LANGAGES ORIENTÉS OBJET

Jacqueline ZIZI

ORIGINES

C'est vers la fin des années 60 qu'est né le langage SIMULA en Norvège. C'est de ce langage de simulation, que proviennent les 3 notions d'objets, de classe et de procédures locales attachées à une classe, que l'on retrouve dans tous les langages à objets d'aujourd'hui et qui les caractérisent.

Simula a été construit à partir d'ALGOL qui donna naissance par ailleurs à PASCAL grâce aux travaux de N. Wirth.

A partir de ces concepts issus de SIMULA, A. Kay créé un langage autonome, où tout est objet : SMALLTALK dont la première version date de 1972. Plus qu'un langage de programmation, SMALLTALK est aussi un environnement de développement et dans les interactions entre l'homme et la machine, les fenêtres et la souris sont prédominants. Ce sont ces idées qui guident l'ergonomie du MACINTOSH et arrivent maintenant, quelques 20 ans après, dans le monde PC avec WINDOWS.

CARACTERISTIQUES

De nombreux langages à objets et LOO, Langages Orientés Objet ont vu le jour depuis, soit dans la lignée de SIMULA, soit dans celle de SMALLTALK et présentent, suivant leurs tendances, des caractéristiques fort différentes. Mais quoiqu'il en soit, on retrouve dans tous les langages à objets :

- des objets ,
- des classes,
- des procédures locales appelées méthodes.

L'évolution de l'ensemble se fait sous la pulsion des envois de messages aux objets et de leurs réactions. Le processus d'héritage,

permet d'obtenir automatiquement sur des sous-classes et donc des objets de ces sous-classes, les propriétés définies au niveau des classes.

Programmer en LOO consiste à :

- définir des classes et des sous-classes et indiquer les relations d'inclusion des classes,
- définir des objets et leur appartenance à une classe, ce que l'on appelle les instances d'une classe,
- définir des méthodes valables pour des classes d'objets,
- formuler des envois de message.

Au déroulement d'un programme LOO provoqué par l'enchaînement des messages est souvent associé un contrôle à l'aide du langage de base. Ceci bien sûr ne s'entend que pour les langages construits au-dessus d'un autre langage de programmation et quand l'accès au langage de base est possible à partir de la couche objet. Dans ce dernier cas, l'utilisateur peut tout aussi bien définir des classes et des méthodes et en même temps bénéficier d'un langage de programmation procédural ou applicatif, voire les deux, suivant la nature du langage sous-jacent.

Très grossièrement parlant, de même qu'algorithmique et fonctionnement séquentiel de la machine sont très étroitement liés et ont souvent été représentés par un organigramme sur lequel on suit le déroulement des opérations, la conception par objets met en évidence les concepts intervenant dans un problème ainsi que leur points communs et leur relations. La structure globale de l'ensemble est souvent représentée par une forêt, chaque arbre représentant les relations d'inclusion entre classes, les feuilles étant les objets de la classe la plus imbriquée. Les liens entres les différents arbres sont alors parfois explicités par des flèches mais bien plus souvent sont pensés ou imaginés comme le sont ceux des différentes vues des objets mathématiques.

En pratique, l'approche globale et la suite des opérations à effectuer ne sont pas toujours aussi distinctes. Par exemple, dans la vie courante, pour apprendre à un utilisateur à sortir un café d'une machine expresso on peut soit lui indiquer la marche à suivre, soit décrire les fonctionnalités des différents constituants, soit encore associer les deux en mettant en évidence le point de vue qui paraît le plus important à un moment donné. Par exemple, on lui indiquera : "d'abord tu tournes le bouton noir, ensuite tu appuies sur le bouton rouge et puis tu tournes le bouton noir dans l'autre sens pour arrêter la vapeur". En effet, expliquer

"à quoi ça sert" ou donner un cadre global au déroulement d'une action facilite sa mémorisation. De même, pour la programmation et la conception par objets, l'approche globale qu'elle permet est souvent imbriquée avec l'approche séquentielle permise par le langage de programmation sous-jacent. L'approche globale permet alors de fixer des repères où le fil séquentiel prend corps. Mais on peut tout aussi bien faire de la programmation objet "pur" dont l'aspect algorithmique est écarté ou tout au moins complètement transparent pour l'utilisateur.

