L'ENFANT ET LES ACTIVITES DE PROGRAMMATION

Patrick MENDELSOHN,
Attaché de Recherche au C.N.R.S.
Laboratoire de Psychologie Expérimentale de GRENOBLE

L'ordinateur a commencé son entrée dans l'éducation sous la forme de l'enseignement programmé par ordinateur. Son objet était alors de permettre l'acquisition de connaissances nouvelles en suivant les principes pédagogiques de l'enseignement programmé, développés par SKINNER. Depuis quelques années, se dessinent progressivement les contours de ce qu'on appelle maintenant couramment l'Enseignement Assisté par Ordinateur (E.A.O.), qui comprend toute une série de techniques dont les objectifs sont orientés vers l'application et l'organisation des connaissances déjà acquises par ailleurs.

Parmi ces différentes approches, on peut citer :

– L'utilisation de programmes construits pour l'enseignement (on les appelle des didacticiels).

– La simulation de modèles pour les sciences expérimentales.

– L'apprentissage et la pratique de la programmation (Basic, Pascal, LSE . . .)

– L'acquisition de techniques à usage scolaire ou professionnel (calcul, documentation, traitement de texte . . .)

– L'approche LOGO, qui se distingue de la pratique de la programmation dans le sens où LOGO est à la fois un langage de programmation, une théorie de l'apprentissage des connaissances et un dispositif matériel.

Ces pratiques supposent chacune des projets pédagogiques différents :

– Individualisation de l'enseignement.

– Acquisition de méthodes d'analyse et d'organisation du travail.

– Acquisition d'outils permettant de résoudre des problèmes.

– Exploration par les enfants d'un domaine particulier de connaissances (géométrie, musique, mathématique . . .)

Toutes ces techniques reposent de manière plus ou moins explicite sur des modèles d'apprentissage des connaissances. Nous détaillerons ici plus particulièrement ce qui touche à la pratique d'un langage de programmation en examinant tout d'abord les caractéristiques psychologiques de la situation de programmation, puis nous nous intéresserons à la programmation comme

situation d'apprentissage en essayant de répondre à la question : "Qu'apprend un enfant qui construit des programmes ?", ensuite, nous exposerons brièvement les caractéristiques du projet LOGO et nous détaillerons les éléments pertinents de l'analyse de la situation de programmation. Nous proposerons pour terminer, à titre d'exemples, quelques activités de programmation destinées aux enfants dans le cadre scolaire.

A – QUELLES SONT LES CARACTERISTIQUES PSYCHOLOGIQUES DE LA SITUATION DE PROGRAMMATION ?

1 – **LA MOTIVATION** est souvent mise en avant par les spécialistes comme étant un élément déterminant de l'introduction de l'ordinateur dans l'enseignement. Certes, cet aspect du problème n'est pas négligeable : attrait de la nouveauté, aspect ludique (les jeux vidéo), objet fascinant et magique . . . mais si l'ordinateur n'était que cela, l'intérêt tomberait bien vite.

Nous n'analyserons pas ici en détail cette caractéristique de la situation, mais nous la reprendrons plus loin, à propos de l'approche LOGO, car cette motivation dépend en grande partie du matériel mis à la disposition des enfants.

2 – **LA SITUATION DE PROGRAMMATION** est assimilée classiquement à une situation de résolution de problème. Cette situation a fait l'objet de nombreuses recherches en psychologie cognitive et en psychologie du développement. Elle permet d'étudier comment le sujet mobilise ses connaissances pour trouver une solution à une tâche problématique. Or, la difficulté majeure à résoudre, quand on est face à un problème, ce n'est pas tant le calcul qu'il faut effectuer que le fait de déterminer les éléments pertinents sur lesquels faire porter ce calcul. Les recherches dans ce domaine tentent de déterminer les conditions dans lesquelles s'opère ce choix et la situation de programmation offre dans ce domaine des perspectives nouvelles. En effet la démarche algorithmique (voir ci-dessous) suppose au préalable la construction d'unités sémantiques qui rassemblent toutes les idées que se fait l'enfant de l'objet qu'il a décidé de programmer.

Une situation de résolution de problème se définit par trois critères :

– Un état initial (par exemple la disposition des pièces au départ sur un échiquier).

– Des règles du jeu (par exemple le déplacement des pièces d'un jeu d'échec).

– Un état final (dans notre exemple, le mat ou le nul).

En programmation, l'état initial est représenté par le projet du "programmeur" : dessiner un carré, conjuguer un verbe du 1er groupe . . . ; les règles du jeu sont la syntaxe du langage de programmation et le fonctionnement du microprocesseur ; l'état final, c'est la construction de la liste d'instructions qui conduit au résultat escompté.

Ce qui distingue la programmation des situations de résolution de problème classiques, c'est, d'une part, l'infinité des buts que l'on peut se donner, et, d'autre part, le fait que le résultat n'est pas ce but lui-même, mais la procédure de construction qui y mène.

