TECHNIQUES ET MÉTHODES PÉDAGOGIQUES

Sous la direction de
Jean-Luc GURTNER et Jean RETSCHITZKI

LOGO ET APPRENTISSAGES

DELACHAUX ET NIESTLÉ

CHAPITRE 4

LOGO: QU'EST-CE QUI SE DEVELOPPE ?

Patrick Mendelsohn

LOGO est à l'origine d'une littérature abondante dans les domaines de la recherche en éducation (apport de LOGO à l'enseignement des disciplines) et de la psychologie cognitive (influences de LOGO sur le fonctionnement cognitif, transfert d'apprentissage). Il n'est pas dans mon propos de faire ici une revue de questions exhaustive de cette littérature (on peut se reporter à plusieurs articles de synthèse sur ce sujet: Mendelsohn, 1988; Mayer, 1988) mais plutôt d'examiner une question qui me paraît pourtant essentielle dans le contexte de ce symposium consacré aux rapports entre LOGO et les apprentissages. A ma connaissance ce problème est peu traité par les psychologues et pourrait se formuler de la façon suivante: quel rapport y a-t-il entre l'apprentissage de LOGO et l'étude du développement cognitif?

LOGO EST-IL UN BON MODELE POUR L'ETUDE DU DEVELOPPEMENT COGNITIF ?

Il faut rappeler que le point de départ de LOGO est, si l'on en croit un des textes fondateurs de cette entreprise (Papert, 1980), une tentative pour «implémenter» un modèle du développement cognitif d'inspiration constructiviste. L'idée originale est de proposer aux enfants un «micro-monde» de commandes informatiques qui se laisse organiser comme les schèmes opératoires «piagétiens». Dans cet univers constructiviste, toute connaissance est conçue comme une composition plus ou moins complexe d'unités élémentaires de savoirs et de savoir-faire. Les plus élémentaires d'entre elles faisant l'objet d'une «programmation» cachée plus contraignante. C'est le sujet actif, par son activité assimilatrice, qui est alors le seul bâtisseur de ses structures cognitives par le jeu des adaptations progressives de ses schèmes aux buts qu'il se fixe.

Il semble qu'une étude génétique des acquisitions liées à LOGO n'ait jamais été réellement tentée par les psychologues alors qu'elle se justifie pleinement d'un point de vue théorique. L'objectif de cet exposé est de voir s'il est possible de réparer cette lacune; manière comme une autre de revenir

aux sources de LOGO qui était considéré, à l'époque de sa conception, comme un des enfants né du mariage entre l'intelligence artificielle et le constructivisme génétique.

Si l'on s'en tient à cette position de principe, que peut-on dire de LOGO comme «métaphore» du développement cognitif? Pea et Kurland (1984a) avaient déjà constaté que très peu de travaux étaient consacrés à cet aspect du problème. Leur analyse, qui consiste à distinguer des niveaux d'expertise dans la pratique de la programmation, est d'une inspiration plus «résolution de problème» que «développementaliste». Je n'ai trouvé, jusqu'à cette époque, qu'une seule entreprise de cette nature (Mc Keough, 1985) dont la conception théorique est nettement d'inspiration néo-piagétienne. Cet auteur se propose de considérer la programmation graphique "LOGO» comme la transformation d'une représentation analogique en une représentation propositionnelle avec quantification. Son modèle l'amène à distinguer, suivant un principe analogue à celui de Case (1985), quatre niveaux pour décrire l'évolution des compétences des enfants dans des tâches de reproduction de tracés graphiques:

- un niveau 0 (3-5 ans) caractérisé par une stratégie polaire: les ordres utilisés sont du type AVANCE un peu/beaucoup, Tourne à droite ou à gauche, sans maîtrise des aspects dimensionnels de la figure.
- un niveau 1 (5-7 ans) dans lequel l'enfant peut coordonner 2 dimensions polaires (avance beaucoup, tourne à droite). La quantification ne porte, à ce stade, que sur une seule dimension.
- un niveau 2 (7-9 ans) dit des «coordinations bifocales» avec la possibilité pour l'enfant de construire des quantifications sur 2 dimensions (côtés et angles par exemple).
- un niveau 3 (9-11 ans) où l'enfant peut réaliser des coordinations élaborées par compensation de 2 variables différentes en covariation. Ce dernier niveau conduit l'enfant à manipuler des objets abstraits (comme le polygone) qui deviennent ainsi, comme dans le modèle de Case, des unités élémentaires pour le stade de niveau supérieur.