Trouver les classes qui permettent le mieux de rendre compte d'une situation déterminée relève de l'expérience de l'utilisateur et de son aptitude à dégager des concepts et leur liens. Définir des méthodes et envoyer des messages relève de l'aptitude qu'a le programmeur de se mettre à la place d'objets complètement différents et de prendre ses responsabilités.

Les représentations simples les plus intuitives conviennent bien souvent. Il arrive aussi que la répétition d'un certain nombre d'opérations fasse comprendre à l'utilisateur qu'une factorisation de certaines propriétés est possible.

Souvent les premières structures sont plus compliquées que nécessaire, mais du fait de l'indépendance des composants, une remise en question simplifiant une partie de la structure n'implique aucune conséquence pour les autres parties. C'est ainsi qu'émergent souvent les concepts fondamentaux après des phases de taille, d'abstraction et de simplification.

Les logiciels adaptés du monde MAC ou les systèmes complexes, sont souvent le fruit de travaux faits dans cette direction. En voici deux exemples.

PREMIER EXEMPLE : WORD

Un traitement de textes agit sur des documents, lettres rapports, etc. Donc sur la classe des documents. A la classe des documents sont donc attachées un certain nombre de méthodes : ouverture, fermeture, enregistrement, fusion avec un autre document, impression etc. Quel que soit le document, les méthodes employées sont les mêmes. Elles sont donc définies au niveau de la classe des documents. Ces méthodes se retrouvent dans le menu déroulant FICHIER, qui en donne la liste. Sélectionner un des item, par exemple IMPRIMER revient alors à

envoyer le message imprimer au document sur lequel on travaille. A réception du message IMPRIMER, le document communique alors au système d'exploitation ses caractéristiques propres. Cela peut être celles qui sont prises par défaut au niveau de la classe de tous les documents, comme cela peut être aussi des caractéristiques instancielles, c'est-à-dire prises par l'utilisateur, au niveau de l'instance c'est-à-dire du document particulier auquel il s'intéresse.

Sur cet exemple, nous voyons qu'il y a deux façons de déterminer les propriétés qui conditionnent les réactions d'un objet. Une première voie est celle de l'héritage. Les propriétés de l'objet sont celles définies au niveau de sa classe ou d'une des sur-classes dont elle hérite. Une deuxième solution consiste à définir au niveau de l'objet particulier auquel on s'intéresse, un certain nombre de propriétés, qui prédominent alors sur les propriétés transmises par héritage, si elles existent. Au passage, on notera, qu'il n'est pas utile d'avoir défini toutes les propriétés d'un objet pour le définir ou le manipuler.

Par exemple, les propriétés héritées par un document sont celles qui sont définies par défaut au niveau de l'application. Les propriétés instancielles sont définies grâce au menu FORMAT qui concerne tous les formats de toutes les classes pour lesquelles ce concept a un sens. Il y a par exemple, en plus de la classe des documents, la classe de tous les paragraphes et aussi celle de tous les caractères.

Un paragraphe n'est pas un document particulier. Un caractère n'est pas un paragraphe particulier et il n'y a donc pas d'inclusion entre les classes des caractères, celle des paragraphes et celles des documents. Au niveau de la classe des caractères sont définies un certain nombre de méthodes concernant la taille, la forme etc. Au niveau de la classe des paragraphes sont définis d'autres méthodes comme le retrait à gauche ou à droite, l'encadrement etc. Au niveau du document sont définis les tailles du corps du document, les tailles des marges etc.

