

ÉCOLE NORMALE SUPÉRIEURE DE CACHAN

Mémoire

pour obtenir le diplôme universitaire

Nouvelles Techniques Cognitives d'Apprentissage

Représentations mentales en informatique

École Doctorale de Sciences Pratiques de l'ENS Cachan

E.D. numéro 285

présentée et soutenue publiquement

par

Pascal Lafourcade

le 28 septembre 2006

rendu le 1er septembre 2006

« L'homme honorable commence par appliquer ce qu'il veut enseigner ;
ensuite il enseigne. »

Confucius (Kong fu Zi 551-479 av J-C).

Table des matières

1	Introduction.	1
2	Représentations mentales.	3
2.1	Définition.	3
2.2	Différentes définitions dans la littérature.	4
2.3	Comment prendre conscience des ces représentations mentales. . .	9
2.4	Pourquoi prendre conscience des ces représentations mentales. . .	9
3	Représentations mentales en informatique.	13
3.1	Programme	13
3.2	Déclaration.	14
3.2.1	Type.	15
3.2.2	Variable.	16
3.2.3	Tableau.	17
3.2.4	Structure.	19
3.3	Instructions élémentaires.	20
3.3.1	Affectation.	20
3.3.2	Conditionnelle.	21
3.3.3	Itération.	22
4	Conclusion.	25

TABLE DES MATIÈRES

Introduction.

«Ce qu'on sait, savoir qu'on le sait ;
ce qu'on ne sait pas, savoir qu'on ne le sait pas :
c'est savoir véritablement.»

Confucius.

Les processus d'apprentissage sont des processus complexes. De nombreux travaux en psychologie cognitive, neuro-psychologie tentent de mieux comprendre ce qui se passe dans le cerveau humain lors de ces processus. Un des thèmes de recherche dans la compréhension de l'être humain consiste à élaborer un modèle de la perception humaine. Les travaux de Piaget, Bruner, Johnson-Laird modélisent les représentations mentales afin de mieux comprendre le fonctionnement des processus d'apprentissage et de développements. Lors d'une phase d'apprentissage, l'enseignant apporte à l'apprenant un nouveau savoir, une nouvelle notion. L'apprenant assimile cette nouvelle notion en la manipulant d'abord par lui-même, ensuite il se forgera sa propre «image» mentale, pour enfin être capable d'utiliser le concept abstrait associé à cette nouvelle notion. Le rôle du pédagogue est donc de faciliter cette phase d'apprentissage en lui proposant des explications claires, des exercices pour manipuler ces nouvelles notions et des représentations éclairantes, ceci en fonction des connaissances antérieures de l'élève. Nous cherchons à définir ce qu'est une représentation mentale. Pour cela nous présentons dans le premier chapitre les principales théories connues associées aux représentations mentales ou modèles mentaux. Nous illustrons l'importance des représentations mentales par quelques exemples. Ensuite nous consacrons le reste de ce document à l'élaboration de représentations mentales afin de faciliter l'apprentissage de notions d'informatiques pour des étudiants de première année à l'université.

1. Introduction.

Chapitre 2

Représentations mentales.

« Dis-moi et j’oublierai,
Montre-moi et je me souviendrai.
Implique-moi et je comprendrai. »

Confucius.

Le terme de « représentation mentale » est un terme utilisé en philosophie et en psychologie. Nous cherchons d’abord à affiner cette définition, puis nous présentons les différents travaux existants sur ce sujet.

2.1 Définition.

Regardons d’abord la définition du Petit Larousse Illustré :

- action de rendre sensible quelque chose au moyen d’une figure, d’un symbole, d’un signe.
- image, figure, symbole, signe qui représente un phénomène, une idée.
- *philosophie* : ce par quoi un objet est représenté à l’esprit (image, concept).
- *psychologie* : perception, image mentale dont le contenu se rapporte à un objet, à une situation, à une scène.

Les représentations, c’est ce que chacun sait, pense, croit, rêve à propos de quelque chose. Une représentation mentale désigne donc ce que nous nous représentons, ce qui forme le contenu concret d’un acte de pensée, autrement dit les structures de connaissances acquises depuis longtemps et stabilisées dans la mémoire.

Exemple 1 *Chacun possède sa propre représentation mentale d’un « vélo » en associant le terme à un objet, à une situation ou à une scène de son histoire personnelle. Ainsi ma représentation mentale d’un vélo correspond à mon vélo de course bleu ciel de mon enfance. Elle peut être différente selon chaque individu.*

2. Représentations mentales.

Pour une notion aussi commune qu'un vélo, nous possédons tous une représentation mentale différente, alors pour des notions plus abstraites et plus complexes, il est fort probable que nous possédions aussi des représentations mentales différentes. Lors d'un processus d'apprentissage, l'enseignant espère qu'une fois la notion enseignée, l'apprenant possède une représentation mentale en accord avec la définition proposée par l'enseignant. Il est donc important pour un enseignant de fournir les meilleures chances à un apprenant afin qu'il se crée ses propres représentations mentales et qu'elles soient le plus juste possible. Pour cela, il est nécessaire que l'enseignant vérifie que les représentations mentales de l'étudiant ne soient pas erronées par ses interprétations personnelles du discours du pédagogue. De plus avoir une bonne représentation mentale d'un problème aide à sa résolution. Nous montrons à travers l'exemple 2 du génie de Carl Freidrich Gauss que posséder une « bonne » représentation mentale peut permettre de résoudre rapidement un problème.

Exemple 2 *L'instituteur de Carl Freidrich Gauss, afin d'obtenir le calme dans sa classe pendant quelques minutes, posa à ses élèves le problème suivant : Combien vaut la somme S des entiers positifs de 1 à 100 ?*

À la surprise de l'instituteur le petit Gauss répondit immédiatement 5050. Intrigué, l'instituteur demanda comment il fit. La représentation mentale du problème de Gauss lui permit de trouver rapidement la solution. Gauss s'aperçut que s'il calculait deux fois la somme des entiers positifs de 1 à 100 le calcul était beaucoup plus simple. Il fit la somme des entiers positifs de 1 à 100 plus la somme des entiers positifs de 100 à 1 sur son ardoise. Ainsi il lui suffit pour calculer deux fois la somme des entiers positifs de 1 à 100 d'additionner 101 fois l'entier 100, comme le montre le calcul suivant qu'il écrivit sur son ardoise :

$$\begin{array}{r} 1+ \quad 2+ \quad \dots + \quad 98+ \quad 99+ \quad 100 = S \\ + 100+ \quad 99+ \quad 98+ \quad \dots + \quad 2+ \quad 1 = S \\ \hline 100+ \quad 100+ \quad 100+ \quad 100+ \quad \dots + \quad 100+ \quad 100 = 2S \end{array}$$

*Pour calculer $2S$ il suffit d'additionner 101 fois 100. Nous obtenons donc $2S = 101 * 100$, ainsi $S = 101 * 50 = 5050$.*

Nous avons défini simplement ce qu'est une représentation mentale et montré qu'elle avait un rôle important dans les processus d'apprentissage et de résolution de problèmes. Nous présentons maintenant les différentes études existantes à propos des représentations mentales.

