BULLETIN D'INFORMATIQUE APPROFONDIE ET APPLICATIONS

COMPUTATION - INFORMATION

N° 66 - DECEMBRE 2003

COMITE SCIENTIFIQUE :

Patrick Abellard

Françoise Adreit

Jalal Almhana

France Chappaz

M'hamed Charifi

Roger Cusin

Bernard Goossens

Patrick Isoardi

Robert Jacquier

Jean - Philippe Lehmann

Nadia Mesli

Patrick Sanchez

Rolland Stutzmann

André Tricot

CORRESPONDANTS :

Afrique :
Mohamed Tayeb Laskri

Amériques :
Sylvie Monjal

Asie :
Moussa HadjAli

Europe :
José Rouillard

Océanie :
Kalina Yacef

http://scamup.univ-mrs.fr/biaa

Publication trimestrielle, gratuite, de l'Université de Provence

Dépôt légal : janvier 2003

ISSN 0291 - 5413

BULLETIN D'INFORMATIQUE APPROFONDIE ET APPLICATIONS

COMPUTATION - INFORMATION

N° 66 – DECEMBRE 2003

DIRECTEUR :

Jean - Michel Knippel

REDACTEUR EN CHEF :

Edmond Bianco

REDACTEUR ADJOINT :

Sami Hilala

SECRETARIAT :

Kalassoumi Adjilani

Université de Provence
Equipe Hermès. Case 33
3, place Victor Hugo
F - 13331 Marseille Cedex 3
Téléphone: (0)4 91 10 62 30
Télécopie : (0)4 91 50 91 10

DEPOSITAIRE :

Université de Provence
Bibliothèque Vniversitaire
1, place Victor Hugo
F - 13331 Marseille Cedex 3
Téléphone: (0)4 91 10 85 29
Télécopie : (0)4 91 95 75 57

IMPRIMEUR :

Université de Provence
Service Reprographie
3, place Victor Hugo
F - 13331 Marseille Cedex 3
Téléphone: (0)4 91 10 60 48

http://scamup.univ-mrs.fr/biaa

Publication trimestrielle, gratuite, de l'Université de Provence

Impression : juin 2004

ISSN 0291 - 5413

ÉDITORIAL

Intermittence et gouvernance

Edmond Bianco

Intermittence.

Un contraire de ce terme devenu subitement fameux, ne serait-il pas " permanence ", par hasard ? Nous avons déjà réfléchi aux bizarreries des sauts fractals qui peuvent apparaître dans l'expression sémantique de certains mots en fonction du contexte même lointain de leurs occurrences. Passer de " gouverner ", un verbe, à " gouvernance ", un substantif, fait surgir l'un de ces sauts inattendus. Le verbe exprime une action toute bête, qui n'impressionne plus personne, d'autant qu'on sait à quoi rime ce verbe ... berner, lanterner, cerner, concerner (et l'inverse !), etc... Le substantif, lui, représente une notion, il élève, il ennoblit, d'autant que la finale " ance " semblerait avoir tendance à rajouter un degré de poésie. Voire.

Il fut, lors de cet été torride, terriblement question des Intermittents. Qui sont les personnages que représente cet adjectif substantivé ? Tout le monde l'ignorait à peu près lorsque soudain on s'aperçut qu'ils " sousjaçaient " fondamentalement à tout spectacle. Au point qu'un mouvement revendicatif un peu ferme de ce genre de personnel qui vit normalement dans l'ombre, comme vivent les éminences grises, était capable de bouleverser l'organisation des spectacles et des festivals les plus fameux et les mieux organisés, au point d'en bloquer plusieurs et d'en perturber beaucoup. Or, quelle est la propriété fondamentale des Intermittents, sinon l'intermittence ? Ce substantif serait-il la partie noble du verbe " intermitter " ? Il ne semble point. La partie active de la notion correspondante serait plutôt " intervenir par intermittence ". On est alors en droit de s'inquiéter de la qualité d'un tel substantif qui se permet de n'avoir même pas un correspondant vulgairement verbal, qui permettrait d'affirmer modestement : " J'intermitte ! ", ou encore : " Il eût fallu que j'intermitasse ! ".