Tout ceci n'est pas anodin pour l'utilisateur : d'une part cette ergonomie permet d'aller rapidement au résultat souhaité lorsque l'on a bien compris de quelle nature est l'entité à laquelle on s'intéresse, et d'autre part, l'utilisateur dispose de facilités pour uniformiser le texte. Par exemple, la notion de styles définie sur la classe des paragraphes permet une uniformisation de tous les paragraphes de la sous-classe des paragraphes ayant même style. Ainsi, tous les paragraphes qui reçoivent un message de style RETRAIT défini par l'utilisateur se mettront en retrait de ce que l'utilisateur a défini. Si l'utilisateur décide de changer

cette donnée et de passer par exemple à un retrait de 1,5 cm au lieu de 1 cm, il suffit que le changement soit effectué dans la méthode définie au niveau de la classe pour que tous les paragraphes correspondant au style RETRAIT se décalent. Pratiquement toutes ces manipulations se font au niveau de la classe des méthodes de STYLE qui forment une sous-classe de celle des méthodes de formatage.

Mais tout avantage a son revers. Il faut bien comprendre à quel niveau sont définies certaines actions pour éviter les désagréments. Par exemple, les petits problèmes d'alignement, de cadrage etc. proviennent souvent du non respect des principes précédents, comme le montre l'exemple suivant.

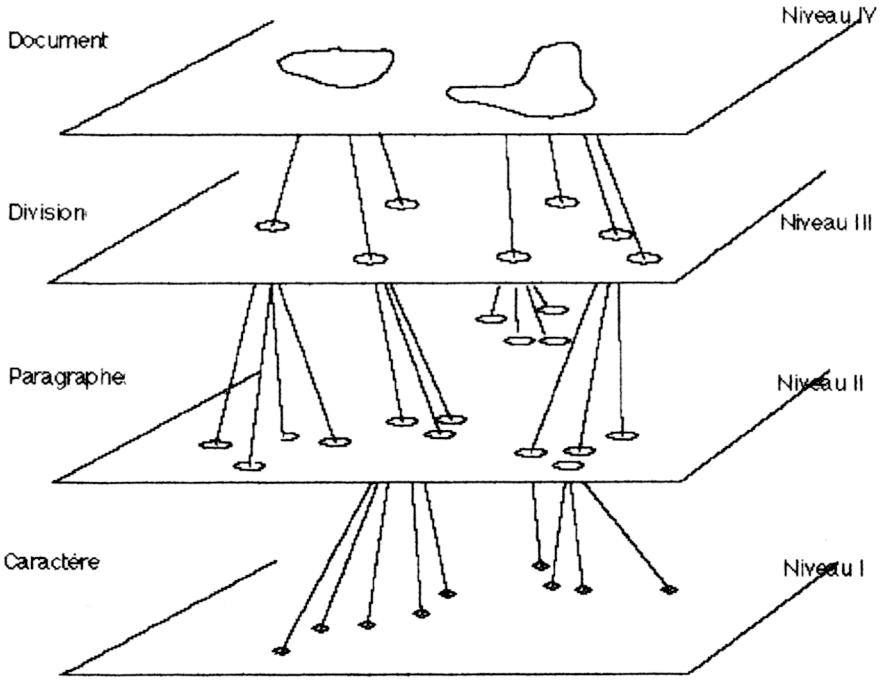
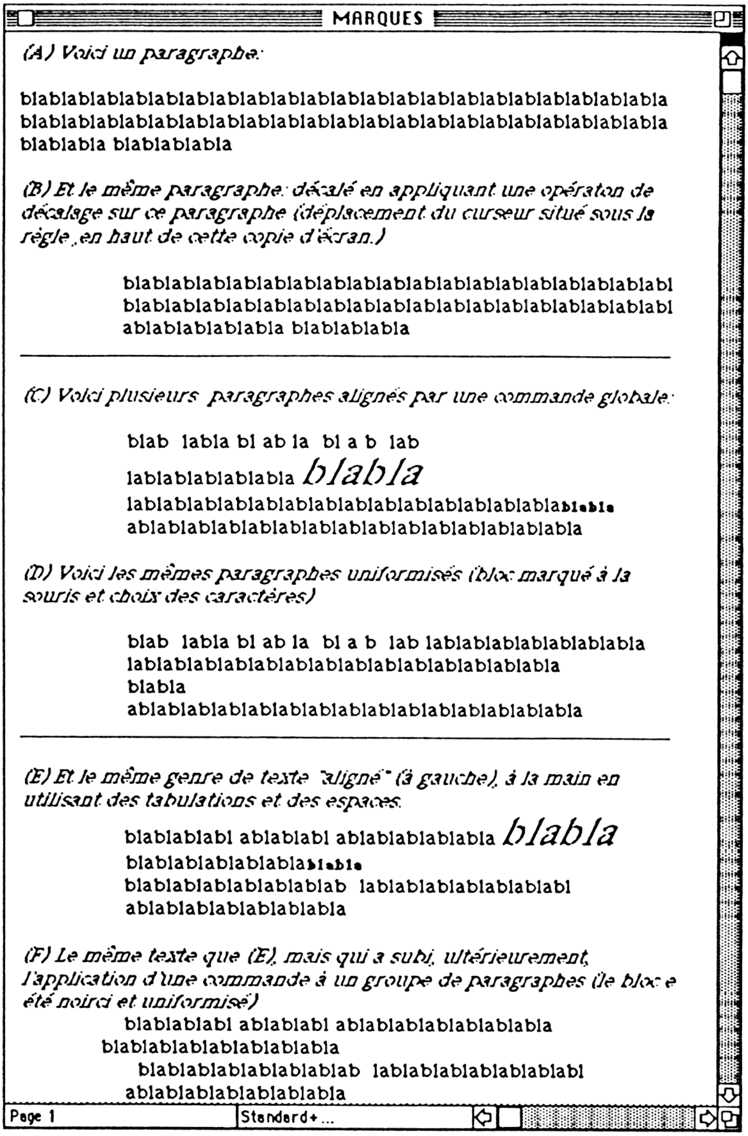