2.2 Différentes définitions dans la littérature.

Dès le XIXème siècle de nombreux psychologues ont cherché à définir la notion de représentation mentale afin de comprendre sa mise en place et son fonctionnement.

Citons Ausubel et Gagné, le premier s'est intéressé à savoir comment nous sélectionnons une information nouvelle. Il défend la thèse selon laquelle l'apprentissage de manière active permet une meilleure compréhension. Gagné était psychologue dans l'armée de l'air américaine lors de la seconde guerre mondiale. Il a analysé minutieusement les différents éléments d'une tâche afin de favoriser son apprentissage. Il propose alors une hiérarchisation des capacités de plus en plus complexes à développer facilitant ainsi l'acquisition de nouvelles compétences.

L. Vygotsky [Vyg62] introduit la notion de « Zone of Proximal Development ». Cette zone constituée à partir de l'état de développement et de connaissances d'un individu délimite la zone de développement dans laquelle l'apprenant va évoluer en fonction de ces interactions avec son environnement et ainsi franchir une nouvelle étape dans son apprentissage.

J. Piaget [Pia71] transforme la vision du développement de l'enfant. Il distingue quatre étapes dans le développement :

- Sensorimotrice, à partir de 2 ans, l'enfant perçoit le monde extérieur à travers son contact physique avec son environnement.
- Pré-opérationnelle, de 2 à 7 ans, l'enfant est alors capable de se représenter mentalement les objets et les événements du monde qui l'entoure. Pendant cette phase l'attention proverbiale se met en place, l'enfant est égocentrique, ne partage pas son point de vue ni ses objets avec les autres.
- Opérationnelle concrète, de 7 à 11 ans, l'enfant se libère de son égocentrisme et une pensée logique se développe. Progressivement au cours de cette phase l'enfant acquiert la capacité d'effectuer des opérations mentales élevées.
- Opérationnelle, 11 ans et plus l'enfant possède alors les moyens pour résoudre des problèmes abstraits de manière logique.

Les idées de J. Piaget ont largement influencé le mode de penser de ces contemporains, en particulier J. Bruner.

J. Bruner [Bru60] appartient au courant constructiviste. Il développe un modèle basé sur l'approche du développement psychologique de Piaget en élargissant le travail de ce dernier en prenant en compte l'interaction avec l'environnement. Il propose trois modes de représentations au cours du développement d'abord le mode « enactive » ensuite le mode « iconic » et enfin le mode « symbolic ».

- « **enactive** » : l'apprenant construit ses représentations à travers l'action sur les objets du monde, par exemple faire du vélo, nouer son lacet de chaussure.
- « **iconic** » : l'apprentissage se fait à travers des modèles, des images internes propres à l'apprenant en utilisant son mode sensoriel de représentation favori (Visuel, Auditif, Kinesthésique, Olfactif, Gustatif), par exemple l'apprenant peut percevoir sa propre représentation mentale d'un vélo sans pour autant que le vélo ne soit présent.

2. Représentations mentales.

- « **symbolic** » : l'utilisation de termes abstraits de concepts permet à l'apprenant de raisonner et de se construire. Par exemple le mot de la langue française « vélo » est un terme abstrait désignant l'objet vélo. Ainsi en manipulant ce terme abstrait l'apprenant peut écrire son expérience de sa balade en vélo, sans être en contact avec le vélo, ni en avoir une représentation iconique.

Maintenant à titre d'exemple, déclinons un problème informatique dans ces trois modes de représentations. Le problème est le suivant : étant donné une liste de nombres entier, les trier en utilisant l'algorithme de tri par bulles. Nous présentons cet algorithme de tri suivant ces trois modes pour la liste d'entiers suivante 15, 5, 6, 60, 70, 4, 7 qui une fois triée donne 4, 5, 6, 7, 15, 60, 70 :

- « *enactive* » : Comparons deux cartes adjacentes. Si la première est plus grande que la seconde permutons les, recommençons jusqu'à ce que la liste soit triée. En manipulant progressivement la liste nous construisons la liste triée.
- « *iconic* » : Cette construction peut être représentée comme un schéma indiquant les mouvements successifs effectués sur la liste d'entiers comme le montre la figure 2.1.

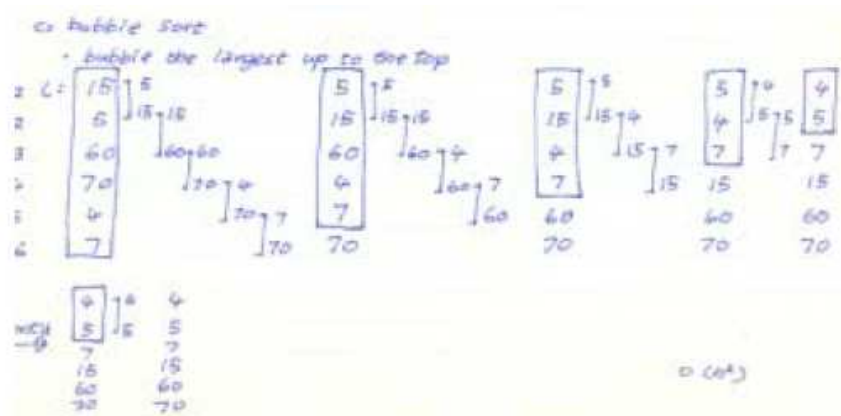


FIG. 2.1 – Représentation iconique de l'algorithme de tri à bulles

- « *symbolic* » : La représentation symbolique est donnée dans ce cas par l'algorithme de tri à bulles écrit dans un pseudo-langage de programmation :

```
for (i=0; i<n-1; i++) {  
  for (j=0; j<n-1-i; j++)  
    if (a[j+1] < a[j]) /* comparons les deux voisins */  
    {  
      /* échangeons a[j] et a[j+1] */  
      tmp = a[j] ;
```

```
    a[j] = a[j+1] ;
    a[j+1] = tmp ;
  }
}
```

Influencé par les travaux de L. Vygotsky, J. Bruner pense que « Learnig is an active process in which learners construct new ideas or concepts based on current/past knowledge. ». Il affirme que tout peut être expliqué à tout le monde à tout âge. Il suffit pour cela de partir de la connaissance de l'apprenant et de l'enrichir progressivement en lui proposant des moyens d'approfondir ces trois modes de représentations en commençant pas le mode « enactive » puis « iconic » et enfin symbolic, et de recommencer à partir de ce nouvel état de la connaissance de l'apprenant telle une spirale. J. Bruner appelle ce cheminement « spiral curriculum ».