A ma connaissance ces travaux n'ont pas eu de suite. Les raisons de cette absence d'intérêt pour les aspects développementaux de la programmation graphique tiennent peut-être pour une part à l'épuisement du paradigme des stades en psychologie cognitive mais elles sont aussi dues aux difficultés introduites par l'intrication des différents domaines de compétence en jeu dans toute activité de programmation. Une chose est certaine, les activités LOGO ne sont pas un simple jeu d'assemblage de «cubes» élémentaires de connaissances. Très vite, l'apprenti-programmeur est confronté à des techniques de programmation sophistiquées qui n'ont plus rien à voir avec la simple juxtaposition de primitives ou la réunion sous un seul terme d'un ensemble de commandes (modularité simple). L'itération, la modularité complexe ou surtout la récursivité posent des problèmes d'expertise qui sont très éloignés des analyses traditionnelles de la psychologie du développement.

Est-ce une raison suffisante pour renoncer à toute tentative d'utiliser LOGO comme un modèle du développement cognitif? Si j'ai pu le penser à un certain moment, je suis maintenant convaincu que nous nous sommes

découragés un peu vite, attirés par d'autres chimères (en particulier le problème du transfert dans les apprentissages). C'est donc à un travail d'analyse et de clarification que je voudrais maintenant convier le lecteur en partant, pour appuyer ma démonstration, d'un exemple de progression LOGO sur la programmation du tracé d'une même figure simple. Cet exemple me sera utile dans un deuxième temps pour proposer un cadre d'analyse des différentes composantes développementales impliquées dans les activités de programmation de ce type. Je terminerai cet exposé par un aperçu des implications didactiques de ce modèle.

UN EXEMPLE: LA CONSTRUCTION D'UN QUADRILLAGE SUR L'ECRAN LOGO

Considérons dans un premier temps, un exemple de progression très simple qui illustre une démarche de type «constructiviste» et que j'ai exposé il y a déjà quelque temps à l'occasion de la «Second European Conference on Developmental Psychology» (Mendelsohn, 1986b). L'enfant doit représenter sur l'écran un quadrillage de 5x5 carrés (cf figure 4.1) en n'utilisant qu'un ensemble limité de commandes LOGO (AV n; RE n; TD n; TG n; LC; BC) et une grammaire simplifiée: séquentialité, répétition, modularité.

Une analyse en niveaux, relativement au type d'objets que contrôle le sujet, nous a conduit à proposer la progression suivante :