Figure 1 : Regroupement en différents niveaux.


MARQUES

(A) Voici un paragraphe:

blablablablablablablablablablablablablablablablablablablablablabla
blablablablablablablablablablablablablablablablablablablablablabla
blablabla blablablabla

(B) Et le même paragraphe: décalé en appliquant une opération de décalage sur ce paragraphe (déplacement du curseur situé sous la règle, en haut de cette copie d'écran.)

blablablablablablablablablablablablablablablablablablablablabl
blablablablablablablablablablablablablablablablablablablablabl
ablablablablabla blablablabla

(C) Voici plusieurs paragraphes alignés par une commande globale:

blab labla bl ab la bl a b lab
lablablablablabla *blabla*
lablablablablablablablablablablablablablablablabla**blabla**
ablablablablablablablablablablablablablablablabla

(D) Voici les mêmes paragraphes uniformisés (bloc marqué à la souris et choix des caractères)

blab labla bl ab la bl a b lab lablablablablablablabla
lablablablablablablablablablablablablablabla
blabla
ablablablablablablablablablablablablablablablabla

(E) Et le même genre de texte "aligné" (à gauche), à la main en utilisant des tabulations et des espaces.

blablablabl ablablabl ablablablablabla *blabla*
blablablablablabla**blabla**
blablablablablablablab lablablablablablablabl
ablablablablablablabla

(F) Le même texte que (E), mais qui a subi, ultérieurement, l'application d'une commande à un groupe de paragraphes (le bloc a été noirci et uniformisé)

blablablabl ablablabl ablablablablablablabla
blablablablablablabla
blablablablablablablab lablablablablablablabl
ablablablablablablabla

Page 1 Standard+...