J. R. Anderson [And82] avance une théorie dans laquelle les apprenants modélisent un problème en plusieurs sous-problèmes et sous-buts en fonction de leurs représentations mentales propres. Ainsi ce découpage permet en explicitant les représentations de chaque apprenant d'identifier et situer les éventuelles erreurs dans les représentations mentales de l'apprenant

Au début des années quatre-vingt, P. N. Johnson-Laird [JL83] propose une théorie des modèles mentaux basée sur une approche propositionnelle des représentations. Comme dans l'exemple de Gauss, il pense que si une personne utilise un modèle mental approprié pour un problème alors, elle le résoudra plus facilement. Dans son approche, il distingue trois types :

- **modèles mentaux** : analogies structurelles avec le monde.
- **images mentales** : corrélations entre les différents modèles suivant un point de vue particulier.
- **représentations propositionnelles** : informations liées au langage.

Cette approche regroupe sous le nom de modèles mentaux la phase « enactive » et « iconique » de J. Burner, et détaille la phase « symbolique » en images mentales et représentations propositionnelles.

En 1986, Paivio [Pai86] classifie les représentations mentales par analogie aux représentations physiques qui sont soit des images soit des concepts du langage. Prenons en exemple la Tour Eiffel. Dans la figure 2.2 le plus à gauche nous avons la représentation imagée de la Tour Eiffel grâce à une photographie et progressivement l'image se transforme en un dessin, un croquis puis un symbole représentant ce monument pour enfin arriver à la représentation conceptuelle du langage avec la dénomination Tour Eiffel.

Cette classification permet de mieux comprendre le long travail que nous effectuons lors de l'apprentissage de nouvelle notion en partant d'un point de vue concret pour arriver à un concept, un terme abstrait.

2. Représentations mentales.

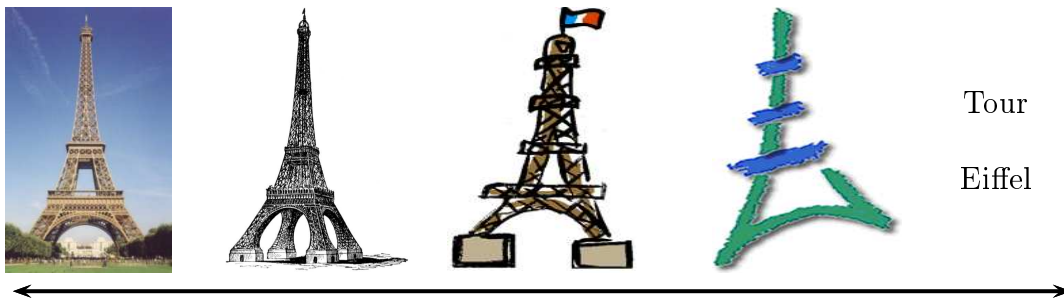


FIG. 2.2 – D'une représentation imagée à une représentation liée au langage

Lors des Journées Apprentissages 2005, A. Finkel suit un point de vue similaire à celui de J. Bruner. Comme le montre la figure 2.3, il distingue trois types de représentations mentales correspondant à des connaissances : les représentations conceptuelles, les représentations imagées et celles qui sont liées à l'action. Il porte plus particulièrement son attention sur les différents modes de représentation mentales suivant les cinq sens (VAKOG), sur la figure seuls sont mentionnés le visuel, l'auditif et le Kinesthésique.

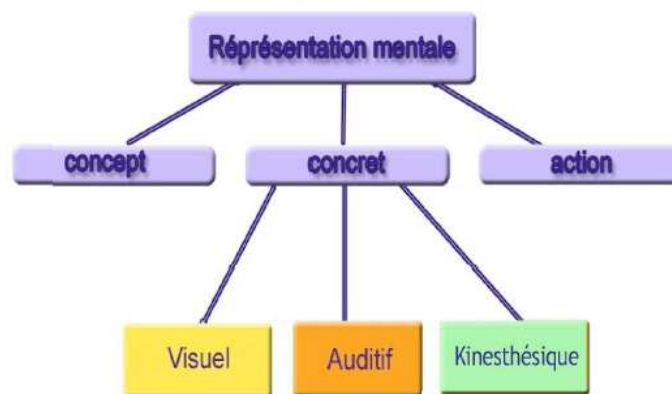


FIG. 2.3 – Définition des représentations mentales selon A. Finkel aux Journées Apprentissages 2005

Le terme de concept pour les représentations mentales selon A. Finkel correspondrait à celui de « symbolic » dans la présentation de J. Bruner, celui d'action au mode « enactive » et celui de concret à « iconique ».

Remarquons que dans ces deux approches les trois modes de représentations mentales peuvent se rattacher aux trois modes usuelles de mémoires à long terme. Le mode « enactive » ou de l'action se rattacherait à la mémoire procédurale qui

correspond à la mémoire d'une action. Le mode « symbolique » ou conceptuel s'apparente à l'encodage en mémoire à long terme grâce au réseau sémantique. Enfin le mode « iconic » ou concret s'identifie à la mémoire dite épisodique qui prend en compte tout les souvenir selon nos cinq sens du vécu d'un passage de notre vie. Par exemple, nous nous souvenons précisément de notre dernière sortie avec des amis. Nous somme capables grâce à nos ressentis de nous remémorer la décoration du restaurant ou le sourire du serveur (V), la musique d'ambiance (A), le confortable fauteuil dans le quel nous étions assis (K), l'odeur de notre charmante voisine (O) et le goût du dessert (G). Cette mémoire épisodique fait appel à tous nos sens (VAKOG) et permet de stocker nos représentations mentales « iconic » selon chacun d'eux.

Nous mentionnons également quelques uns des nombreux auteurs qui se sont penchés sur la question comme Changeux [Cha98], Giordan [GV87], De Vecchi [Vec84], Guerin [Gué88], Gyselink [Gys95], et qui ont aussi étudié le cerveau humain, son fonctionnement et ses représentations mentales.

2.3 Comment prendre conscience des ces représentations mentales.

La technique d'explicitation développée par Vermersch [Ver94] est un outil efficace qui peut être utiliser pour découvrir ces représentations mentales. Cette technique permet à la personne interrogée de décrire, d'évoquer quelles sont ses représentations mentales, grâce à un questionnement respectueux et empathique. C. Petitmengin [Pet01] l'utilise pour analyser l'intuition lors de la résolution de problème et propose de nombreux exemples d'explicitation. Notons comme le montrent J. DÉSAUTELS et M. LAROCHELLE [DL92], le recueil collectif des représentations mentales permet d'expliciter davantage les représentations mentales de chacun ceci grâce à un effet d'entraînement et d'association.