On vient d'être surpris par le saut fractal qui sanctionne l'accident sémantique séparant le substantif du verbe, dont il jaillit. Peut-on alors imaginer ce que serait l'inverse d'un tel saut fractal qui mènerait d'un substantif, comme nous l'avons vu pour gouvernance, au verbe terre à terre qui décrit platement l'action correspondante, quand ce verbe justement n'existe pas dans la langue et doit être remplacé par une périphrase. Comme c'est le cas pour " intermitter ". Doit-on en déduire que le manque de verbe support enlève toute richesse et tout pouvoir évocateur à la terminaison correspondante en " anse " ? Ou alors, au contraire, ce substantif non maintenu au sol par un verbe vulgaire, est digne d'un essor encore plus éclatant ?

Pourrait-il s'appliquer à d'autres sortes de personnages susceptibles de s'affubler ou au contraire se parer d'une telle qualité ? Il me paraît naturel, voire indispensable de se poser la question. Un Intermittent est, par définition un personnage qui ne surgit que de manière sporadique dans le genre d'activité qui le passionne ou le contraint. Il est apparu clairement que les coulisses du spectacle sont bourrées de gens, semble-t-il extrêmement passionnés qui n'interviennent que de manière aléatoire en fonction des opportunités. Ceci présente l'avantage d'une grande souplesse d'emploi et d'une grande diversité de choix, mais en contrepartie, la sécurité de leurs ressources n'est pas toujours bien assurée.

L'occasion était trop belle, dans le cadre d'une bonne " gouvernance ", de stigmatiser ces gens dont on peut facilement déclarer qu'ils passent plus de temps à lézarder au soleil qu'à vivre les heures sages d'un labeur bien éprouvant. En période de fortes économies il y a des choses insupportables, et le spectacle débilitant de personnes trop souvent inoccupées, triste exemple, est insupportable donc, agir en conséquence est une simple " raffarinance ".

Mais, au fait notre définition s'applique-t-elle aux seuls " théâtreux " ? N'existe-t-il pas d'autres activités qui ressembleraient un peu à ces activités du spectacle et dont les intervenants agiraient pendant des périodes discrètes, alors que leurs prestations ne le sont pas forcément ... Suivez mon regard ...

Qu'est ce qu'un maire, un conseiller, un député, un ministre même ? Sinon un personnage ballotté au gré des tempêtes politiques. Qui surgit, ça et là, fait quelques tours, quelques discours, quelques contorsions et puis s'en va balayé par les vents de l'opinion publique.

Les politiciens ne seraient-ils donc, eux aussi, que des Intermittents ? Et également Intermittents du spectacle qu'ils nous donnent dès que la moindre occasion s'en présente aux étranges lucarnes ? Spectacle dont je ne commenterai pas la qualité, vous avez certainement une opinion là dessus. A propos, puisque vous réfléchissez aussi à ce problème, est-ce que vous connaîtriez la différence entre un ministre victime d'un remaniement ministériel, et un Intermittent qui ne trouve plus de spectacle pour faire valoir ses qualités ?

- Comment ? Qu'est-ce que vous dites ? Où est la chute, la conclusion ?
- Pourquoi voulez-vous chaque fois une chute ? Vous la croyez finie, vous, l'affaire des Intermittents ?

Mais puisque vous insistez, voilà la chanson de la " raffarinance " :

Vivons la grandeur de la France,
Soumise à bonne " gouvernance "
Elégance en " communicance "
Faisons à Ernest allégeance.

Une informatique alternative

Jean - Michel Knippel

knippel@up.univ-mrs.fr

En 1998, je recevais une importante documentation de Carlos Derbez proposant à la France de construire une informatique alternative conviviale faite de l'ordinateur adaptable et de l'algèbre historique. Cette informatique alternative est aussi scientifique que celle en usage de nos jours, mais les buts à atteindre ne sont pas les mêmes. Par rapport à l'informatique habituelle, l'informatique proposée est facile à programmer et plus universelle et rigoureuse. Elle rendrait l'informatique par la programmation accessible à tous.

Les réalisations sont restées dans les tiroirs de BULL qui n'a pas fourni les " 12 années-homme " pour sa simulation. Le coût des circuits de l'ordinateur adaptable n'a pas été évalué. Afin d'éviter que ces travaux tombent dans l'oubli, je les ai présentés à mes collègues Edmond Bianco et Bernard Goossens. Nous publions ici trois parties introductives : une note de conférence sur l'architecture et informatique alternative; une présentation de la gamme des ordinateurs adaptables de 1990 est ensuite donnée. Nous terminerons par un résumé technique de la machine molle.