Figure 2 : Conséquences facheuses d'un bricolage en W*ORD* *au mauvais niveau.*

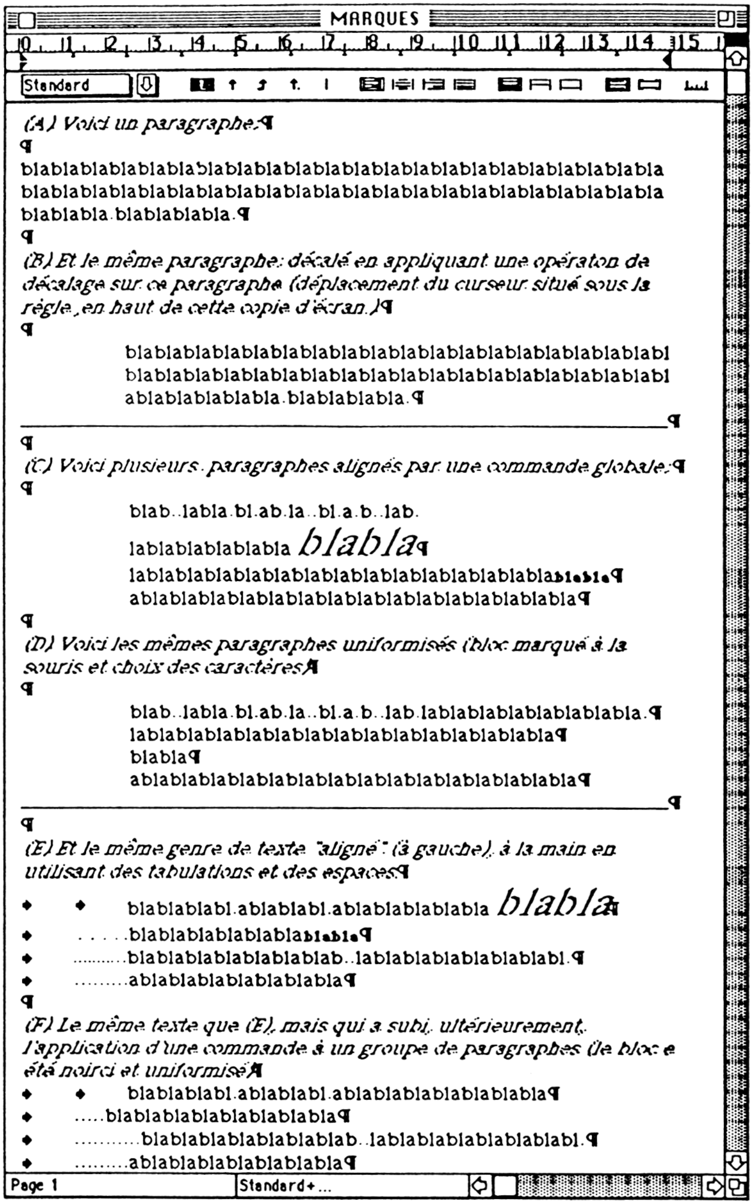


Figure 3 : Pour y voir claire, affichons les marques.

Constat : figure 2

Le premier texte de la première case contient un paragraphe. Il est mis en-dessous en relief par l'action de la règle qui porte sur les paragraphes.

La deuxième case contient plusieurs paragraphes, dans lesquels certaines parties du texte ont été surchargées localement (gras, italique, taille des caractères). Ils sont alignés en retrait à gauche.

La troisième case présente le même texte dans lequel les mises en relief précédentes ont été supprimées et qui montre un problème d'alignement.

Explications : figure 3

Pour comprendre ce qui s'est passé, faire [1] : afficher les marques du menu EDIT. Chaque action effectuée au clavier est alors mise en évidence. Les flèches correspondent à des tabulations [2], les espaces sont représentées par des petits points et les retours chariot [3], par des ¶. On constate que :

- Le premier paragraphe a été mis en retrait par deux tabulations successives.
- Les deuxième, troisième et quatrième lignes ont été mises en retrait par une tabulation et des espaces.