2.4 Pourquoi prendre conscience des ces représentations mentales.

Tout d'abord découvrir comment nous fonctionnons offre une meilleure connaissance de soi et permet une avancée considérable dans le développement personnel de chacun. En analysant notre mode de fonctionnement nous découvrons que nous avons souvent une préférence pour un mode de représentation mentale. Certaines personnes ont besoin de manipuler les choses pour les comprendre, d'autres préfèrent raisonner sur des concepts abstraits ou symboliques, alors que d'autres ont une représentation sensorielle de la notion, de plus parmi cette représentation «

2. Représentations mentales.

iconique » chaque individu a une préférence pour un sens.

Apprendre à identifier notre mode favori de représentation mentale permet de mieux comprendre comment nous fonctionnons. Cela permet aussi d'expliquer pourquoi nous comprenons mieux quand les explications sont données dans notre mode de représentation favori, alors que certaines fois l'explication dans un autre mode de représentation ne nous paraît pas forcément claire. Ainsi si l'enseignant utilise notre mode préféré de représentation lors de son explication, cela facilite notre compréhension. Ainsi tout en continuant à accroître notre mode de représentation mentale favori, nous pouvons chercher à développer les autres modes pour augmenter notre compréhension et enrichir notre encodage en mémoire.

Enfin, prendre conscience de ces représentations mentales aide les élèves à comprendre une notion pour eux incompréhensible. En explicitant la représentation mentale de l'élève le professeur découvre son mode de fonctionnement. Si nécessaire il peut alors modifier la représentation mentale de l'apprenant pour faciliter sa compréhension, et ainsi résoudre son problème de compréhension. Pour illustrer notre propos, nous présentons dans l'exemple 3 le célèbre problème « des trains et de la mouche de Von Neumann ». Nous montrons qu'en fonction de la représentation mentale que nous avons de ce problème, la solution apparaît plus ou moins facilement suivant les individus.

Exemple 3 *L'énoncé du problème de trains et de la mouche de Von Neumann est le suivant :*

Deux trains séparés de deux cents kilomètres roulent sur la même voie l'un vers l'autre. Ils avancent à la vitesse de cent kilomètres à l'heure. Une brave mouche part de l'avant de l'un et vole à la vitesse de soixante-quinze kilomètres à l'heure jusqu'à ce qu'elle rencontre le second train. A ce moment là, elle fait demi tour, et repart vers le premier, et ainsi de suite jusqu'à ce que les trains tuent en se rencontrant.



Quelle distance totale a-t-elle parcouru pendant ce vol ?

Von Neumann répondit à la question immédiatement et quand on lui demanda comment il avait fait il dit : « Simple, I summed the series ! » (J'ai simplement sommé les séries). Si nous adoptons le mode de pensée de Von Neumann notre représentation mentale du problème va être de considérer le parcours de la mouche et de voir que nous pouvons l'écrire comme la somme des trajets entre les deux trains, il faut alors l'écrire de manière mathématique et effectuer la somme de la série obtenue.

Il est possible d'adopter un autre point de vue, en remarquant que nous connaissons la vitesse de vol de la mouche, il ne reste plus qu'à trouver le temps de vol jusqu'à la collision pour en déduire la distance parcourue par la mouche. Les trains

2.4. Pourquoi prendre conscience des ces représentations mentales.

roulant à cent kilomètres à l'heure et étant séparés de deux cents kilomètres, ils vont mettre exactement une heure pour se rencontrer. La mouche, en une heure, aura donc parcouru soixante-quinze kilomètres, ce que répondit Von Neumann.

À travers cet exemple nous voyons bien que chacun possède son mode de représentation, ce qui permet de trouver parfois des solutions différentes à un même problème.

Tout dispositif d'apprentissage peut être considéré comme un essai pour enrichir les représentations mentales de quelqu'un. Comme le rappelle Wicker [Wic78] la représentation mentale n'est pas seulement une image, une photographie, une illustration fidèle de la réalité. Chacun construit sa propre représentation en fonction de son histoire et de sa propre perception du monde. Notons que la présentation d'une illustration et l'interaction sociale sont deux moyens permettant de faciliter la génération de représentations mentales, comme le montrent les études de Denis [Den76] et Pressley [Pre77].

Dans la suite, nous présentons des idées à propos de quelques notions informatiques afin de susciter la création de représentations mentales chez les élèves de première année d'université. Dans ce document, nous adoptons la vision de J. Bruner et celle reprise par A. Finkel. Ainsi pour faciliter leur compréhension de ces nouvelles notions abordées. Nous essayons de donner à chaque notion une approche « enactive » puis « iconique » et enfin « symbolique ».

2. *Représentations mentales.*

Représentations mentales en informatique.

«Une image vaut mille mots.»

Confucius.

Nous proposons des représentation mentales en informatique relatives au programme de première année de licence d'informatique, pour une initiation à la programmation en utilisant le langage C. Nous aborderons différentes notions en tenant compte de la spécificité du langage considéré. Nous nous efforçons de proposer pour chaque notion une approche active (« enactive ») puis une représentation imagée (« iconic ») et ensuite une approche symbolique (« symbolic ») afin de faciliter la compréhension d'un novice et lui permettre ainsi de construire sa propre représentation mentale. Étant donné que mon mode « iconique » favori est le mode visuelle, et que la matière abordée dans ce mémoire est l'informatique, je proposerai principalement des représentations iconiques visuelles. Nous commençons par les notions de déclarations de variables du langage impératif. Ensuite nous abordons les instructions élémentaires des langages de programmation impératif, illustré au travers du langage C [SR88]. Nous présentons enfin des représentations mentales permettant de comprendre l'affectation d'une variable, la notion de condition et d'itération.

3.1 Programme

Nous introduisons de manière générale ce qu'est un programme informatique suivant nos trois modes de représentation.

« **Enactive** » Nous proposons de se représenter concrètement un programme comme une recette de cuisine. Pour faire un gâteau, nous utilisons des ingrédients,

3. Représentations mentales en informatique.

des ustensiles et nous effectuons certaines actions à partir de tous ces éléments pour fabriquer le gâteau. Un programme utilise lui aussi des ustensiles ce sont des variables, des opérateurs, des fonctions. Il utilise également des ingrédients qui correspondent à des ressources mémoire, de calcul du processeur. L'ensemble des instructions décrites par la recette de cuisine pour faire le gâteau correspondent exactement aux instructions d'un programme informatique qui permettrait de faire un gâteau.

« **Iconique** » Nous pouvons voir également un programme comme un « jeu de l'oie » comme le montre la figure 3.1, où chaque joueur correspond à un processus exécutant le programme. Le passage d'une case à une autre du jeu de l'oie consiste à effectuer une instruction du programme. L'ensemble du jeu de l'oie représente le programme en entier, arriver à la fin du jeu correspond à terminer le programme.