J'ai tout de suite pensé à rapprocher ces travaux de notre tâche, depuis plus de vingt années, qui consiste à montrer les articulations des différentes parties qui composent un ensemble informatique complet, qui va du matériel au service de l'utilisateur. Les différentes liaisons doivent être harmonieuses du matériel à la tâche de l'utilisateur en passant par systèmes et compilateurs. Nos deux réalisations ont vu le jour en 1981 "MCA-0" (voir le numéro 0 de mars 1981 de notre bulletin) et 1994 "PROC-1" (voir le numéro 53 de juin 1999 de notre revue).

Dans la " machine molle ", la longueur d'un mot peut varier. C'est ce qui crée son adaptabilité, les règles de connectivité des circuits logiques n'étant plus restreintes par cette condition limitante. L'utilisateur peut alors se servir des circuits logiques de l'ordinateur selon ses propres besoins, sans être " contraint " par un langage de programmation particulier (*).

Les principes de cette approche alternative sont à lier aux travaux de mes collègues Jean - Philippe Lehmann et Patrick Isoardi qui ont proposé et réalisé une nouvelle structure de mémoires adressables par une fenêtre à déplacement continu pour pallier à l'absence de souplesse que détermine la structure cloisonnée des mémoires (lire le numéro 9 de décembre 1984 de notre périodique).

Dans les deux pages suivantes, nous donnons un raccourci historique de la carrière des deux auteurs et quelques lectures complémentaires, que nous essaierons de publier, telles que " du besoin informatique : vers la conception unifiée " et " la machine ".

Carlos Derbez et André Richard

André Richard, né en 1922, fit ses études à SUPELEC. En 1959 il travailla à la Société d'Electronique et d'Automatisme (SEA), où il réalisa la conception, puis la commercialisation de l'ordinateur CAB 500, vendu à plus de 100 exemplaires en France et au Japon.

Carlos Derbez, né en 1937 et français de l'étranger, fit ses études en électronique à McGILL, Montréal, Canada, où il travailla à la Nothern Electric, filiale de la BELL TELEPHONE. En 1960, il étudia l'informatique à l'IMAG, Grenoble, France.
En 1962, Derbez fit la connaissance de Richard à la SEA, alors qu'il automatisait le chargement de hauts fourneaux d'USINOR à Dunkerque.

En 1967, De Gaulle lança le " plan calcul " ou le plan pour une informatique française. Il décida la fusion de la SEA et de la CAE (Compagnie d'Automatisme Electronique) pour former la CII (la Compagnie Internationale pour l'Informatique). La SEA proposa alors la CAB 1500, successeur de la CAB 500, et la CAE proposa IRIS 50, un ordinateur dérivé du IBM 360. La direction technique de la CII choisit IRIS 50 qui fut construit et commercialisé par ses soins.

C'est alors que naquit la machine molle ou ordinateur adaptable. En effet, les machine proposées partaient de deux mémoires différentes : 24 bits et 32 bits respectivement pour bâtir les langages d'applications. Selon Derbez, elles étaient construites " à l'envers ". Il se demanda s'il était possible, à partir des langages d'application d'arriver à définir les configurations de la mémoire de l'ordinateur d'exécution. Ce fut donc la définition de la mémoire adaptable. Richard accepta cette idée, et petit à petit, pendant leur temps libre, ils bâtirent l'ordinateur adaptable jusqu'à 1984.Toutefois l'ordinateur adaptable était incomplet. Il était incapable de prendre en charge le temps réel, même si l'on adoptait la solution classique mise en œuvre chez Bull (désormais l'employer depuis 1977), les processus coopérants, comme ceux programmés par Derbez en 1990 dans les messageries tels Minicom 3612 du Minitel. Cette solution, adopté dans les serveurs UNIX, le rendait aussi inapte au temps réel !

Entre temps, Richard réfléchissait pour résoudre les problèmes temps réel posés aux ordinateurs. Il décida alors d'écarter la méthode classique des processus coopérants, difficiles à concevoir et à mettre en œuvre sans erreur. Ce fut la naissance de l'algèbre historique et de la manière de l'implanter dans les ordinateurs (dans l'ordinateur adaptable on l'implante au moyen d'un précompilateur).
En 1992 donc, nous avons proposé à BULL les résultats de nos recherches. Faute d'argent, BULL décida de ne rien tenter et nos travaux firent proposés par nous à l'ANVAR sans aucun résultat. Or, la France n'a conçu ni réalisé aucun ordinateur universel après BULL DPS7 vers 1975.
La micro-informatique, d'origine américaine, naquit en 1986. Fille de la grande informatique, elle améliora, grâce à son petit prix et à son petit volume, l'accès à l'ordinateur, mais les difficultés liés à la programmation demeurèrent. De nos jours, ces ordinateurs, si puissants, font en réalité bien peu de choses par rapport a leur possibilités et presque toujours sous la houlette des informaticiens.