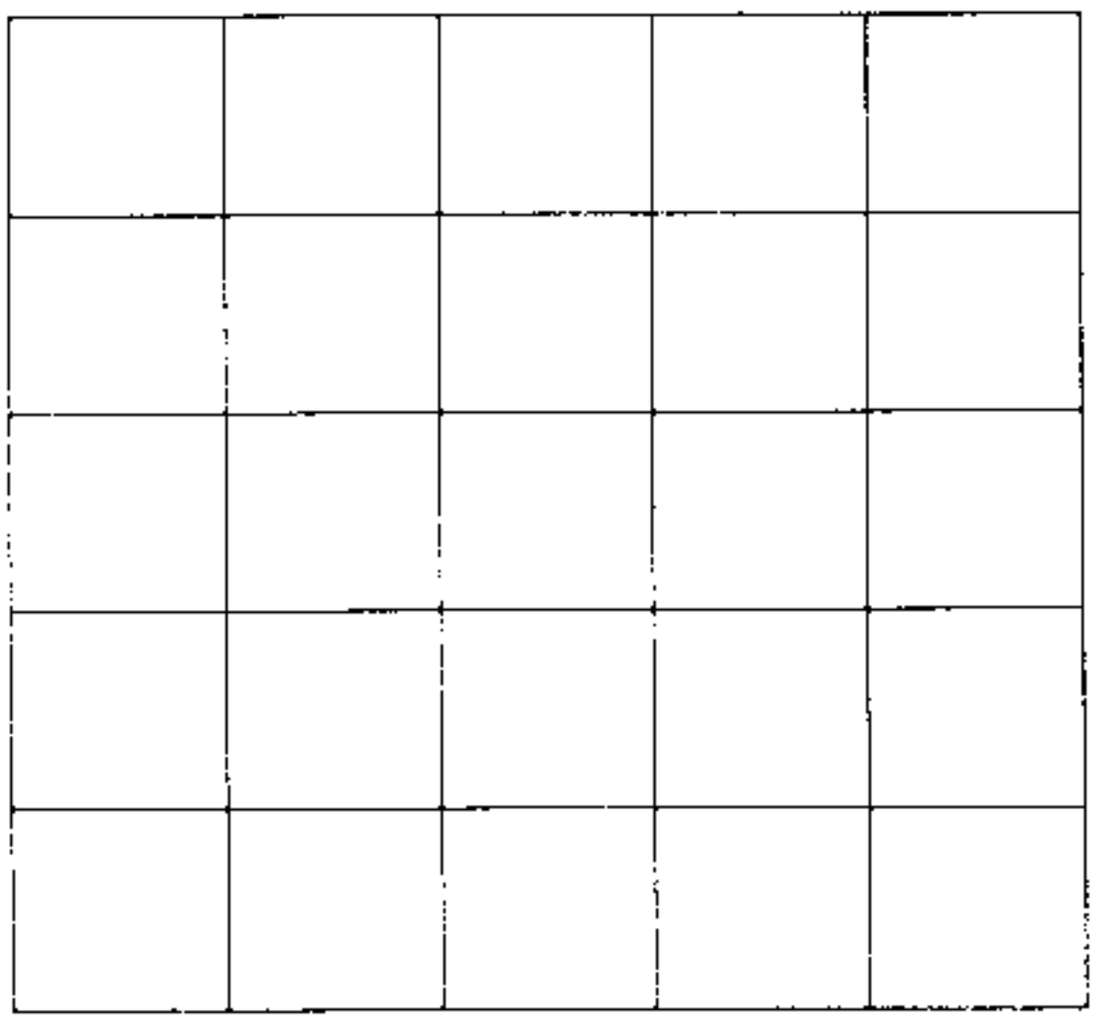

Figure 4.1
Figure à produire en LOGO

- Niveau 0: Pilotage de formes élémentaires. L'enfant de ce niveau considère le quadrillage comme un ensemble de déplacements, son objectif se limite à promener la tortue sur l'écran en suivant pas à pas les effets qu'il obtient. Les procédures de ce niveau sont du type:

```
AV 100 TD 90 AV 20 TD 90 AV 100 TD 90 AV 20.....
```

- Niveau 1: Pilotage de séquence (modularité simple). Ici, le sujet considère la figure complexe comme un ensemble de lignes et de déplacements. Chaque LIGNE est elle-même un déplacement de la tortue et une réinitialisation. L'objectif poursuivi comprend un sous-but: construire la séquence LIGNE et un objectif principal: décrire la figure comme un ensemble non organisé de lignes. Le programme ressemblera plus ou moins à:

```
POUR LIGNE
AV 100 RE 100
FIN

POUR QUADRILLAGE
LIGNE TD 90 AV 20 TG 90 LIGNE TD 90....
FIN
```

- Niveau 2: Coordination de séquences (modularité élaborée). A ce niveau, l'apprenti-programmeur peut se représenter la figure comme un ensemble de lignes horizontales et verticales; cette description est déjà un ensemble coordonné de séquences et chaque séquence, un ensemble de primitives. Il y a donc à ce niveau deux sous-buts coordonnés: 1) la construction d'une LIGNE et d'un DEPLACEMENT et 2) la construction d'une GRILLE formée de LIGNES parallèles. Ces deux sous-buts permettent enfin la construction du QUADRILLAGE.

```
POUR LIGNE.DEP
AV 100 RE 100 TD 90 AV 20 TG 90
FIN

POUR GRILLE
REPETE 5 [LIGNE.DEP]
FIN

POUR QUADRILLAGE
GRILLE LC AV 100 TD 90 RE 120 BC GRILLE
FIN
```

- Niveau 3: Généralisation (paramétrisation). Ce n'est qu'à ce stade que le quadrillage proposé est perçu comme un cas particulier d'une structure plus complexe (l'ensemble des quadrillages NxN de longueur variable). Cette description permet d'extraire les invariants spatiaux caractéristiques

d'une telle figure et les paramètres associés à ce qui peut varier dans un tel ensemble. Les procédures de ce niveau ressemblent à la décomposition suivante:

```
POUR LIGNE :PAS
AV :PAS RE :PAS
FIN

POUR DEPLACEMENT :PAS
TD 90 AV :PAS TG 90
FIN

POUR GRILLE :N
REPETE :N [LIGNE :N * :COTE DEPLACEMENT :COTE]
FIN

POUR QUADRILLAGE :NOMBRE :COTE
GRILLE :NOMBRE LC AV :NOMBRE * :COTE
TD 90 RE (:NOMBRE + 1) * :COTE
BC GRILLE :NOMBRE
FIN
```

Cette procédure paramétrisée permet maintenant de tracer n'importe quel quadrillage NxN.