FIG. 3.1 – Représentation iconique de la notion de programme

Nous approfondissons cette illustration de programme sous forme de jeu de l'oie dans la section où nous aborderons les instructions d'affectation, de comparaison et d'itération.

« **Symbolique** » De manière plus abstraite, un programme est une suite d'instructions, exécutées par un processus afin de réaliser une tâche plus ou moins complexe.

Nous décrivons maintenant les différentes instructions qu'un programme peut effectuer lors de son exécution.

3.2 Déclaration.

Nous commençons par les notions de types, de variables, de tableaux et de structures. Ces notions sont essentielles pour la programmation et permettent au programme de manipuler différentes données.

3.2.1 Type.

Nous considérons le langage de programmation C qui est un langage typé. Nous expliquons la notion de type en informatique suivant les trois modes de représentations choisis.

« **Enactive** » Pour manipuler concrètement cette nouvelle notion, nous utilisons des billes colorées. Imaginez que vous possédez 5 billes rouges et 3 blanches. Puis vous gagnez 1 bille rouge. Combien avez-vous de billes de chaque couleur ?

Nous supposons implicitement qu'il y a deux types différents de billes, les rouges et les blanches. Nous pouvons les distinguer facilement grâce à leurs caractéristiques. Ainsi à la fin, vous possédez 6 billes rouges et 3 blanches. Manipuler des objets et être capable de les distinguer suivant un critère permet de les classer dans des catégories, les billes rouges ou billes blanches. Ces catégories sont appelées en informatique des types. Nous abordons maintenant cette notion suivant une représentation « iconique ».

« **Iconique** » Dans la figure 3.2, nous illustrons la notion de type en montrant que l'addition d'une pomme plus une poire est impossible alors qu'une pomme plus une pomme cela fait deux pommes.

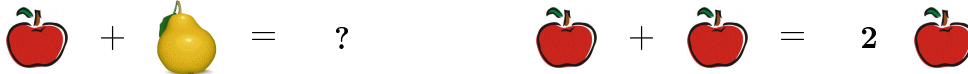


FIG. 3.2 – Représentation iconique de la notion de type

Grâce à cette représentation iconique, nous voyons bien que les objets avec des caractéristiques différentes sont de types différents, notre connaissance nous permet de distinguer le type de chaque objet.

« **Symbolique** » Après avoir donner ces représentations « enactives » et « iconiques », nous donnons la définition symbolique d'un type telle qu'elle est directement proposée en général aux étudiants :

Le type précise essentiellement les propriétés physiques d'un objet :

- La taille en mémoire qui lui est réservée
- La méthode de représentation

Dans le langage C, il existe de nombreux types pour représenter les entiers (integer), les décimaux (float), les lettres (char), les chaînes de caractères (string). Pour les entiers par exemple, il existe différents types (long int, short int) en fonction de la taille maximale d'un entier que nous désirons, ceci correspond à la place mémoire utilisée par l'entier.

3. Représentations mentales en informatique.

Nous poussons un peu plus loin la comparaison entre les différentes catégories de fruits et les différents types associés aux nombres. Nous pouvons répondre à la question suivante : Combien il y a de pommes si nous avons une pomme golden et une pomme granny ? la réponse est bien entendu deux. Si le type associé aux pommes granny correspond aux entiers et le type associé aux pommes golden correspond aux décimaux, nous pouvons bien entendu les ajouter, le type associé au résultat est déterminé par des règles implicites ou explicite. Dans le cas du langage C, la somme d'un entier et d'un décimal donnera un entier implicitement, mais il est possible d'obtenir un décimal en précisant explicitement le type désiré (cast).

3.2.2 Variable.

Maintenant qu'un type définit les caractéristiques d'un objet, nous manipulons ces objets à l'aide de variables.

« **Enactive** » En reprenant l'idée des billes, nous considérons qu'elles sont toutes de la même couleur. Nous vous offrons un sac pour ranger vos billes. Vous rangez vos billes dans votre sac, il contient donc toutes vos billes. Le sac correspond à une variable de type entier car il contient le nombre de billes en votre possession. Si maintenant vous souhaitez ranger des pommes de terre vous prendrez un sac plus grand. En fonction du type des objets contenus dans votre sac vous n'utiliserez pas le même sac, ainsi la taille de la variable « sac » dépend du type d'objets que vous allez y ranger.

« **Iconique** » Dans la figure 3.3, une variable de type entier correspondant à notre sac de billes est représentée comme une boîte portant le nom « Sac ». Cette variable contient le nombre de billes que nous possédons.

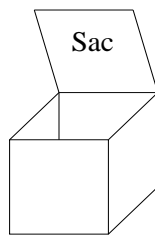


FIG. 3.3 – Représentation iconique de la notion de variable pour un entier de type int.

Remarquons que la taille et la forme de la boîte sont définies par le type de données que nous y rangeons. Si nous considérons des entiers long (long int) à la place des entiers, notre boîte correspondrait à celle de la figure 3.4 qui est

beaucoup plus grande que celle de la figure 3.3, car elle va contenir des objets plus grands.

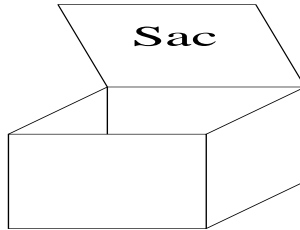


FIG. 3.4 – Représentation iconique de la notion de variable pour un entier de type long int.

« **Symbolique** » Une variable est un objet contenant une donnée d'un certain type. Par exemple une variable de nom « Sac » de type entier s'écrit comme suit en C :

```
int sac ;
```

Si nous savons que le sac contient au début 10 billes nous écrivons :

```
int sac=10 ;
```

Nous reviendrons plus précisément sur ceci dans la section concernant l'affectation.

3.2.3 Tableau.

Nous souhaitons maintenant ranger toutes nos billes dans un même sac et les classer par couleur.

« **Enactive** » Nous utilisons un sac dans lequel nous avons plusieurs poches pour ranger les billes de chaque couleur. Supposons dans un premier temps que nous avons trois couleurs. Notre variable s'appelle toujours « Sac » et permet de ranger dans trois compartiments les billes de différentes couleurs.

« **Iconique** » Nous reprenons l'idée de boîte et créons trois compartiments à l'intérieur, pouvant chacun contenir une valeur de type entier, correspondant à chacune des couleurs considérées.