On peut aussi mentionner que les règles du jeu, c'est-à-dire la syntaxe et le fonctionnement du langage de programmation, nécessitent un apprentissage spécifique et parfois assez long. La programmation à un haut niveau est une technique complexe et difficile mais cela ne

veut pas dire que les bases de cette technique ne sont pas abordables par tout un chacun. L'inaccessibilité du calcul matriciel n'empêche pas un enfant de cinq ans de commencer à apprendre à compter.

3 – LA DEMARCHE ALGORITHMIQUE

L'enfant (ou l'adulte) pour construire un programme, doit définir clairement le résultat qu'il veut obtenir et les moyens qu'il se donne pour l'atteindre. Cela suppose d'exprimer en un langage clair un énoncé qui tienne compte de toutes les particularités du problème.

A ce stade de l'analyse, le langage quotidien tient lieu de langage universel de programmation. Par exemple, avec la tortue LOGO, pour dessiner un carré, il faut avancer d'une certaine longueur, tourner à droite (ou à gauche) d'un angle de 90° , et recommencer cette séquence trois fois.

Cette analyse de la tâche repose sur la construction préalable par l'enfant d'un modèle qui rassemble de manière plus ou moins structurée suivant l'âge ou le niveau de connaissances, l'ensemble des idées et des conceptions que le sujet se fait de l'objet à construire :

– Qu'est-ce qu'un carré ? un cercle ?

– Qu'est-ce qu'une liste de nombres ordonnée ? . . .

C'est peut être à ce stade de la programmation que seront le plus sensibles les différences de niveaux de développement entre les enfants. L'enfant ne programmera qu'en fonction des représentations qu'il se fait de l'objet à construire. Cela ne veut cependant pas dire que la programmation ne l'aidera pas à construire de nouvelles représentations chemin faisant. Nous reviendrons ultérieurement sur ce sujet.

Puis, l'enfant doit traduire ce modèle dans un langage clair, c'est-à-dire dépouillé de toute considération superflue, comprenant une série de propositions compatibles avec l'automatisation (exécutables par un tiers) et avec la structure du langage de programmation. D'où l'intérêt de posséder, pour débuter, un langage de programmation proche de la logique quotidienne de l'action, pertinent du point de vue logique et puissant, c'est-à-dire capable, si l'on combine ses ordres, de créer de nouveaux outils.

Cette activité que nous venons de décrire repose sur une notion déjà bien étudiée par les psychologues : la planification des actions. On sait que la résolution de problèmes est facilitée si on incite le sujet à planifier ses actions avant de réaliser une tâche. On sait aussi que les jeunes enfants avant 7 ou 8 ans n'en sont pas capables. A cet âge, c'est l'action seule qui permet le contrôle du déroulement des séquences, à la vue des effets de l'action sur les objets construits. La construction de programmes, qui suppose obligatoirement cette activité, se révèle par là même très intéressante dans l'apprentissage des connaissances.

4 – L'APPRENTISSAGE D'UN LANGAGE DE PROGRAMMATION

Programmer, si on ne veut pas en rester à la seule algorithmique qui peut se pratiquer même sans ordinateur, suppose à un moment ou à un autre que l'on acquière une certaine maîtrise d'un langage de programmation compatible avec le fonctionnement du processeur.

Cet apprentissage nécessite la mise en place d'une représentation de la syntaxe et de la structure du langage : qu'est-ce qu'une primitive, une procédure ? Comment édite-t-on une procédure ? Comment définit-on une variable ? Comment transférer mon programme sur un support magnétique pour le sauver ? . . .

Cet apprentissage suppose une décentration entre son propre système de pensée et le fonctionnement de la machine. La machine répond toujours juste à la question qu'on lui pose. Il s'agit pour l'utilisateur de pénétrer la logique de la machine pour pouvoir mieux utiliser sa puissance.

Comment apprendre un langage de programmation ? Ce n'est pas parce que nous avons dissocié cet apprentissage de l'algorithmique que nous considérons ces deux activités comme indépendantes. Au contraire, la représentation et la mémorisation du langage passent par la construction d'unités sémantiques liées aux concepts qu'elles font tourner. Si on vous demande d'apprendre une suite de 25 lettres de l'alphabet prises au hasard et les mêmes lettres organisées en mots signifiants, le résultat de la restitution des lettres apprises sera très différent. La mémorisation d'un langage de programmation, comme pour une langue naturelle, passe par une nécessaire pratique qui affecte progressivement aux mots et aux tournures un sens lié au résultat de l'action qu'ils produisent.

5 – LA PROGRAMMATION EST UNE ACTIVITE STRUCTURANTE ET EVOLUTIVE :

Construire un programme, surtout dans la phase initiale de l'apprentissage, peut, chemin faisant, donner des idées pour la construction d'autres programmes. L'utilisateur a toujours l'impression en programmant qu'il progresse sans cesse et que chaque but devient un sous-but pour un nouveau projet. Il est classique d'observer un enfant construire un carré, puis de le voir rajouter un triangle pour le transformer en maison et enfin de décider de répliquer la maison pour obtenir un village.

Cette situation incite progressivement les "apprentis programmeurs" à construire des outils et des sous-programmes (en LOGO des procédures) de plus en plus puissants et généraux. Le "programmeur" apprend à dissocier dans une situation ou une configuration ce qui est stable (invariant) de ce qui varie (paramètre).