Quelques lectures complémentaires

Sous la direction de Marcel Blanc
(*) *L'état des sciences et des techniques*
La découverte - Maspero / Boréal Express. 1983

Marie - Christine Blanc
Présentation du système adaptable
Revue Terminal 19/84, n°9, 1982

Marie - Christine Blanc
Le péché originel de notre informatique quotidienne
Revue Terminal 19/84, n°9, 1982

Derbez Carlos
Arbre de mémoire
1984

Derbez Carlos
Ordinateurs adaptables & mémoires adaptables
1985

Derbez Carlos
La machine
1990

Derbez Carlos
Le projet "liberté"
1990

Derbez Carlos, André Richard
Recherche en informatique
1992

Derbez Carlos, François H. Raymond, André Richard
Du besoin informatique : vers la conception unifiée
1996

Derbez Carlos, André Richard
Une informatique alternative conviviale
1997

Guy Lacroix
Informatique et contrôle social
Revue Terminal 19/84, n°9, 1982

Andréas Suchard
Comment la machine adaptable augmente les possibilités d'expression
Revue Terminal 19/84, n°9, 1982

Architecture et informatique alternative

(Notes de conférence)

André Richard

Je participe à l'élaboration d'une revue qui s'appelle TERMINAL 19/84, que je peux vous transmettre pour que vous la regardiez. Cette revue est produite par une association 1901, qui se préoccupe de tous les impacts de l'informatisation dans la société. Nous constituons une équipe pluridisciplinaire qui étudie l'impact des technologies nouvelles dans la société.

Je vais parler de ceci; je crois que la première chose, qu'on doit regarder, dans la façon dans laquelle les hommes vivent, se rattache à leur activité et c'est l'activité, qu'ils font, qui détermine leur organisation sociale d'une part et d'autre part la ville, les villages, la campagne etc... Or, le fait prenant de notre époque c'est la constitution des énormes sociétés, des multinationales comme on dit, que ce soit Thomson, IBM, General Motors ou des énormes états comme les Etats-Unis, le Brésil etc...

Alors ces énormes entreprises ont un besoin pressant, c'est qu'elles se font la concurrence les unes aux autres et elles ont une tendance : toujours grossir. En France, on arrive presque au maximum à l'heure actuelle avec la nationalisation faite par l'état, de toutes les grandes entreprises françaises et je ne dirai pas qu'elles forment une seule entreprise mais c'est presque ça. Alors, il y a quand même un problème quand un manager arrive à avoir, disons, 600 000 personnes sous ses ordres. Il faut pouvoir commander cette entreprise, ce bataillon de personnes. Pour cela naturellement, l'instrument idéal qui est arrivé après la guerre, c'est l'ordinateur. L'ordinateur inséré dans les organisations sociales permet de collecter l'information, prendre des décisions, gérer la ville, gérer les feux rouges avec plus ou moins de succès, comme tout le monde le sait, mais c'est l'instrument fondamental. Je voudrais quand même faire un rapport avec le passé. On a l'impression que la technologie est quelque chose à part de l'homme mais en réalité, c'est l'homme qui construit la technologie et, à son tour la technologie nous construit. Je vais vous donner un exemple qui est bien connu, surtout ici. A Paris, au 18ème siècle, les bâtiments étaient construits avec environ six étages. Pourquoi ? Parce qu'à l'époque la technique utilisée pour monter les étages était l'escalier, et monter plus de six étages pose des problèmes. La stratification sociale à cette époque était en gros comme ceci : le premier étage c'était l'étage noble, pour arriver jusqu'aux serviteurs, en gros. Bon, c'est schématique, si vous voulez. Naturellement il y avait les quartiers, les arrondissements où la vie sociale se mélangeait à l'intérieur. Ensuite, il y a eu l'apparition de l'ascenseur et de l'automobile. Alors, qu'est ce qu'il s'est passé ? Avec l'ascenseur. Il n'y a plus naturellement de problème pour monter les six étages. On peut faire des immeubles plus hauts. Même les anciens immeubles de