La taille des compartiments est fixée par le type associé à la boîte ainsi tous les compartiments sont de la même taille et peuvent accueillir les mêmes objets. De plus le nombre de compartiments est aussi fixé lors de la création de la boîte et il n'est pas possible de le changer lors de l'exécution du programme. Dans le langage C, il existe des moyens de créer des tableaux « dynamiques » et de changer leur taille au cours de l'exécution du programme. Par souci de clarté, ces

3. Représentations mentales en informatique.

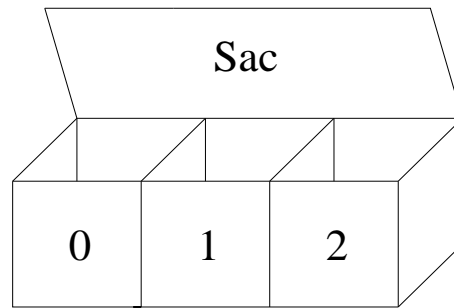


FIG. 3.5 – Représentation iconique de la notion de tableau

notions ne seront pas abordées dans ce document s'adressant à des débutants. Il nous semble pertinent de signaler l'existence de ces tableaux « dynamiques » au lecteur, car savoir s'il est possible de manipuler des tableaux dynamiques est une question naturelle qu'il est légitime de se poser.

Maintenant, nous présentons, uniquement de manière « iconique », les tableaux à deux dimensions dans la figure 3.6, ceci pour ne pas alourdir le document.

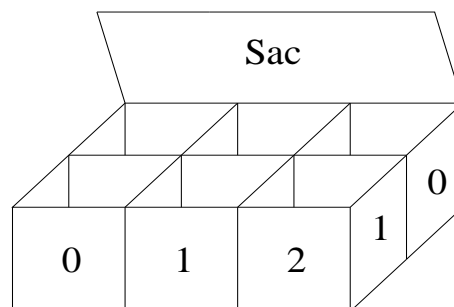


FIG. 3.6 – Représentation iconique de la notion de tableau à deux dimensions

Cette fois-ci, il y a deux rangées de compartiments numérotés horizontalement et verticalement. Pour parler d'un compartiment nous devons donner ses coordonnées exactes, par exemple le compartiment le plus en haut à droite correspond à la rangée zéro et à la colonne deux.

« **Symbolique** » Un tableau est une collection de n objets de même type rangés séquentiellement dans un certain ordre, constituant ainsi un objet plus grand regroupant tous cette collection d'objets.

3.2.4 Structure.

Nous voulons ranger dans un même objet différents éléments comme dans un tableau mais chaque objet est de type différent. Pour faire cela, il existe la notion de structure décrite maintenant.

« **Enactive** » La manipulation de dates nécessite d'utiliser des entiers de 0 à 31 pour les jours, des chaînes de caractères pour les mois et des entiers pour les années. Ces trois données sont de types différents mais les dates elles sont toutes de même type : le type date. Nous pouvons ajouter et soustraire deux dates et nous obtenons une nouvelle date. Nous manipulons naturellement donc cette structure « date » en isolant chacune des composantes d'une date et en effectuant mentalement des opérations pour obtenir une nouvelle date.

« **Iconique** » Nous prenons l'exemple de la date et dans la figure 3.7 nous proposons une représentation montrant la différence entre un tableau et une structure. Dans une structure contrairement à un tableau, nous définissons nous même le type de chacune des boîtes et leur nom. Nous choisissons le nom du premier compartiment « Jour », il contient un entier. Le second sert à ranger une chaîne de caractères dans le compartiment appelé « Mois » et le dernier emplacement, nommé « An », contient un entier pour l'année associée à une date.

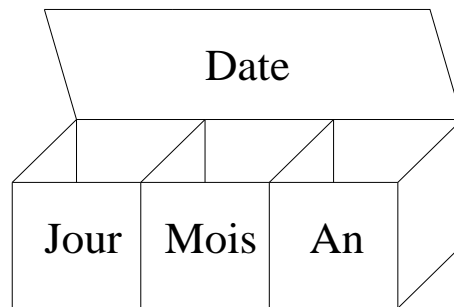


FIG. 3.7 – Représentation iconique de la notion de structure

La différence principale entre une structure et un tableau est que le tableau contient uniquement des compartiments de même type.

« **Symbolique** » Une structure est une collection d'objets, éventuellement de types différents, regroupés sous un seul nom pour simplifier le traitement lors de l'exécution du programme.

3.3 Instructions élémentaires.

Nous présentons maintenant les différentes instructions élémentaires d'un programme.

3.3.1 Affectation.

Nous venons de définir la notion de type et de variables, nous présentons maintenant comment modifier une variable. Il est également possible de modifier le contenu des éléments d'une structure et d'un tableau, le principe étant le même (seule la syntaxe change) nous nous restreignons ici au cas d'une variable.

« **Enactive** » Reprenons notre représentation d'une variable avec le sac de billes. Affecter une valeur à la variable « sac » correspond à remplir le sac avec des billes. Si nous mettons 5 billes dans le sac vide nous affectons alors la valeur 5 à la variable « sac ». Si lors de l'exécution du programme, nous ajoutons une autre bille dans le sac, le nombre de billes augmente et nous avons alors 6 billes. Nous avons affecté une nouvelle valeur à la variable « sac ».

« **Iconique** » L'affectation d'une valeur à une variable sous la représentation de boîte correspond à changer le contenu de la boîte, comme le montre la figure 3.8. Partant de la boîte « sac » vide au début de l'exécution du programme, nous ajoutons 5 billes dans le sac. La variable « sac » contient donc le nombre 5. Ensuite, nous ajoutons une bille de plus. Pour cela nous devons lire la valeur de la boîte et ajouter « un » à cette valeur, ceci est possible si les types sont les mêmes, et affecter cette nouvelle valeur calculée dans la boîte. La boîte contient donc l'entier 6.

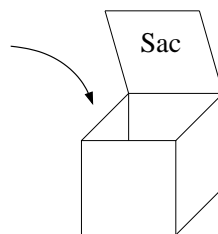


FIG. 3.8 – Représentation iconique de la notion d'affectation d'une variable

« **Symbolique** » De manière symbolique la première affectation se fera grâce à l'égalité en C. `sac = 5 ;`

Cela signifie que la variable `sac` déclarée comme entier plutôt dans le programme par l'instruction `int sac ;` reçoit la valeur entière 5. Ensuite nous notons la

seconde affectation permettant d'ajouter une bille dans notre variable « sac » par :

```
sac = sac + 1 ;
```

La variable *sac* reçoit la somme du contenu de la variable *sac* et de l'entier 1. Cette instruction n'est pas à considérer comme une équation mathématique à résoudre mais comme une nouvelle affectation de la variable « sac ». Remarquons en C, l'égalité entre deux valeurs se note par un double symbole égal « == ». Cette différence a été introduite pour justement distinguer l'affectation du symbole d'égalité de comparaison..

3.3.2 Conditionnelle.

Nous analysons comment introduire l'instruction conditionnelle « if ».

« **Enactive** » Cette instruction correspond à une notion de choix, de prise de décision que nous connaissons tous, car nous prenons en permanence des décisions. Notre porte-monnaie, représenté par une variable « bourse », contient des euros. Nous souhaitons acheter une paire de chaussures qui coûte 12 euros. Pour cela nous exécutons une instruction mentalement disant que si j'ai la somme d'argent disponible alors je pourrai acheter ces chaussures.