La démarche inverse de celle que nous venons de décrire, et qui s'impose très vite au sujet programmeur, est celle de la décomposition d'un problème en petites parties ou sous-procédures. Un projet de programme, dès qu'il devient un peu plus ambitieux que la construction d'un carré à l'aide de la tortue graphique, nécessite l'élaboration d'une construction modulaire où chaque élément est une entité indépendante qui ne prend cependant son vrai sens que juxtaposée ou coordonnée en un tout. Cette méthode de programmation est connue des spécialistes sous le nom de "programmation structurée".

MINSKI et PAPERT ont même élaboré une "théorie sociale de l'esprit" qui défend l'idée générale que tout système repose sur des sous-systèmes relativement indépendants les uns des autres. Dans cette perspective, la programmation conduit à organiser les connaissances et à construire de nouveaux outils. Pour PIAGET, la pensée se développe de la même manière, un beau sujet de réflexion pour les psychologues et les pédagogues.

B – QU'APPREND L'ENFANT QUI PROGRAMME ?

On peut sur un plan général distinguer deux "pratiques" de la programmation quant aux processus d'apprentissage que l'on sollicite chez l'enfant :

– La programmation permet un travail d'analyse et d'organisation des objets sur lesquels portent le programme. L'apprentissage est alors dirigé sur les contenus programmés, l'activité de programmation n'est alors qu'une situation pédagogique particulière qui permet d'aborder la géométrie, le calcul, la musique . . .

– On peut, d'un autre point de vue, centrer l'apprentissage sur la programmation comme activité structurante sans s'intéresser particulièrement aux contenus.

La première perspective concerne au premier chef le didacticien, dont l'objet d'étude est l'enseignement d'une discipline, alors que la seconde interpelle plutôt le psychologue de l'apprentissage.

En ce qui concerne le deuxième point, il est probable (en attendant des recherches plus approfondies) que la programmation favorise l'exercice de certaines activités intellectuelles. Parmi les hypothèses les plus probables, on peut citer :

1 – Le développement des habiletés de description formelles. L'enfant (ou l'adulte) peut, par la programmation, manipuler des concepts difficiles à aborder dans le contexte classique comme ceux d'état, de procédure, de variable, de récursivité. . . Ce point nous renvoie à ce que l'on sait déjà du développement des opérations mentales chez l'enfant. A partir de 6 ans, l'enfant met en place des opérations logiques, d'abord sur des contenus concrets (sériation, classification . . .) puis, vers 10-11 ans, devient capable progressivement d'appliquer ces opérations à des objets plus abstraits (raisonnement hypothético-déductif). Il en résulte la construction d'opérations spécifiques (les opérations formelles) qui conduisent à la maîtrise de la proportionnalité, de la combinatoire, de la logique des propositions et, de manière plus générale, à la possibilité de coordonner des systèmes d'opérations par inversion ou réciprocité.

Ce développement, d'après PIAGET, s'effectue par intériorisation et coordination des schèmes opératoires et la controverse sur l'acquisition des opérations formelles pourrait trouver avec l'apprentissage généralisé de la programmation dès le cycle primaire, un prolongement intéressant.En effet, la coordination des procédures dans le langage LOGO est un processus analogue à la coordination des schèmes, et aboutit à la construction d'opérateurs stables, généraux et répétables (récursivité, itération, branchements conditionnels . . .)

2 – Le codage et l'objectivation des actions, c'est-à-dire la possibilité de décrire de manière précise le résultat éventuel d'une transformation que l'on fait subir à un objet. La programmation, comme nous l'avons déjà souligné, rend les idées, les plans et les représentations du sujet directement observables. Le "programmeur" prend l'habitude de relier directement la représentation d'une transformation au déroulement effectif de l'action. Cette acquisition est fort précieuse pour le développement des activités de planification de l'action, qui sont elles-mêmes déterminantes pour la résolution de problème.

3 – La découverte, en les apprenant à un automate, de ses propres manières de penser. Il s'agit d'une véritable aide à la réflexion sur sa propre pensée. Le programme qui se construit peut être considéré comme une surface de projection de ses propres schèmes. Le sujet peut ainsi objectiver sa façon de résoudre le problème et cela nous conduit à analyser plus finement le statut particulier des erreurs en programmation qui ont fait l'objet de nombreuses recherches expérimentales.

On distingue trois types d'erreurs en programmation :

– Les erreurs syntaxiques qui provoquent au moment de l'exécution de l'instruction un refus de la part du processeur, refus exprimé de différentes manières suivant le langage ("Syntax Error" en BASIC, "Ne sait pas comment TOTO" en LOGO si TOTO est une procédure non définie préalablement). Ce type d'erreur est facile à repérer dans un programme, elles sont surtout le fait des débutants.

– Les erreurs sémantiques sont des erreurs qui ne conduisent pas l'algorithme à ce qu'il était sensé réaliser. Par exemple, les enfants, pour dessiner un carré à l'aide de la tortue LOGO, écrivent souvent REPETE 4 (AVANCE 50 DROITE 50) en pensant que la tortue tournera à angle droit.