En résumé, gardons à l'esprit, pour la suite de notre propos, que l'enfant construit ses procédures à partir 1) d'une représentation spatiale plus ou moins élaborée du modèle (en déplacements, lignes ou ensembles de lignes), 2) d'une représentation propositionnelle de ces mêmes objets (dénomination en référence à des objets, des actions), 3) d'un traitement logique associé à l'organisation temporelle des commandes ou des modules (modularité, itération...), cette organisation temporelle traduit une coordination de plus en plus poussée des éléments de la figure et enfin 4) du traitement des aspects dimensionnels de la figure (variables, paramètres...).

UN MODELE POUR ABORDER LES ACTIVITES LOGO D'UN POINT DE VUE DEVELOPPEMENTAL

Ce court exemple nous amène à distinguer au moins quatre progressions parallèles qui semblent impliquées à des degrés divers dans le développement des compétences de l'apprenti-programmeur: les deux premières reprennent la proposition de Mc Keough (1985), qui considère LOGO graphique comme une transposition d'une représentation analogique en une représentation propositionnelle. Pour la compléter, nous proposons d'intégrer à cette analyse le fait que cette transposition peut porter, d'une part sur

les relations logiques (par exemple les relations invariantes entre les éléments de la figure) et, d'autre part, sur les dimensions des éléments de la figure (par exemple la variation des valeurs affectées aux côtés du quadrillage). Cet apport nous amène à proposer une représentation schématique des compétences qui se développent chez l'apprenti-programmeur sous la forme de cercles concentriques s'élargissant sur au moins quatre dimensions complémentaires: propositionnelle, analogique, dimensionnelle et logique (figure 4.2).

