutume, deux registres des interactions didactiques », In: Actes *du premier colloque franco-allemand de didactique des mathématiques et de l'informatique (Lumigny).* Grenoble : La Pensée sauvage. *1988.*

[Balacheff 1994] Balacheff N. « La transposition informatique, un nouveau problème pour la didactique », In: Artigue M. et al. (eds) *Vingt ans de didactique des mathématiques en France.* (pp.364-370). Grenoble : La Pensée Sauvage éditions.

[Blanchard 2000] BLANCHARD-LAVILLE C., « De la codisciplinarité en sciences de l'éducation », *Revue Française de Pédagogie*, N° 132, 2000.

[Brousseau 1981] Brousseau, G., « Problèmes de didactique des décimaux », *RDM Vol 2*.1, La Pensée Sauvage, Grenoble, 1981

[Brousseau 1990] Brousseau, G., « Le contrat didactique : le milieu », *RDM Vol. 9/3*, La pensée sauvage, Grenoble. 1990

[Brousseau 1997] Brousseau G., « Theory of didactical situations in mathematics », *Kluwer Academic Publishers*, Dordrecht, 1997

[IMS-QTI 2004] IMS Question and Test Interoperability: "Item Implementation Guide" & "Item Information Model", V2.0 Editor Steve Lay, University of Cambridge, http://www.imsglobal.org/question/qti_item_v2p0pd page rédigée en Juin 2004

[Morin 1997] Morin, E., Sur la transdisciplinarité, Guerre et paix entre les sciences, Disciplinarité, inter et transdisciplinarité, Revue du MAUSS, Numéro 10, 1997.

[Nicaud 1990] Nicaud, J.F., Aubertin, C., Nguyen-Xuan, A., Sa, M. and Wach P. APLUSIX: "A learning environment for student acquisition of strategic knowledge in algebra". *Proceedings of PRICAI'90.* Nagoya. Japon, 1990.

[Nicaud 2004] Nicaud, J.F., Bouhineau, D., and Chaachoua H. « Mixing microworld and CAS features in building computer systems that help students learn algebra », in *International Journal of Computers for Mathematical Learning*, Volume 9, Issue 2, Springer-Verlag, 2004.

[Perrin-Riou 2004] Perrin-Riou, B. « Programmation d'exercices OEF (open exercise format) » http://wims.auto.u-psud.fr/wims/ page rédigée en Janvier 2004 pour le projet WIMS.

Les Cahiers Leibniz

Le **Laboratoire Leibniz** est fortement pluridisciplinaire. Son activité scientifique couvre un large domaine qui comprend des thèmes fondamentaux aussi bien en informatique qu'en mathématiques, avec une ouverture sur l'apprentissage machine, la modélisation de systèmes complexes adaptatifs, et les applications aux environnements informatiques pour l'apprentissage humain.

Les **Cahiers Leibniz** ont pour vocation la diffusion de rapports de recherche, de supports de cours, de textes de séminaires ou de projets de publications réalisés par des membres du laboratoire. Ils peuvent accueillir aussi des textes de chercheurs n'appartenant pas au laboratoire Leibniz mais qui travaillent sur des thèmes proches et ne disposent pas de tels supports de publication. Ces chercheurs sont priés de contacter un des membres du comité éditorial ; le comité décidera de l'acceptation du texte proposé.

Le contenu des textes publiés dans les **Cahiers Leibniz** relève de la seule responsabilité de leurs auteurs.

The research at **Laboratoire Leibniz** is multidisciplinary. It covers a large domain of fundamental and applied subjects in informatics an mathematics, with openings to machine learning, the modelisation of adaptive complex systems, and applications to teaching software.

The **Cahiers Leibniz** aim at diffusing research reports, lectures, and texts of conferences or pre-prints of the members of the laboratory. Moreover, the **Cahiers** welcome manuscripts of researchers belonging to other laboratories, working on subjects close to ours, but not disposing of such a medium. These researchers should contact one of the members of the editorial board; the latter will decide whether to accept the proposal.

The responsibility of the contents of the **Cahiers** lies exclusively with the authors.

Comité éditorial

Mirta B. Gordon (responsable), Annie Bessot, Gerd Finke, Humbert Fiorino, Denise Grenier, Philippe Jorrand, Andras Sëbo

Directeur de la publication

Nicolas Balacheff

Réalisation : Jacky Coutin

ISSN : 1298-020X - © laboratoire Leibniz