Toutes ces commandes se situent au niveau des caractères (niveau I). Ensuite, l'utilisateur a marqué le groupe de paragraphes et lui a appliqué l'option standard du menu FORMAT, c'est-à-dire qu'il a effectué une action au niveau II portant sur un regroupement de paragraphes. Tous les caractères concernés subissent donc cette modification, et en particulier les espaces utilisés pour l'alignement. Or, la taille des espaces change suivant le format, les polices et les corps des caractères. Les 5 espaces de la deuxième ligne initialement en italique grand format, occupaient une certaine place. Après changement, ces espaces n'occupent plus la même place, d'où le décalage.

1 Ceci correspond aux options WORD.

2 Sur les MAC, c'est la touche où est inscrite une flèche (en haut à gauche du clavier-lettres).

3 Grosse touche du clavier-lettre avec une flèche.

Le même phénomène se produit à la troisième ligne, mais dans l'autre sens, les espaces initiaux en minuscules grasses, occupant une place plus petite que les mêmes espaces en taille normale.

Qu'aurait du faire l'utilisateur ?

Lorsque l'utilisateur s'est trouvé devant le problème d'aligner en retrait son texte, il aurait dû regarder le niveau des éléments concernés par son souhait.

Il s'agit d'un ensemble de paragraphes et donc, ce sont des actions des niveaux II, III ou IV qu'il faut appliquer. Par exemple, il aurait pu tout simplement, une fois le bloc marqué, l'aligner avec le curseur sous la règle. Il aurait pu aussi, avoir la même action au niveau des caractères sur tous les paragraphes concernés (2 tabulations ou mêmes espaces). Il aurait pu également avoir un style retrait à gauche (défini au niveau IV) et l'appliquer aux paragraphes concernés. Plus la méthode employée est décidée à un haut niveau, moins fréquents sont les effets secondaires indésirables [4].

DEUXIÈME EXEMPLE : SYSTÈMES DE CALCUL FORMEL

Les systèmes de calcul formel (cf EPI n° 69 et 70) comme DERIVE, MAPLE et MATHEMATICA sont constitués de centaines, et même de milliers de fonctions organisées en classes et sous-classes. Les barrières de syntaxe franchies, et elles le sont très rapidement, résoudre un problème par manipulation directe des systèmes de calcul formel, consiste à trouver pour un problème donné, la ou les bonnes fonctions. Ceci se fait grâce à un Browser qui permet de parcourir les classes et sous-classes et offre les informations et outils d'édition adaptés. Les browsers de MAPLE et MATHEMATICA sont représentés aux figures 4 et 5. C'est à SMALLTALK que l'on doit cette idée.

4 Souvent appelés effets de bord en informatique. Ce vocabulaire provient d'une traduction littérale de l'anglais. Ici, l'utilisateur a voulu remettre les caractères en standard et il a récupéré un problème d'alignement.