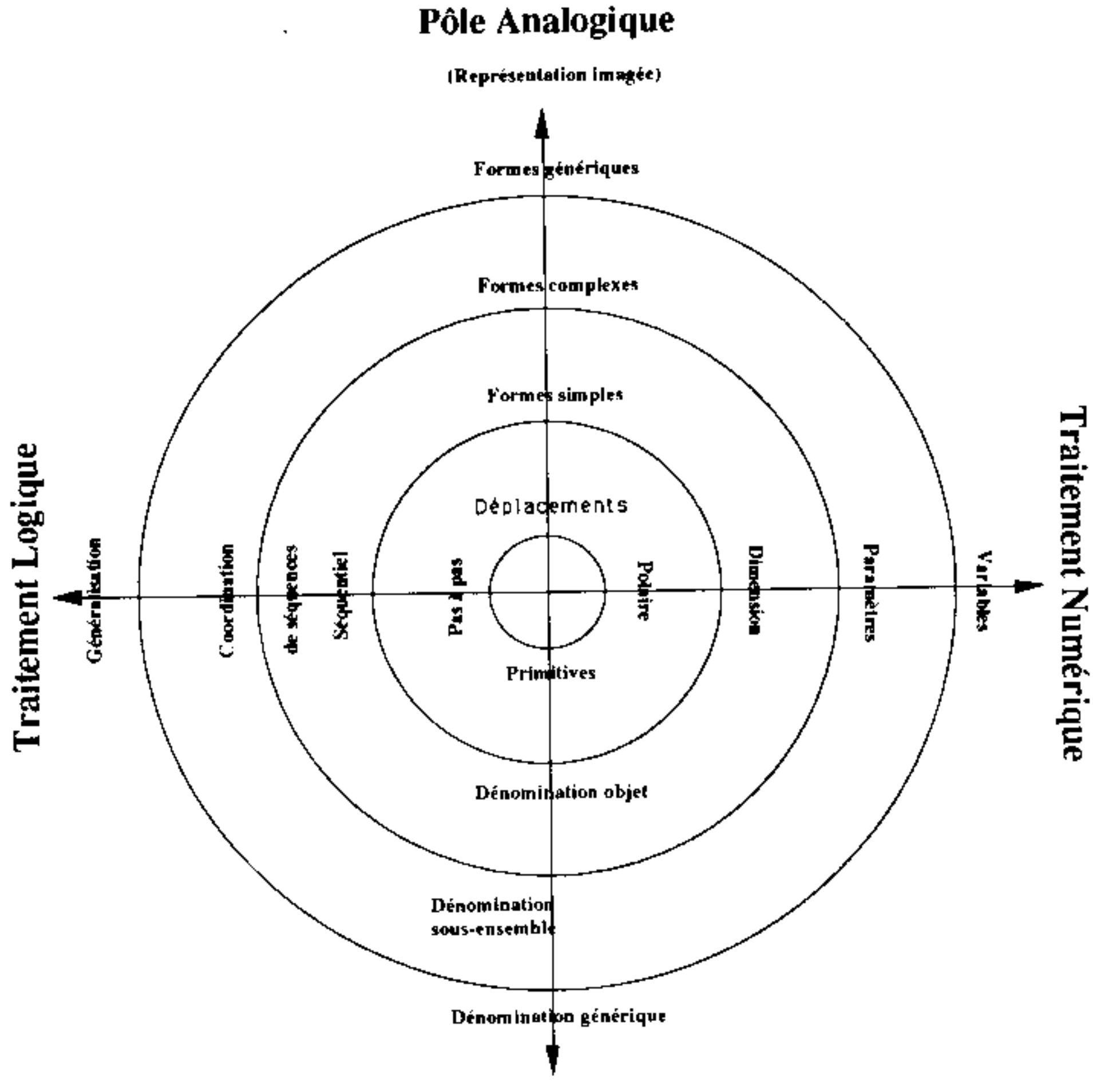

Figure 4.2 Un modèle pour décrire les activités LOGO d'un point de vue développemental

L'analyse que nous pouvons faire maintenant de notre exemple sur la construction du quadrillage fait ressortir toute la richesse - et aussi la complexité - d'une analyse développementale des acquisitions liées à LOGO:

- **Un pôle propositionnel:** LOGO est avant tout un «langage» c'est-à-dire un système symbolique qui renvoie au problème de la dénomination et de la signification des commandes. Le vocabulaire conceptuel associé aux objets de complexité croissante fait référence au mécanisme bien connu des psychologues depuis Miller (1956): le «chunking». En effet, un terme générique comme LIGNE dans notre exemple se distingue de la commande AVANCE, dans la mesure où il comprend implicitement un retour à l'état initial. De même, GRILLE est une dénomination qui comprend tout un ensemble d'opérations élémentaires intégrées sous un seul appel. Ce développement conduit de l'ensemble des primitives de départ «pré-programmées» à la conceptualisation d'objets génériques (polygones, quadrillage), en passant par la dénomination d'objets simples puis de sous-ensembles. L'aspect purement «conceptuel» et linguistique de ce pôle ne doit pas être sous-estimé dans la performance globale des enfants. Il est très fréquent de voir ces derniers s'appuyer exclusivement sur la signification sémantique d'une commande plus que sur ses effets procéduraux (voir par exemple les difficultés liées à la commande STOP chez les plus grands).

- **Un pôle analogique:** les commandes graphiques ont un effet sur l'écran qui évoque des formes de complexité variable qu'il s'agit de «reconnaître»: lignes parallèles, carrés, rectangles emboîtés... Cette composante de l'activité de programmation ne se limite pas à la seule pratique du LOGO graphique: la gestion du curseur avec LOGOWRITER pose des problèmes analogues. Le sujet doit être capable de distinguer progressivement des objets graphiques, des déplacements, puis de situer les premiers dans un système de référence (en général cartésien). Ce développement conduit à la capacité de «voir» des formes génériques complexes comme des ensembles coordonnés d'opérations spatiales élémentaires. La forme graphique étant soit une instanciation de relations logiques invariantes (carré sur sa base ou sur une «pointe»), soit une représentation de dimensions (petit carré ou grand carré).