La prise de décision peut être plus compliquée : pour acheter les chaussures il faut que j'ai assez d'argent dans mon porte-monnaie et que les chaussures soient de la bonne couleur, par exemple noires. Si les deux conditions sont satisfaites, alors j'achète les chaussures, sinon je continue mon chemin.

« **Iconique** » Revenons à notre représentation de programme par un jeu de l'oie introduit au début du chapitre. Dans la figure 3.9, nous représentons une case correspondant à une instruction conditionnelle. Elle est dénotée par « if ». Maintenant, si la condition écrite dans la case « if » est vraie alors nous effectuons la suite d'instructions symbolisée par la case « then » sinon nous passons par la séquence d'instructions contenues dans la case « else ».

Remarquons que les cases « then » et « else » peuvent être n'importe quelle séquence d'instructions valides du programme. Par définition de la structure de la case « if » la suite d'instruction de la case « then » ne peut être vide, par contre il se peut que la case « else » soit vide ainsi nous continuons notre programme en passant immédiatement à la case suivante. Cela revient à n'effectuer les instructions de la case « then » que si la condition est vraie.

« **Symbolique** » Une instruction conditionnelle s'écrit sous la forme suivante en C :

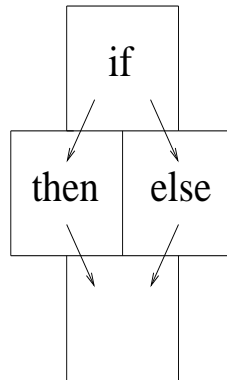


FIG. 3.9 – Représentation iconique de la notion d’instruction conditionnelle

```
if ( <expression> ) <I1>  
[else <I2>]
```

Si l’évaluation de l’expression dans la condition est vraie alors nous effectuons la suite d’instructions I_1 sinon nous effectuons la suite d’instructions I_2 si elle existe, car elle peut être vide.

3.3.3 Itération.

Nous proposons une approche en trois points pour l’instruction itérative « while », et illustrons plus particulièrement la différence entre l’instruction « do ... while » et l’instruction « while » dans la représentation « iconique ».

« **Enactive** » Pour introduire cette notion, nous nous rappelons le temps des punitions à l’école quand nous avons à copier 50 fois une définition. Lorsque nous copions ces 50 définitions nous entrons dans un processus itératif souvent appelé boucle en informatique : tant que je n’ai pas écrit 50 fois la définition je continue.

Nous manipulons naturellement cette notion quotidiennement, par exemple lors de nos courses, nous restons dans le magasin tant que nous n’avons pas trouvé tous les produits inscrits sur notre liste de course.

Une question naturelle se pose donc, que se passe-t-il si un produit n’est pas présent dans le magasin. D’après la définition du processus itératif nous restons dans le magasin jusqu’à ce que tous les produits soient trouvés, ce processus peut dans ce cas ne jamais s’arrêter, ceci est une « boucle infinie ». Nous sommes entrés dans un processus infini, comme dans la mythologie grecque, où Zeus condamna Sisyphe à pousser éternellement un rocher au sommet d’une montagne sans jamais y parvenir : à peine Sisyphe est-il arrivé à proximité de son but que le rocher roule vers le bas, et tout est à recommencer.

Dans la vie quotidienne nous avons la possibilité de modifier la condition de la boucle, en décidant que nous pouvons nous passer du produit manquant après avoir cherché un certain temps. Ceci nous permet de sortir du processus itératif infini et de sortir du magasin.

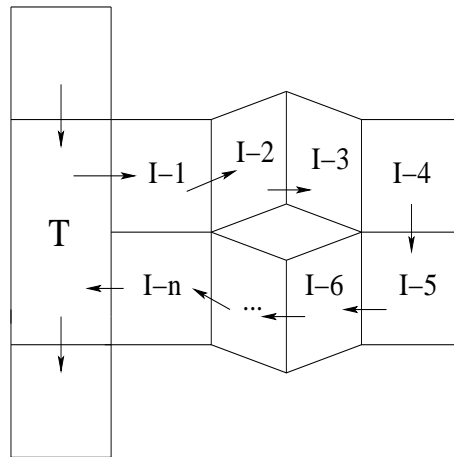


FIG. 3.10 – Représentation iconique de la notion d'itération

« **Iconique** » Reprenons l'illustration par un jeu de l'oie, et décrivons maintenant l'idée de boucle en regardant comment se présente une itération. Nous arrivons sur la case « T » qui est un test comme dans le cas de l'instruction conditionnelle, si le test est satisfait alors nous continuons la suite du programme et ne rentrons pas dans la boucle d'instruction. Sinon nous commençons la séquence d'instructions circulaires présentes dans la figure 3.10. Une fois toutes ces instructions effectuées nous repassons par la case de test « T », et faisons le même raisonnement.

Le « do ... while » est représenté par la figure 3.11.

Avec le « do .. while » la séquence d'instructions débutant par le marqueur « do » est d'abord effectuée et ensuite nous vérifions la condition. Si elle est vraie nous continuons la suite du programme sinon nous refaisons les instructions décrites à partir du marqueur « do ».

« **Symbolique** » Dans le langage C, l'itération s'exprime par l'instruction « while » de la manière suivante :

```
while <expression> { I-1 ; ... , I-n ; }
```

Tant que l'évaluation de l'expression est vraie j'exécute la séquence d'instructions I-1, ..., I-n. Et ensuite je réévalue l'expression, et ce jusqu'à ce que l'expression soit fausse.

3. Représentations mentales en informatique.

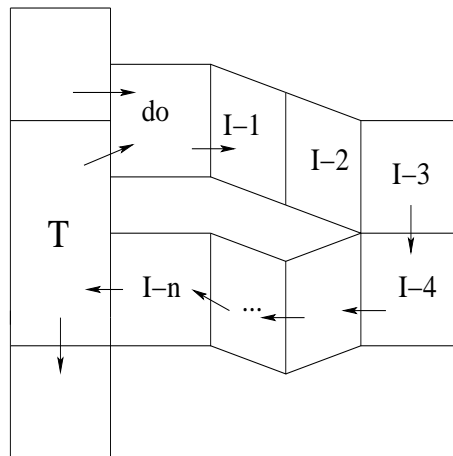


FIG. 3.11 – Représentation iconique de la notion d'itération

Notons que si aucune instruction ne change l'évaluation de l'expression nous restons indéfiniment dans la boucle à exécuter la suite d'instructions.

L'exemple de la punition est représenté de manière humoristique dans la figure 3.12, où l'élève au lieu d'écrire les lignes demandées, écrit le programme C associé en utilisant une boucle « for », qui correspond à une réécriture simplifiée d'une boucle « while ».