– Les erreurs logiques, qui sont des erreurs de conception de l'algorithme. Ce dernier traduit bien ce que le sujet voulait faire, mais il ne conduit pas au résultat voulu.

Les erreurs du 2ème genre sont les plus fréquentes, elles sont souvent très difficiles à corriger et concernent principalement les affectations de valeurs dans les variables.

Dans la perspective psychologique qui est la nôtre, les erreurs logiques sont les plus intéressantes, elles débusquent les faiblesses de nos représentations de la tâche et permettent ainsi d'avoir de nouvelles idées fécondes.

4 – De manière plus générale, tous les auteurs sont unanimes pour penser que la programmation développe la capacité à résoudre des problèmes par un exercice des aptitudes d'analyse et de synthèse.

5 – Sur le plan affectif et relationnel, on peut noter que l'apprentissage de la programmation, avec un dispositif attrayant et simple comme les "micro-mondes" du langage LOGO (voir ci-dessous), provoque une attitude de satisfaction chez l'enfant et une réduction du niveau de frustration face à l'apprentissage. En effet, l'enfant devient un sujet actif, il prend plaisir à découvrir de nouvelles propriétés aux objets, de nouveaux moyens pour réaliser ses projets, de nouveaux buts. Les erreurs ne sont plus l'objet d'une sanction de la part de l'adulte, mais conduisent l'enfant à perfectionner sans cesse son travail par un ajustement progressif du programme.

Il faut aussi souligner les importantes possibilités d'interactivité entre élèves que permet la mise en place de projets collectifs. De plus, la maîtrise de cet objet technologique complexe qu'est le micro-ordinateur crée un lien, bien rare à notre époque, entre leur monde et le monde des adultes.

C – EN QUOI CONSISTE L'APPROCHE "LOGO" ?

L'approche "LOGO" est un projet éducatif développé par MINSKI et PAPERT, spécialistes en Intelligence Artificielle (science de la simulation de l'intelligence par des automates) du M.I.T. aux Etats-Unis.

Ces chercheurs s'appuient sur le rôle des activités concrètes dans l'acquisition et la structuration des connaissances. Ils se proposent d'utiliser les technologies nouvelles pour fournir aux enfants des situations qui leur permettent d'acquérir une expérience réelle sur les objets de connaissance.

L'organisation d'un domaine de connaissance doit être une "reconstruction" impliquant la mise au point de concepts simples, accessibles à tout enfant, mais suffisamment puissants pour permettre d'aborder les notions essentielles liées à ce domaine. Par exemple la géométrie tortue de LOGO repose sur deux idées force :

– Le concept d'état (la tortue est repérable sur l'écran par sa position et sa direction)

– Les commandes de changement d'état (la tortue peut avancer et tourner).

A partir de ces deux notions l'enfant peut aborder une grande partie des concepts de la géométrie classique : les angles, la symétrie, les polygones . . . Il participe activement à la découverte des théorèmes et des propriétés des figures qu'il construit.

Ces principes font ressortir l'influence que PAPERT a reçue, d'une part des travaux de PIAGET avec lequel il a collaboré pendant de nombreuses années, et d'autre part des travaux de SIMON en Intelligence Artificielle.

De PIAGET, PAPERT a retenu le rôle essentiel de l'expérience et des activités concrètes dans la construction des opérations de la pensée et de SIMON la nécessité de proposer aux enfants un langage de programmation proche du langage quotidien, simple mais suffisamment puissant pour aborder tous les domaines de la résolution de problème.

Si LOGO est une théorie de l'apprentissage, c'est aussi et avant tout un langage de programmation, proche parent du LISP qui a été mis au point pour traiter les situations de résolution de problème en Intelligence Artificielle. Ce langage est interprété, il permet donc une utilisation directe des commandes pour tester la mise au point des procédures (cet aspect technique se révèle important quand on travaille avec des débutants).

Les caractéristiques du LOGO mettent à la disposition du "programmeur" un langage qui comprend des primitives traitant aussi bien le graphisme (la "tortue" LOGO) que les mots et les listes. Il est construit pour une programmation structurée, sans numérotation de lignes. Un programme en LOGO comprend des procédures qui sont mises au point par l'intermédiaire d'un éditeur dont le maniement par les enfants ne pose pas de gros problèmes d'apprentissage. Les enfants apprécient tout particulièrement la possibilité de la libre dénomination des procédures et des variables, qui leur permet à la fois de personnaliser leurs programmes et de travailler sur la lisibilité des instructions.

Le "micro-monde" de la "tortue", qui est l'aspect de LOGO le plus connu du public, comprend d'une part des commandes qui permettent de modifier la direction ou la position de la tortue (AVANCE RECULE CENTRE DROITE GAUCHE . . .) et d'autre part des fonctions ou opérations qui nous renseignent sur sa localisation à l'écran et son cap (POSITION CAP XCOR YCOR . . .). Ces primitives font appel à des notions simples de déplacement que l'enfant peut lui-même effectuer dans l'espace. Toutes les expériences de "terrain", qui ont testé la manipulation du LOGO graphique avec des enfants, soulignent ses qualités de simplicité et d'intelligibilité dans la forme des primitives proposées, ainsi que la commodité de sa syntaxe. Ces caractéristiques donnent à l' "apprenti programmeur" un accès rapide à une bonne représentation du fonctionnement du langage.