six étages, on les dote d'ascenseurs. Les gens qui habitent actuellement ces quartiers évitent littéralement toutes les basses couches sociales qui se motorisent pour habiter en banlieue à 30 kilomètres de là. Disons ce qui c'est passé réellement : la stratification sociale était à la verticale, et avec la voiture et l'ascenseur on l'a couchée.
Je donne cet exemple pour montrer que la technologie n'est pas indépendante ni de l'organisation sociale ni de la façon de vivre. Naturellement par exemple, les grands buildings comme ceux de la Défense...

Question - Vous ne caricaturez pas un petit peu quand même ?

Réponse - Je crois que dans un grand building comme à la Défense, sans ascenseur vous n'irez pas loin ! Il faut des ascenseurs pour les grands buildings. Et le jour où il y a une panne d'ascenseur, ce sera une véritable panne...

Bon, je vais continuer. Ce qui est remarquable dans toute cette affaire c'est que l' on a produit une certaine technologie et cette technologie n'a jamais été quelque chose d'appropriable par le public dans le sens large. C'est bien connu, par exemple, une usine comme Renault ou d'autres a des chaînes de production déterminées par le management utilisant la méthode bien connue qu'on appelle le Taylorisme. C'est-à-dire qu'il y a une division scientifique du travail qui assigne à chaque personne une tâche particulière à accomplir. Alors, il y a quand même un autre problème. Ces immenses usines, si vous voyez Renault-Flins par exemple, concentrent énormément de personnel dans un seul endroit. A ce moment-là, il faut quand même loger le personnel, il faut qu'il vienne travailler, il faut qu'il y ait des écoles... Alors, probablement une des choses qui est arrivée c'est qu'une fois qu'on a construit le lieu de travail avec une énorme densité, on est obligé de construire pour l'habitat des lieux concentrés et l'on connaît des banlieues qui sont de véritables « parcs à huîtres ». L'organisation sociale continue donc.
Une usine comme Renault n'est pas appropriable parce que si jamais quelqu'un décidait de construire les voitures autrement, c'est impossible. Si le personnel souhaitait un autre type de voiture, c'est rigoureusement impossible parce que la division du travail l'empêcherait. Alors beaucoup de personnes croient que la technologie est par nature non appropriable ; cela est totalement faux. La technologie, à une certaine époque, a été appropriable, avec plus ou moins de difficultés, mais dans certaines civilisations il y avait des actes que l'on pouvait faire : la cuisine par exemple a toujours été appropriable et elle continue à l'être à l'heure actuelle. Tout le monde sait plus ou moins faire la cuisine. Maintenant, plus près de nous dans les immeubles, quand vous faites un parquet, on peut le fait venir ; ça c'est une technologie qui n'est pratiquement pas appropriable parce que si jamais il y a un incident qui se passe, la personne qui habite son logement n'est pas capable de réparer seule son parquet. Par contre, si le parquet est ciré, vous pouvez toujours le réparer. Vous me direz naturellement quelle barbe, c'est vrai, mais vous pouvez toujours le faire. Tandis que dans le 1er cas, c'est rigoureusement impossible. Il faut faire venir l'entreprise, etc... La voiture c'est un peu semblable. A une époque, pour les deux chevaux en France...

Bon, je ne dirais pas que tout était réparable par tout le monde, mais beaucoup de choses pouvaient être réparées. Actuellement, si vous essayez de réparer une deux chevaux vous n'arriverez pas parce qu'il vous manque une clef telle, ou le cran tel... On va donc chez le garagiste pour changer la moindre vis.