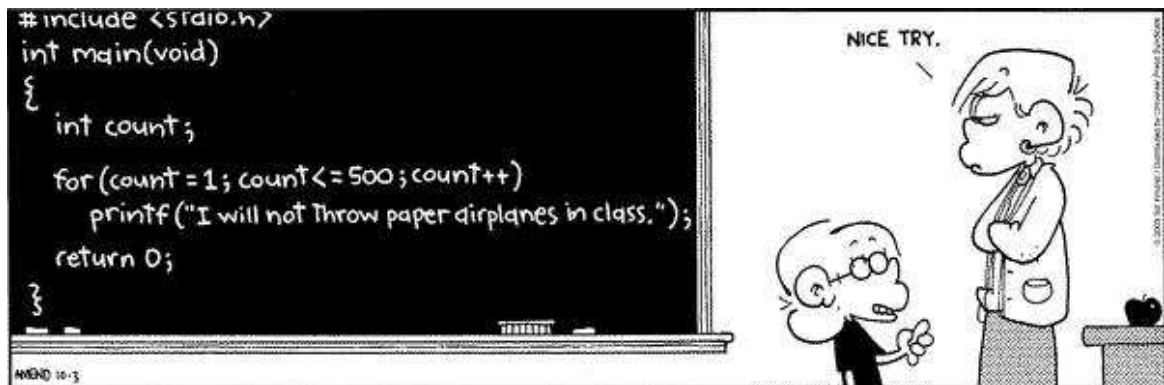


FIG. 3.12 – Illustration d'itération.

Chapitre 4

Conclusion.

« Le sage a honte de ses défauts,
mais n'a pas honte de s'en corriger. »

Confucius.

À travers ce document, nous avons voulu illustrer en appliquant à l'informatique l'importance des représentations mentales dans un processus d'apprentissage. Elles permettent à chacun comme nous l'avons vu de se construire sa propre vision du monde extérieur et de faciliter l'acquisition de nouvelles notions..

Dans un premier temps nous avons donné un bref aperçu des différents travaux existants sur les représentations mentales dans la littérature. Ces travaux montrent que plusieurs modèles différents existent suivant les auteurs. Nous avons choisi de prendre le point de vue de J. Bruner qui distingue trois types de représentation. Cette vision des représentations mentales a également été employée à de nombreuses reprises par A. Finkel lors des journées apprentissages afin de faire découvrir aux étudiants et aux enseignants l'importance des représentations mentales dans l'apprentissage.

Ensuite nous avons proposé des représentations mentales suivant ces trois modes de représentation pour certaines notions informatiques afin de faciliter leur apprentissage pour des élèves de première année de Licence. Ces propositions de représentations mentales ne sont bien sûr pas uniques. Elles constituent une première approche pour parler de notions informatiques aux étudiants en essayant d'appliquer les découvertes en sciences cognitives afin de faciliter l'apprentissage de chacun. Nous pensons qu'elle permettent à l'apprenant d'aborder une nouvelle notion dans un premier temps en la manipulant de manière concrète à partir de sa propre expérience. Puis elle offre une représentation iconique de la notion abordée afin que l'élève se construise ses propres représentations et puisse dans un dernier temps manipuler la représentation symbolique de cette nouvelle notion. Cette notion symbolique et abstraite est souvent la seule proposée par un

4. Conclusion.

enseignant du supérieur à ces étudiants, nous imaginons alors quel peuvent être les difficultés de compréhension rencontrées par les étudiants qui n'arrivent pas à construire par eux mêmes leurs propres représentations mentales via ce mode de représentations.

Grâce à ce travail, nous espérons que le lecteur aura pris conscience de l'importance de la représentation mentale dans le processus d'apprentissage. Nous pensons que les trois modes de représentations sont trois étapes cruciales pour l'acquisition de nouvelles connaissances. Chaque enseignant doit avoir conscience que chaque élève possède son propre mode de représentation. Le pédagogue doit être capable d'offrir à ses étudiants pour une nouvelle notion une approche suivant ces trois modes de représentations afin de faciliter l'apprentissage. Nous avons abordé que quelques notions élémentaires d'informatique de première année d'université, nous envisageons d'étendre cette étude pour l'ensemble des notions du programme de licence d'informatique afin d'offrir une base de représentations mentales que chaque pédagogue puisse ensuite s'approprier, améliorer et intégrer dans son enseignement.

Bibliographie

- [And82] J. R. Anderson. Acquisition of cognitive skills. *Psychological Review*, 89(4) :369–406, 1982.
- [Bru60] J. Bruner. *The Process of Education*. Harvard University Press, 1960.
- [Cha98] J.-P. Changeux. *L'homme neuronal*. Hachette-Pluriel, 1998.
- [Den76] M. Denis. *Les images mentales*. Presses Universitaires de France, 1976.
- [DL92] J. DÉSAUTELS and M. LAROCHELLE. *Autour de l'idée de science - itinéraires cognitifs d'étudiants*. De Boeck Université, Bruxelles, 1992.
- [Gué88] P. Guérin. *Importance des représentations mentales initiales dans un processus d'apprentissage et expression libre*. Les documents du Nouvel Educateurs n 196, 1988.
- [GV87] A. Giordan and G. D. Vecchi. *Les origines du savoir*. Delachaux et Niestlé, 1987.
- [Gys95] V. Gyselinck. *Les modèles mentaux dans la compréhension de textes : le rôle des illustrations*. PhD thesis, Université René Descartes, Paris V, 1995.
- [JL83] P. N. Johnson-Laird. *Mental Models*. Cambridge, Mass. : Harvard University Press, 1983.
- [Pai86] A. Paivio. *Mental Representations : A Dual-coding Approach*. Oxford University Press, 1986.
- [Pet01] C. Petitmengin. *L'expérience intuitive*. L'Harmattan, 2001.
- [Pia71] J. Piaget. *Biology and Knowledge*. Chicago University Press, 1971.
- [Pre77] M. Pressley. Imagery and children's learning : Putting the picture in developmental perspective. *Review of Education Research*, 4(47) :585–622, 1977.
- [SR88] A. Sayah and J.-M. Rigaud. *Programmation en langage C*. Cepadues, 1988.

BIBLIOGRAPHIE

- [Vec84] G. D. Vecchi. *Modalité de prise en compte des représentations enfantines en biologie à l'école élémentaire et leur intérêt dans la formation des maîtres*. PhD thesis, Université de Paris VII, 1984.
- [Ver94] P. Vermersch. *L'entretien d'explicitation*. Presses Universitaires de France, 1994.
- [Vyg62] L. Vygotsky. *Thought and Language*. London : Jaohn Wiley, 1962.
- [Wic78] F. W. Wicker. Our picture of mental imagery : Prospects for research and development. *Educational Communication and Technology Journal*, 26(1) :15–24, 1978.