Les trois structures fondamentales de l'écriture des programmes : les structures séquentielles, les branchements conditionnels, les boucles itératives, sont accessibles de manière claire. A ces structures s'ajoute la possibilité d'utiliser la récursivité qui autorise, dans la définition d'une procédure, l'appel à tous les niveaux de cette même procédure.

Le traitement des MOTS et des LISTES (les deux sortes d'objets LOGO) se fait par l'intermédiaire de primitives qui sont, là encore, soit des commandes (ECRIS, MONTRE . . .), soit des fonctions (PHRASE, MOT, PREMIER, SAUFDERNIER . . .). Il ne nous est pas possible dans le cadre de cet article de détailler davantage la syntaxe et la structure de LOGO, nous renvoyons le lecteur qui voudrait en savoir plus sur son fonctionnement aux ouvrages de PAPERT, REGGINI et BOSSUET cités en bibliographie.

Pour conclure, on peut dire que l'idée de PAPERT est de mettre à la disposition des enfants un véritable "laboratoire d'apprentissage et de recherche" qui permet d'une part à l'apprenti programmeur de construire progressivement ses propres opérations de programmation et d'autre part au psychologue de l'apprentissage, au pédagogue ou au didacticien, de ne plus uniquement faire un discours sur l'apprentissage mais aussi une observation des processus mis en œuvre par le sujet qui est confronté à la résolution d'un problème.

D – LA SITUATION D'AUTO-APPRENTISSAGE

Dans une note de recherche, J. HILLEL et J.H. ERLWANGER (1984) rappellent, comme d'autres auteurs auparavent (PAPERT 1979, BOSSUET 1982, LE TOUZE et Col.1979...), les difficultés que rencontrent les enfants dans leur approche de LOGO. Ces difficultés sont de nature différente suivant l'attitude de l'enseignant-expérimentateur lors des séances d'apprentissage. Dans la conception première de PAPERT, LOGO est construit pour permettre à l'enfant un auto-apprentissage avec un minimum d'intervention de la part de l'adulte. Dans cette situation,tous les auteurs constatent que spontanément l'enfant et même l'adulte utilisent largement la technique "essais et erreurs" en travaillant en mode pilotage. Chaque instruction est exécutée et l'activité se résume à un pilotage de la tortue graphique pas à pas. La rencontre de l'enfant avec d'autres instructions que les primitives de déplacement et d'orientation, comme l'instruction REPETE, entraîne chez lui un changement de stratégie : il devient moins sensible à un projet défini à l'avance et il utilise de manière contrôlée la puissance de la machine pour tenter des expériences où le contrôle de l'action passe d'un guidage pas à pas, aux systèmes de contrôle inclus dans le système.

Toujours en situation de libre apprentissage, on peut constater que l'enfant tient peu compte de paramètres fondamentaux en programmation :

– L'état de la tortue entre sous-séquences de programme qui renvoie aux problèmes que rencontre tout programmeur débutant dans les opérations d'initialisation des variables.

– L'indépendance entre la définition d'une sous-procédure (assimilable à un sous-programme) et son utilisation dans un contexte différent comme par exemple un programme principal ou autre procédure. Ceci se traduit par le fait que l'enfant ne reconnaît pas dans une suite d'instructions qui dessine un carré, un invariant qui permet de le tracer dans n'importe quelle position. On retrouve ici un résultat classique de la psychologie génétique : un carré posé sur une pointe n'est plus un carré, et pourtant, en situation de programmation, la liste d'instructions matérialise l'invariant de la figure à travers les actions qui déterminent son tracé.

– Nous ne ferons que mentionner les difficultés inhérentes aux contenus des objets programmés, comme les figures dessinées avec la tortue graphique et qui relèvent plus de l'enseignement de la géométrie que des activités de programmation : la notion de rotation, la paramétrisation des angles en degrés, les problèmes de symétrie, les "théorèmes de géométrie tortue" que découvre l'enfant dans la construction de figures comme les polygones réguliers (ROUCHIER 1981 et 1983).

En ce qui concerne l'utilisation de LOGO dans une perspective d'auto-apprentissage, nous pensons qu'il s'agit plus d'une position idéologique et pédagogique qui veut instituer certains rapports entre le sujet et les objets sur lesquels il peut expérimenter. Des comptes rendus détaillés de ces recherches sont entre autres disponibles dans les MEMO-LOGO, publiés par le M.I.T.. Ce qui frappe le lecteur c'est l'extrême hétérogénéité des observations et l'absence de méthode d'analyse des productions des enfants (Rapport Brooklyn, 500 pages de description de cas et d'observations, PAPERT 1978).

E – LES CRITERES D'ANALYSE DE LA SITUATION DE PROGRAMMATION

Le point de départ de notre propre recherche se situe dans une perspective diamétralement opposée à celle que nous venons de décrire. Sans condamner sur le plan du principe l'approche de PAPERT, il nous semble indispensable de la compléter par une analyse plus rigoureuse de ce que peuvent réellement faire les enfants en programmation dans des situations où on leur impose certaines contraintes.