Ce que nous appelons une informatique alternative, c'est un effort que nous faisons pour essayer de construire un ordinateur appropriable par tout le monde. Ce que veut dire que tout le monde arriverait à le programmer.
Actuellement, pour les ordinateurs, ça ne se passe pas exactement comme pour la voiture. Tout le monde peut conduire une voiture ; dans les ordinateurs, c'est pratiquement l'apanage des ingénieurs informaticiens et de quelques bricoleurs de génie mais, dans l'ensemble, c'est assez difficile de s'approprier un ordinateur, je parle pour le public en général. Même le public cultivé comme vous, architectes, si vous essayez d'écrire un programme d'ordinateur, vous allez avoir beaucoup de problèmes. Ce n'est pas si évident que ça. Alors, ce que nous pensons, c'est que si jamais on arrivait à rendre l'ordinateur accessible à tout le monde, il est probable que le schéma du Taylorisme ne serait plus nécessaire. Ce que je veux dire par là, ce qui nous paraît possible du point de vue technique, des entreprises de petite taille pourraient exister et être compétitives pour créer et vendre leurs produits sans être obligées de passer par des organisations mégalomanes comme à l'heure actuelle. Donc la thèse fondamentale au niveau social ce sont des petites entreprises, quand je dis « petite » c'est entre environ 10 et 500 personnes. Certains pourraient croire qu'avec 500 personnes on ne peut rien faire. A une époque, j'ai travaillé dans une entreprise informatique qui s'appelait la S.E.A.. Il y avait 600 personnes en tout et, à l'époque, on fabriquait des ordinateurs dont le produit principal était la C.A.B 500 qui s'est vendue à 600 exemplaires en France. Bon, c'est un peu vieux, mais ce que je veux dire, c'est que malgré qu'I.B.M était énorme on pouvait quand même produire des ordinateurs et en vendre. Dans notre organisation sociale actuelle, cela nous est impossible ; ça devient un problème d'état, de politique industrielle, dans l'ensemble de la France. Si jamais les entreprises étaient petites, il pourrait y avoir une décentralisation pour commencer de la production et naturellement de la conception des produits.
Par exemple, quand vous voyez les voitures que se produisent à l'heure actuelle par l'ensemble des constructeurs, toutes se ressemblent. On a du mal à distinguer les unes des autres. Ca se passe un peu comme dans l'architecture : on a du mal à distinguer un immeuble d'un autre. Bien sûr, il y a la marque Renault ou Peugeot mais c'est différent de ce qui existait au départ de la construction des véhicules, disons en 1925, dans laquelle les caractéristiques de construction étaient quand même assez différentes.

Si les entreprises sont petites, la distribution géographique de l'habitat et du bureau n'a plus besoin d'être dans les grandes villes ; elle pourrait être dispersée à la campagne ; probablement le rêve d'Alphonse Allais pourrait devenir vrai : construire la ville à la campagne. Pour l'instant, c'est absolument impensable et, d'ailleurs, cette histoire d'avoir tout concentré, que ce soit à Paris, à Mexico, à Moscou ou ailleurs provient du fait que l'organisation du travail exige des grands ensembles qui sont tous centralisés. Donc, forcément, on est tous à Paris

comme des sardines, d'où les problèmes des architectes évoqués tout à l'heure : comment rendre beau, la densité etc... Ca découle directement des problèmes de production.

Un autre aspect très important, c'est que si l'ordinateur était commandé par tout le monde il permettrait la créativité individuelle, parce qu'aujourd'hui, en dehors de la cuisine je vois mal ce que l'on peut faire pour créer quelque chose. A la cuisine, on peut encore se surpasser. C'est vrai, mais en dehors de ce domaine, la peinture par exemple, c'est assez difficile ; la musique aussi ; prenez n'importe quoi : la mécanique n'en parlons pas.
Certains bricoleurs du dimanche font des choses formidables, il y en a quelques uns qui arrivent, mais c'est assez difficile finalement. L'ordinateur est un outil assez fantastique à ce niveau-là. Une fois qu'on arrive à le commander, on peut imaginer mettre dans un programme ce que l'on souhaite, on peut voir les choses sans les construire, on peut corriger les erreurs, on peut faire un produit fini et même le tester en simulation. On peut faire beaucoup de choses.