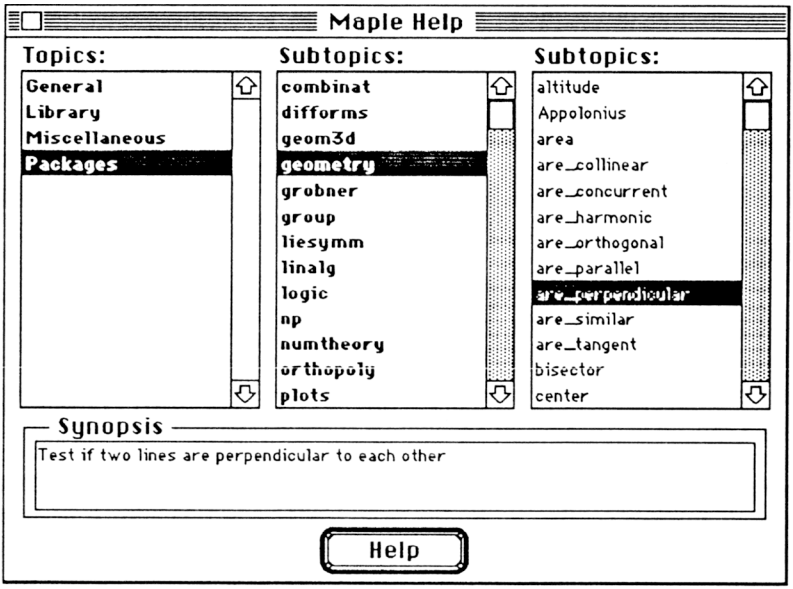


Figure 4 : Le browser de MAPPLE.

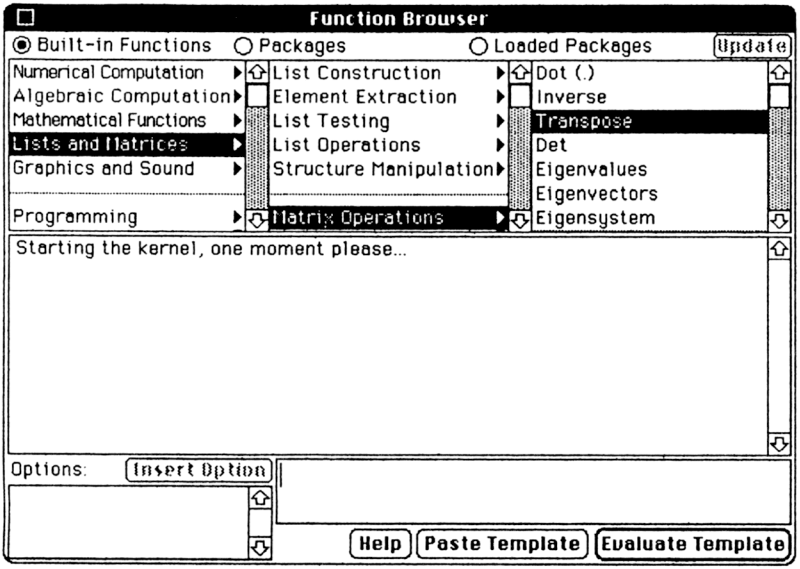


Figure 5 : Le browser de MATHEMATICA.

Il n'y a pas de browser en Derive, mais les fonctions sont organisées en classes et sous-classes de façon interne. Ceci peut se voir par exemple sur la liste des commandes disponibles dans la documentation papier de DERIVE par exemple. En effet, page 241 (version 2.57), on trouve :

Commandes de la fenêtre Algèbre

Auteur

Bâtis

Calcul

- Dérive
- Intègre
- Limite
- Produit
- Somme
- Taylor

Déclare

Il s'agit des fonctions de la fenêtre Algèbre par opposition aux fenêtres graphiques, et qui comportent comme sous-classes de fonctions, entre autres, la classe des fonctions d'Analyse (Calcul est la traduction de Calculus qui veut dire Analyse en Américain). C'est donc dans cette classe que l'on va trouver les fonctions pour dériver, intégrer, déterminer des limites, etc. Par contre, pour faire 2+2, c'est une demande "Auteur" c'est-à-dire de l'utilisateur qu'il faut faire intervenir. En pratique, une bonne compréhension de ces phénomènes amène à appuyer sur les bonnes touches pour accéder aux fonctions désirées et obtenir du système ce que l'on souhaite.

Jacqueline ZIZI

NDLR : les figures sont extraites de l'ouvrage de Jacqueline Zizi : *Mathématiques, Informatique et Enseignement* - Éditions du Choix-Éditions Archimède (voir rubrique "Nous avons lu").