Nous avons donc cherché d'une part à standardiser (de manière assez souple comme on le verra) les séances d'apprentissage du code de programmation afin que tous les enfants de notre expérience aient eu la possibilité de manipuler l'ensemble des commandes, des primitives et des structures de programmation constituant le sous-ensemble LOGO qui faisait l'objet de nos investigations, et d'autre part à définir ce que l'on entend par programmation en LOGO. Sur ce dernier point, on peut retenir quatre critères qui font à peu près l'unanimité des auteurs. :

– L'apprenti-programmeur a un but (ce but peut être suivant les cas plus ou moins bien défini à l'avance)

– L'apprenti-programmeur est capable de traduire l'essentiel des caractéristiques du but sous la forme d'une référence à ce qu'il connaît des primitives et des commandes de LOGO.

– L'apprenti-programmeur est capable de maîtriser, au moins sous leurs formes les plus simples, les principales structures de programme : la séquentialité des sous-programmes, l'itérativité, les branchements conditionnels.

– L'apprenti-programmeur est capable de repérer la signification des erreurs qu'il commet et peut mettre en oeuvre une procédure pour les supprimer.

Ces critères qui marquent une frontière, que certains jugeront arbitraire, entre ce qui peut être considéré comme de la programmation et ce qui ne l'est pas, suppose seulement à notre avis que, pour pouvoir être qualifiée de "programmation", l'activité de l'enfant doit être une activité dirigée vers un but explicite (et non totalement aléatoire), planifiée par la connaissance que l'enfant a acquise du fonctionnement du dispositif et des contraintes syntaxiques du langage et enfin faire l'objet d'une indispensable mise au point qui vise à réduire, autant que faire se peut, l'écart entre le but défini et le résultat final.

Cette position théorique oblige le chercheur à tenir compte de certains critères pertinents pour l'analyse de l'activité de programmation. Nous retiendrons les suivants :

1 – L'existence explicite d'un but à atteindre.

Ce but peut être un projet personnel de l'enfant ou un projet imposé par l'observateur. Ce dernier doit tenir compte dans ses observations de la nature des variations que va subir ce but au cours de la construction du programme : ces variations sont-elles dues à une difficulté que rencontre l'enfant ? à une meilleure définition de ce but en fonction de l'analyse que le sujet fait de la tâche ? ou bien encore à la découverte d'un "modèle pratique" qui transforme ce but en un sous-but d'une procédure plus générale ?

2 – La compréhension sémantique des primitives du code de programmation.

Le langage de programmation peut devenir avec tous les degrés de précision qu'il est possible d'imaginer, un support symbolique de communication qui permet aussi bien la production de sens que la compréhension de messages codés. Ce statut particulier du code de programmation peut être évalué à travers des épreuves de compréhension de programme, ou encore des épreuves de complétion ou de modification de programme. Il n'y a pas de raisons théoriques à ce que la maîtrise du code de programmation, comme moyen de transmettre de l'information, fasse appel aux mêmes fonctions cognitives que l'écriture des programmes dont la fonction est de produire de l'information.

3 – Les modalités d'interaction entre l'enfant et l'éditeur pour la mise au point des procédures.

Comme nous l'avons déjà souligné, le sujet débutant utilise une stratégie d'interaction maximale entre l'écriture des instructions et leur exécution par le processeur en faisant réaliser chaque instruction pas à pas. Nos observations personnelles nous ont même montré que certains enfants écrivent ces instructions sur une feuille de papier en les modifiant au fur et à mesure, pour pouvoir éditer définitivement la procédure, lorsque celle-ci est au point., par une stratégie de pilotage.

Une seconde modalité d'interaction mixte consiste pour l'enfant à écrire une liste d'instructions groupées, repérée comme un ensemble plus ou moins pertinent (chunk) dont il veut tester l'effet. Cette deuxième modalité est très utile pour l'observateur qui dispose ainsi d'une lecture plus facile des différents "épisodes", pour employer une terminologie de RICHARD 1982, qui constituent l'approche de la solution au problème posé. On pourrait considérer ces sous-ensembles d'instructions comme des embryons de sous-programmes.

Enfin une troisième modalité consiste pour le sujet à écrire directement dans l'éditeur et à ne tester son effet que globalement, quitte à revenir à l'éditeur pour une mise au point.

Dans la phase d'apprentissage, il est bien entendu qu'il faut montrer à l'enfant qu'il est possible d'utiliser ces trois modalités, tout en le laissant libre de choisir celle qui convient le mieux à ses projets et à ses capacités de planification. Mais ce critère doit être maintenu constant dans le cas d'observations systématiques afin de pouvoir comparer les productions des enfants en sachant de manière précise à quel type de processus le sujet a fait appel.

4 – La prise en compte de l'état de la tortue.