Mais pour que cette activité puisse exister, il faut que l'ordinateur descende au niveau individuel, c'est-à-dire que les personnes, le public en général, puisse lui-même laisser libre son imagination pour essayer de construire ce qu'il veut. Alors, à quoi se ramène tout ça ? Comme l'ordinateur pénètre un peu partout.... Il y a la télématique et l'ordinateur lui-même comme instrument de production. Certains pourraient penser qu'une couturière avec une machine à coudre qui est parfaitement décentralisée pourrait, à l'heure actuelle, faire face à une usine de fabrication de pantalons. C'est totalement faux car les machines à coudre que l'on donne aux couturières leur permettent de faire des petites bricoles mais jamais de faire concurrence à quelqu'un disposant d'un moyen technique plus évolué. Quand je parle de décentralisation, je veux dire de fabriquer des outils qui soient concurrentiels vis à vis des autres, de manière à ce que l'initiative individuelle puisse être valable économiquement vis à vis des coûts des grandes organisations, tel que ce soit valable pour un bricoleur d'établir la concurrence à Lee Cooper pour fabriquer des pantalons. Lee Cooper pourrait se trouver alors en difficulté ; je ne dirais pas qu'il disparaîtrait mais ça permettrait de faire la couture de quartier et, à la limite, pourquoi pas la maison de quartier, l'Hôtel de Ville de quartier, A quoi se ramène tout ça ?
A donner une certaine maîtrise de la production à la production et, par conséquence, de tout le reste : la culture, l'organisation de l'espace et de l'habitat, à créer et à inventer par la population elle-même et sans passer par une stricte élite des techniciens qui guident tout le mouvement.

Nous avons sorti un article à ce sujet dans le livre « l'Etat des Sciences et des Techniques, chez Maspero en 1984 ». L'article s'appelle « L'informatique alternative contre le Taylorisme ».

Question - Si j'ai bien compris, votre problème est de libérer l'individu des programmes qu'ils fabriquent en dehors de lui?...

Réponse - Absolument.

Question - Je crois que tout le monde a compris où nous sommes maintenant. On nous propose l'ordinateur, mais nous sommes incapables de faire des programmes.

Réponse - Oui, en gros.

Question - Et il faut que nous allions chercher les programmes chez les spécialistes...

Réponse - Absolument. A ce moment-là vous faites ce que le spécialiste vous dit de faire mais pas ce que vous voulez faire.

Question - Oui évidemment. Vous dites que cela dépend du type d'ordinateur et de sa fabrication. Les ordinateurs qui sont en vente aujourd'hui ne peuvent pas être manipulés généralement par les individuels et il faut que les programmes soient tous faits par les spécialistes. C'est ça ?

Réponse - Oui, c'est ça.

Question - Alors ce que cherchez à faire c'est le contraire. Qu'est ce que votre ordinateur aura par rapport aux autres ? A la limite, il sera plus simple, il sera comme la deux chevaux par rapport à la Jaguar ?

Réponse - Je vais vous dire ce que je cherche. Les modèles d'ordinateurs que l'on construit à l'heure actuelle émanent de celui inventé par Von Neumann aux Etats-Unis en 1945. Tous sont des petites variantes de ce modèle, tous conservent la même architecture de base. Lorsque Von Neumann réalise son ordinateur, il était coûteux, volumineux, il consommait une quantité de courant électrique colossale etc... Maintenant, on a une technologie beaucoup plus souple, miniaturisée, beaucoup plus rapide, mais l'organisation interne de l'ordinateur reste toujours la même, à quelques détails près.

Notre thèse est que cet ordinateur est trop simple, trop simple pour attaquer des problèmes complexes. Il faut alors des spécialistes pour se servir d'un outil trop simple pour attaquer le compliqué. Si la machine elle-même était bien plus complexe de ce que l'on construit à l'heure actuelle, il serait possible que la communication homme-machine soit rendue le plus facile possible. Je vais vous dire quelque chose qui vous fera rire. Nous appelons l'ordinateur classique « l'informatique dure » car c'est un moule qui est fait une fois pour toutes, et pour faire un calcul quelconque il faut que ça rentre dans le moule, ce qui est souvent assez difficile. Quand nous appelons notre informatique « informatique molle » nous voulons dire l'inverse. Autrement dit, lorsque vous commencez à travailler avec cet ordinateur, au fur et à mesure que vous commencez à dialoguer avec lui, il va se modeler et il va vous suivre, il va essayer de comprendre ce que vous dites, essayer de vous faciliter la tâche, et à ce moment-là, les difficultés de communication des programmes seront beaucoup moins grandes. Ça c'est le principe.

Question - Quand vous dites nous, ce que nous proposons, est-ce qu'il y a déjà un secteur de l'industrie qui se propose de mettre demain à la portée de la main des utilisateurs des ordinateurs pareils, ou il y a un horizon probable pour que cela puisse arriver ? Ou c'est seulement la théorie mais ça ne progresse pas, il n'y a pas de réalisation pratique ?

Réponse - Nous avons essayé de présenter notre projet à un certain nombre de responsables : le ministère de la Recherche, des industriels. On n'a jamais réussi à avoir la même réponse parce que nous sommes certainement de très mauvais commerçants.