- **Un pôle «traitement logique»:** qui prend en charge la construction progressive de l'organisation temporelle des commandes (séquences, modules, répétition, appel récursif...). Une figure complexe se traduit, dans un langage de programmation comme LOGO, par un réseau de relations qu'il faut expliciter à travers le déroulement du programme, juxtapositions, coordinations... Cette dimension du développement des activités de programmation a fait l'objet de nombreuses recherches en didactique, en particulier à travers l'étude de la notion de récursivité. Elle désigne les modes de contrôle que se donne l'enfant pour organiser le déroulement temporel des procédures. Cette organisation est, initialement, complètement sous le contrôle du sujet - pilotage pas à pas - elle n'est que progressivement confiée à la machine. Le mode séquentiel qui consiste à différer l'exécution des commandes, en entrant plusieurs instructions sur une même ligne, semble être une étape intermédiaire entre le pilotage à vue et la modularité. La répétition, qui comprend une composante modulaire - la liste des instructions entre [ ]

doit être considérée comme un module - est déjà une forme de composition de séquences, caractéristique du troisième niveau (coordination de séquences); comme l'itération et la récursivité qui sont les exemples les plus accomplis de ce niveau. Ils permettent de confier à une structure de contrôle interne au système le bon déroulement des actions à réaliser.

- Enfin, **un pôle «dimensionnel»:** qui décrit les étapes de la quantification des dimensions de la figure par l'utilisation des paramètres et des variables. Cette dernière composante fait référence aux dimensions (longueur des déplacements, valeur des angles, nombre d'itérations...) qui sont manipulées dans toute activité de programmation graphique. Il est maintenant bien connu que l'enfant commence par travailler avec des dimensions «polaires» (AV un peu, beaucoup; TOURNE à droite, à gauche) avant de pouvoir porter son attention sur les systèmes de mesure et les paramètres associés à ces dimensions. Une autre caractéristique de ce pôle concerne le nombre de dimensions que l'enfant peut traiter simultanément (longueur, angles, nombre de côtés...). Cette contrainte renvoie à la problématique des travaux néo-piagétiens du développement (Case, 1985) qui insistent beaucoup sur l'importance des effets «charge mentale» produits par le nombre d'éléments qui doit être pris en compte simultanément par la mémoire de travail.

A partir de ce modèle, il est possible de reconsidérer la notion de stade de développement dans une activité complexe:

- chaque pôle constitue une dimension autonome dans le développement et les compétences associées à ce pôle suivent une logique interne propre au système considéré.
- le synchronisme dans l'apparition de certaines compétences pourrait tout simplement être dû à la nécessité pour l'enfant de maîtriser simultanément plusieurs composantes d'une tâche en raison des liens de contingence qui relient les différentes dimensions du travail que l'enfant doit réaliser.
- pour obtenir un niveau d'expertise donné, sur un des axes, il est nécessaire d'obtenir une compétence de même niveau sur les autres axes; ceci explique pourquoi il est difficile de faire progresser les enfants au-delà d'un certain seuil sur une des dimensions décrites sans lui imposer un travail analogue sur les autres pôles.

En résumé, il semble que LOGO peut livrer des informations riches comme moyen d'étude du développement cognitif. On a peut-être prématurément pensé qu'avant tout l'apprentissage de LOGO est affaire d'expertise. La grande majorité des auteurs - dont je fais partie - ont aussi trop vite délaissé les aspects développementaux pour ne s'intéresser qu'aux effets de la programmation sur ce même développement comme si la programmation n'en faisait pas partie. Peut être que les langages informatiques sont un formidable moyen pour étudier les effets d'interactions entre toutes les composantes du développement cognitif! Nous avons vu que l'environnement LOGO se prête parfaitement à la manipulation des paramètres qui constituent la complexité d'une opération mentale (espace, temps, nombre,

logique, sémantique...). LOGO retrouve ainsi sa première vocation qui est d'être un micro-monde, pas seulement pour l'enfant qui apprend, mais aussi pour le psychologue qui s'intéresse au développement.

CONSEQUENCES DE CETTE ANALYSE SUR LES OBJECTIFS PEDAGOGIQUES QU'IL EST POSSIBLE D'ASSIGNER A LOGO DANS UN CURRICULUM

Une alternative à la proposition «apprendre à programmer ou programmer pour apprendre» que j'ai moi-même souvent défendue (Mendelsohn, Green, & Brna, 1990) consiste à proposer un projet que semble confirmer l'évolution actuelle des langages informatiques destinés à l'enseignement. Pourquoi ne pas considérer les environnements LOGO actuellement disponibles comme des systèmes qui favorisent l'approche de logiciels informatiques plus élaborés associés aux langages fondamentaux conventionnels (écrire, calculer, dessiner, organiser? Cette voie est clairement celle qu'ont choisie les concepteurs de LOGOWRITER. Au micro-monde des «tortues» programmables, ils ont associé le micro-monde du curseur et de la manipulation des mots, permettant par là même d'explorer l'univers des traitements de textes et de l'édition. C'est aussi celle de LEGO-LOGO qui relie le monde des actions réelles à celui de leur description propositionnelle ou encore de LOGO BASE pour la gestion et l'organisation des informations.