Ce point est implicitement lié à la notion de programmation et l'absence de ce critère, très fréquente chez les débutants, conduit à une impossibilité de mener à bien un projet. Nous avons déjà mentionné plus haut qu'il correspond à la même exigence que l'initialisation des variables dans un programme de traitement de données. La production de sous-programmes, indépendants dans leur écriture, suppose qu'au moment où on les utilise dans un programme principal, chaque passage d'un groupe d'instructions à un autre, laisse le système dans un état connu et prévisible. C'est pour contourner cette difficulté que beaucoup d'enfants écrivent des sous-procédures dépendantes les unes des autres. Ils ont ensuite des difficultés à maîtriser simultanément les deux caractéristiques d'un programme structuré : indépendance et coordination des sous-programmes.

5 – La connaissance et l'utilisation de techniques de programmation.

Il faut avoir vu des enfants livrés à eux-mêmes, sans autres connaissances que celles qui permettent de faire tourner la tortue, pour comprendre la pauvreté des productions dont ils sont capables spontanément (voir à ce sujet H. WERTZ 1983) si on ne leur donne pas la possibilité d'utiliser de véritables techniques de programmation qu'il n'est généralement pas possible d'inventer, comme l'écriture récursive des procédures.

La programmation n'est pas, comme certains le laissent croire dans un souci commercial, un lieu de création où tout peut être découvert spontanément. La programmation repose principalement sur un objet technologique dont les contraintes sont très fortes. Pour produire des programmes d'un certain niveau, l'enfant a besoin qu'on lui enseigne des techniques de programmation. Alors seulement, on pourra voir quel usage il en fera et surtout comment il intégrera ces connaissances technologiques aux connaissances qu'il posséde par ailleurs. Toute observation d'activités de programmation chez l'enfant doit mentionner la liste des techniques qui ont fait l'objet d'un apprentissage et la description des situations dans lesquelles ces techniques ont été acquises par le sujet.

6 – La compréhension des erreurs et la possibilité de mettre en oeuvre une procédure tendant à amoindrir ou à supprimer leurs effets.

PAPERT, dans son ouvrage de 1980, a donné aux erreurs un statut psychologique particulier : "Apprendre à passer maître en l'art de programmer, c'est devenir hautement habile à déceler où se nichent les "bugs" et à y remédier . . .", il continue ainsi "la question à se poser, au sujet d'un programme, n'est pas de savoir s'il est juste ou faux, mais si on peut l'arranger". Construire un programme, c'est le perfectionner sans cesse car il arrive toujours que l'on oublie un paramètre du problème ou qu'on néglige un effet, apparemment mineur, qui peut venir, à un moment imprévisible, démolir une belle construction solide. Ce statut particulier des erreurs en programmation a fait l'objet de nombreuses recherches en "programmation professionnelle" (YOUNG 1974, HOC 1981) et a donné lieu à plusieurs classifications.

7 – Le style général de programmation.

Ce dernier critère, moins opérationnel que les précédents, est cependant très intéressant pour l'analyse des activités de programmation. Chaque sujet, aux différents stades de son apprentissage, se caractérisera par une stratégie générale dans la façon de construire un programme : exploration pas à pas ou par groupe d'instructions, planification des opérations, analyse préalable de l'objet à construire . . . La psychologie différentielle pourra trouver là un objet d'étude pertinent qui pourrait enrichir la problèmatique des styles congnitifs.

F – QUELQUES EXEMPLES D'ACTIVITES DE PROGRAMMATION POUR LA CLASSE

Le projet de cet article était de proposer quelques directions de recherche pour aborder l'étude de la situation de programmation chez l'enfant et ses effets éventuels de transfert sur d'autres activités intellectuelles. Nous menons actuellement de telles recherches dans deux Groupes Scolaires de GRENOBLE (Groupes Scolaires BIZANET et Paul BERT), elles feront l'objet de futurs articles. Nous ne voulons cependant pas terminer sans mentionner à titre d'exemples, bien que cela ne soit pas de notre compétence, quelques exercices abordables en cycle primaire :

1 – L'ALGORITHMIQUE :

On peut faire de la programmation sans ordinateur en essayant de décrire, avec le langage ordinaire, les actions nécessaires à l'accomplissement d'une tâche. Un élève peut même servir d'automate, et exécuter les ordres que ses camarades lui donnent et uniquement ceux-là !

2 – LA MISE AU POINT DE PROGRAMMES :

Il est toujours bon de commencer à faire travailler les enfants en mode "pilotage" (les ordres sont alors exécutés au fur-et-à-mesure de leur introduction au clavier).

Les enfants peuvent commencer avec des projets libres (l'adulte reste là pour indiquer aux enfants si le projet est réalisable) puis progressivement, après plusieurs séances, quand l'enfant commence à acquérir une bonne représentation du fonctionnement de la machine, il est souhaitable de lui proposer des exercices imposés pour lui permettre une exploration plus complète des possibilités de l'ordinateur et du langage.

Il faut continuellement veiller à une bonne verbalisation des actions que fait l'enfant en incitant à expliquer fréquemment ce qu'il est en train de faire. Un bon moyen pour atteindre ce résultat, consiste à lui demander de tenir un "cahier de programmation" sur lequel il inscrit tout ce qu'il a réalisé à l'écran.

Il est aussi important de proposer aux enfants de perfectionner sans cesse leurs programmes, en essayant de découvrir de nouveaux moyens d'atteindre les objectifs qu'ils se sont fixés.

3 – L'EXTENSION DU VOCABULAIRE DU LANGAGE DE PROGRAMMATION :

Un bon exercice, pour maîtriser parfaitement la structure du langage, consiste, avec LOGO exclusivement, à enrichir le langage de nouveaux mots ou de nouvelles abréviations, en définissant de nouvelles procédures, qui sont ensuite utilisées comme des primitives.

Cette activité permet une meilleure appropriation de la machine par l'enfant et lui donne l'occasion de se dégager des conventions imposées par le constructeur.

4 – LIRE ET COMPRENDRE DES PROGRAMMES DEJA REDIGES :

Cette activité développe une bonne maîtrise de la syntaxe et des tournures du langage de programmation. Elle donne aussi à ce dernier un statut de langage de communication.

5 – CONSTRUCTION D'OUTILS D'AIDE A LA PROGRAMMATION :

On peut inciter les enfants à construire des outils qui peuvent servir à dessiner, à mesurer une distance, à se déplacer sur l'écran . . . Si parfois la programmation de ces outils est trop complexe pour eux, on peut néanmoins leur apprendre à demander la construction de tels outils.

6 – PROPOSER DES PROJETS COLLECTIFS :

La programmation n'est pas forcément, comme on se plait souvent à la présenter, une activité solitaire. Il est possible, et même souhaitable, que les enfants apprennent à se partager un problème. Cette activité suppose un travail de coordination entre les élèves qui vient matérialiser la nature structurée de l'activité de programmation.

Ces propositions ne sont bien entendu pas limitatives et le pédagogue intéressé par cette activité n'aura pas de peine à enrichir son répertoire.

BIBLIOGRAPHIE

ABELSON H. – Le Logo sur Apple. Paris, Cedic-Nathan, 1984.

BOSSUET G. – L'ordinateur à l'école. Paris, PUF, 1982.

HILLEL J., ERLWANDER S.H. – Observations, reflexions and questions about childrens' Logo learning. Concordia University, Montreal, 1984.

HOC J.M. – Etude de la formation à une méthode de programmation informatique. Le Travail Humain, 1978, 41, 1, 111-126

HOC J.M. – Le problème de la planification dans la construction d'un programme informatique. Le Travail Humain, 1979, 42, 2, 245-260.

HOC J.M. – L'étude psychologique de l'activité de programmation, une revue de la question. Technique et Science Informatique, 1982, 1, 5, 383-392.

HOC J.M. – La psychologie de la programmation : la percée d'une démarche empirique dans un environnement normatif. Le Travail Humain, 1983, 46, 2, 199-204.

LE TOUZE J.C., N'GOSSO I., ROBERT F., SALAME N. – Apport d'un environnement informatique dans le processus d'apprentissage. Projet LOGO. Ministère de l'éducation, I.N.R.P., Mai 1979.

MENDELSOHN P. – Situation de programmation et fonctionnement opératoire chez l'enfant. Société Française de Psychologie, Colloque "Les apports de l'intelligence artificielle et de l'automatique à la psychologie". Grenoble, Mars 1984.

MENDELSOHN P. – L'analyse psychologique des activités de programmation chez l'enfant. (à paraître dans la revue ENFANCE 1985)

NIVAT M.– Savoir et savoir-faire en informatique. Rapport aux Ministres de l'Education Nationale et de l'Industrie et de la Recherche. Paris, La documentation française,1983.

PAPERT S., WATT D., DISESSA A., WEIR S. – Final report of the Brookline LOGO project, Part II : Project summary and Data analysis, Memo Logo 53, Massachusetts Institute of Technology, 1979.

PAPERT S.– Jaillissement de l'esprit : ordinateurs et apprentissage. Paris, Flammarion, 1981.

PAPERT S.– Les ressources de l'enfant et l'ordinateur. in R. COHEN, Plaidoyer pour les apprentissages précoces. Paris, PUF, 1974

PIAGET J.– Réussir et comprendre. Paris, PUF, 1974.

PIAGET J.– L'équilibration des structures cognitives. Paris, PUF, 1975.

REGGINI H.C. – LOGO, des ailes pour l'esprit. Paris, Cedic-Nathan, 1983.

ROUCHIER A.– Problèmes, procédures, programmes étudiés et réalisés par des enfants de CM 2 utilisant un mini-ordinateur. Revue française de pédagogie, 1981, 56, 18-26.

ROUCHIER A. – Logo et les contenus d'enseignement et de formation. Actes du 1er Colloque LOGO, Clermont-Ferrand, décembre 1982, p.28-44. Edité par l'IREM d'Orléans, mars 1983.

ROUCHIER A. – LOGO, quelle géométrie pour la tortue ? Intervention faite à la faculté des Sciences de Rabat. Document ronéotypé, IREM, Université d'Orléans, 1984.

WERTZ H. – Quelques problèmes concernant l'adéquation des langages applicatifs pour la représentation des processus cognitifs. Actes du 1er Colloque LOGO, Clermont-Ferrand, décembre 1982, p. 3-18. Edité par l'IREM d'Orléans, mars 1983.

YOUGS E.A. – Human Errors in Programming. Int. J. Man-Machine Studies, 1974, 6, 361-376.