Cette proposition consiste à considérer l'apprentissage des langages de programmation et de leur environnement comme **une approche intégrée** de logiciels d'aide à la conception. Si l'on se réfère aux systèmes de représentation les plus conventionnels et non aux connaissances thématiques, les apprentissages scolaires portent essentiellement sur la lecture et l'écriture, sur les différentes formes de représentation graphique, sur le système numérique et les opérations arithmétiques, et enfin sur la logique et le raisonnement. Ces systèmes de représentation sont présents dans toute activité symbolique et ils sont plus ou moins radicalement transformés par le support informatique (figure 4.3).

Les éditeurs de textes: L'écriture est une activité présente dans tout travail d'édition de programmes. Un éditeur n'est rien d'autre qu'un traitement de texte, un espace bien délimité avec ses fonctionnalités propres où l'on peut manipuler des mots, les déplacer ou encore les corriger. L'évolution récente des environnements de programmation (Logowriter, Smalltalk...) rend tout à fait compatible l'apprentissage des commandes d'un éditeur avec celles des traitements de texte professionnels. Le transfert de compétence est ici tout à fait réel et mérite d'autant plus d'être souligné que la possibilité de programmer des macro-actions sur ces logiciels est une activité similaire au micro-monde du curseur dans Logowriter.

Les éditeurs graphiques: Les langages de programmation destinés à l'enseignement sont souvent réduits, par l'usage qui en est fait, à certains

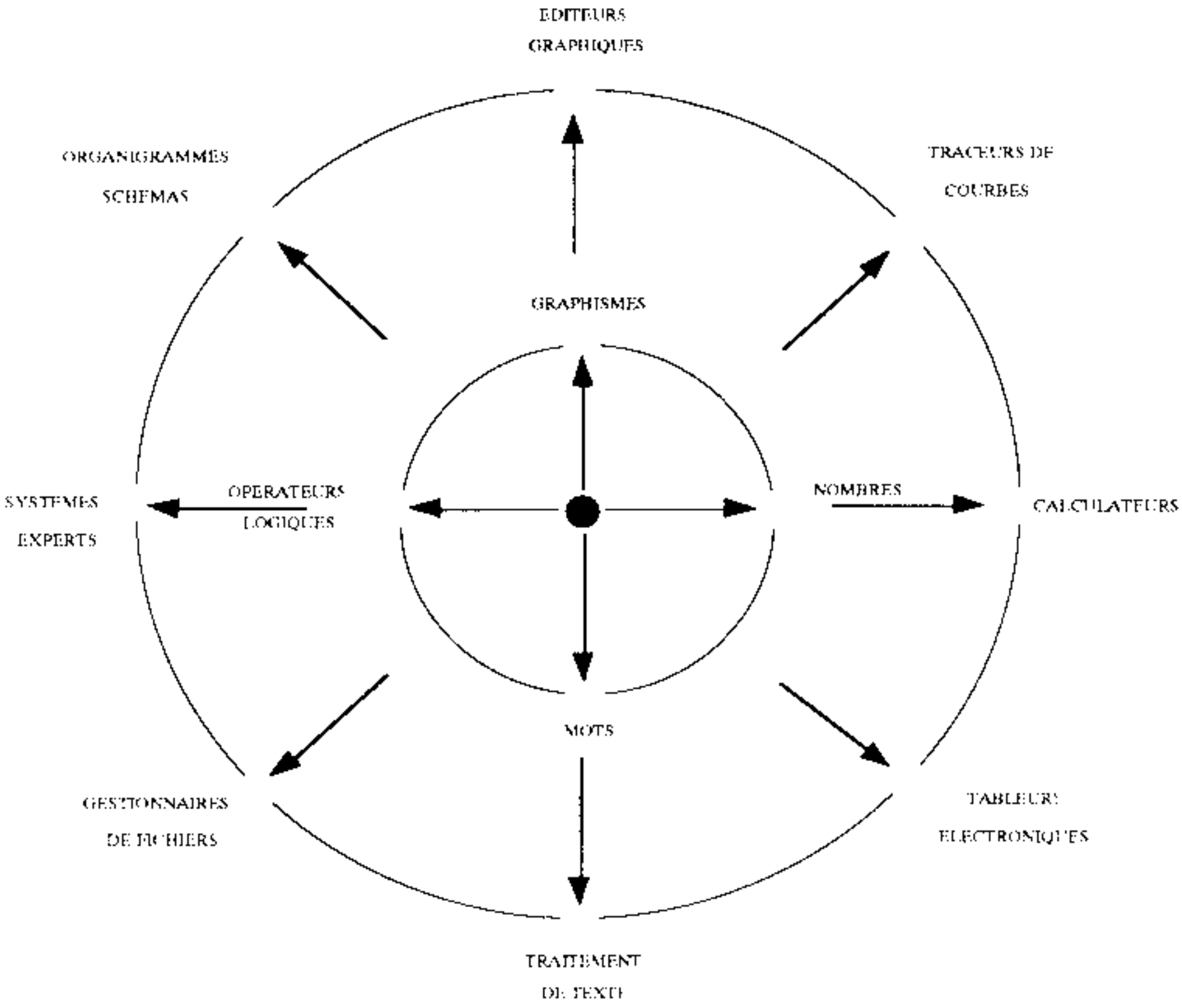

Figure 4.3
Les systèmes de représentation et de traitement et leurs transpositions sur supports informatisés

de leurs sous-ensembles graphiques. Le micro-monde de la tortue LOGO, par exemple, permet d'aborder facilement tous les formalismes graphiques: point, ligne, orientation, déplacement, formes paramétrisables, systèmes de coordonnées. Ces formalismes sont présents, sous des représentations variées, dans tous les éditeurs graphiques proposés sur le marché qu'ils soient associés à la représentation de grandeurs numériques, à la construction de schéma ou encore à l'illustration de texte. Là encore l'évolution des pratiques pédagogiques observées et des environnements de programmation existants tend à rapprocher l'utilisation des langages classiques des logiciels spécialisés (bibliothèques de formes graphiques élémentaires paramétrisables, fonctions de remplissage, fonctions de réduction et d'agrandissement).

Le calcul numérique: Tout programme nécessite la manipulation de dimensions, de fonctions numériques et de variables. Leur expression dans

les langages de programmation classiques diffère peu de celles qu'on retrouve sur une calculette programmable, un tableur ou une base de données. La programmation d'un tableur peut ainsi devenir un objectif réaliste pour l'enseignant qui utilise LOGO et proposer une représentation intermédiaire à mi-chemin entre les langages procéduraux classiques et les langages déclaratifs. En effet, chaque cellule du tableau contient soit une procédure, soit une valeur, sans que soit précisé l'ordre d'exécution du programme.

Le raisonnement logique: Dernière composante, le traitement des relations logiques est aussi une donnée première de la programmation. Que ce soit la logique proprement dite des prédicats (opérateurs logiques), ou l'organisation du flux temporel du programme (raisonnement conditionnel, emboîtement des procédures), on retrouve ce traitement comme ossature de tous les programmes. On peut raisonnablement considérer la programmation structurée comme une sensibilisation à la manipulation des structures arborescentes qui sont elles-mêmes au coeur de tous les systèmes de base de données, comme des systèmes experts.

Ces différents champs conceptuels peuvent être considérés comme des modules autonomes mais sont en général associés deux à deux pour former des unités plus larges spécialisées dans un domaine particulier de l'aide à la conception. Ainsi, l'association du traitement de texte et du calculateur programmable va générer le tableur qui permet de manipuler, sur la même structure de données, du texte et des calculs numériques. L'association du traitement de texte et de la logique (à travers la structure arborescente) engendre le gestionnaire de fichier qui n'est, tout compte fait, qu'une manière d'introduire des relations logiques sur des fragments de texte. La représentation graphique de données numériques (histogrammes, camemberts, courbes de fonctions...) associe les dimensions spatiales (angles, longueurs ou surfaces) à des valeurs numériques de référence; l'organigramme et le schéma, par contre, assimilent les relations logiques avec des relations spatiales (proximité, emboîtements, partition...).

L'univers des pratiques pédagogiques et des langages de programmation pour l'école est en pleine mutation. Plus proche des systèmes professionnels qui ont fait leur preuve, plus convivial dans son environnement, ce dernier est en train de devenir ce qu'il aurait toujours dû être : une préfiguration intégrée de logiciels professionnels. La programmation éducative conserve ainsi son seul objectif légitime, proposer une sensibilisation aux différents concepts informatiques dans le but de préparer les enfants d'aujourd'hui à la culture informatique de demain.