Question - Vous avez un projet précis, un ordinateur que vous souhaiteriez réaliser ?

Réponse - Oui, mais nous n'avons jamais eu les moyens de travailler. Alors, comme nous sommes persuadés que nous n'obtiendrons jamais un centime, c'est notre hypothèse à l'heure actuelle, nous essayerons de le faire à l'extérieur de notre temps de travail ? Cela dure depuis quinze ans.

Question - Alors vous voulez vendre de la liberté et on ne veut pas vous l'acheter ?

Réponse - Quand vous demandez aux constructeurs, ils vous diront dans l'ensemble « on ne veut pas, car cela pose des problèmes techniques, ou de concurrence, ou de vitesse ». Tout cela est possible, mais peut-être il n'y a pas là aussi la volonté de les résoudre. Le premier problème dans une recherche c'est de cerner le problème que l'on veut résoudre. Si l'on ne cherche pas dans cette voie-là, nous n'arriverons jamais, c'est évident. Je préfère vous laisser à votre jugement les conditions de ce cercle vicieux.

Question - La communication, pour quoi faire ? Si vous voulez donner la liberté, nous marquons un point, car jusqu'à présent on ne voit absolument pas pourquoi faire tout ça, les ordinateurs, les mémoires dont on n'arrive pas à se servir...

Réponse - Je vais vous donner un exemple concret : imaginez vous une personne qui veut travailler et qu'elle sait faire quelque chose (des costumes, usiner une pièce). Elle peut avoir son savoir dans un ordinateur avec laquelle elle commande une machine. Il est parfaitement possible que cette personne ayant son ordinateur puisse enregistrer son savoir dans des disquettes magnétiques. Avec ces disquettes, elle va voir un responsable d'une entreprise qui lui demande ce qu'elle sait faire. Elle répond : des axes de roues de voitures. Faites-moi une démonstration, donc prêtez-moi votre ordinateur. Je mets la disquette, on vous embauche, on fabrique des voitures. A un moment donné, on me dit que je ne fais plus l'affaire ; je reprends ma disquette et que le prochain se débrouille ; je m'en vais ailleurs avec ma disquette. Et je recommence avec un autre ordinateur, mais cela exige la compatibilité, c'est-à-dire que tous les programmes puissent

passer n'importe où et à ce moment-là, avec votre disquette, vous pourriez faire de l'informatique n'importe où, dans votre ville, dans votre quartier, dans votre maison, dans votre entreprise, où vous voulez. C'est votre propriété privée, vous l'exploiterez pour votre profit, c'est-à-dire vous allez déterminer votre travail, vous vendez un produit, vous négociez avec une entreprise ou vous vendez directement, cela n'a pas grande importance à la limite...

Question - De toute façon, je pense que cette proposition de Monsieur Derbez permet d'envisager une informatique différente, alternative, c'est-à-dire une informatique qui permettrait d'envisager d'autres rapports entre l'informatique, l'urbanisme et l'architecture. Enfin, c'est finalement notre sujet, d'autres rapports différents de ce que nous ont été proposés jusqu'à maintenant même dans notre colloque, même s'il ne s'agit que de propositions pour l'instant. Je pense que ce rapport entre l'informatique, l'architecture et l'urbanisme nous permet d'envisager un horizon plus confortable, moins étouffant que celui que nous avions envisagé jusqu'à ce moment-là qui, pour moi personnellement m'était désagréable d'envisager comme futur proche ou lointain. Je crois que nous avons tous peur de l'informatique parce que l'informatique nous contrôle et parce que nous avons la sensation d'être complètement contrôlés et que nous ne pouvons pas la contrôler. Ce que vous proposez c'est justement d'avoir une voie alternative qui permettrait de contrôler l'informatique comme instrument de travail, ce serait magnifique, malheureusement, si ça ne passe pas dans l'industrie, on est un peu coincé. Peut-être on sait qu'il y a ces possibilités mais ce n'est pas réel, c'est un peu utopique.

Réponse - Il se peut bien que notre tentative ne réussisse pas. Mais pour nous, il y a deux problèmes. Le premier c'est poser la façon de sortir du type du monde que vous et bien d'autres ne souhaitent pas. Cela, nous le faisons. Une fois le problème posé, il sera connu, et si nous ne réussissons pas, d'autres réussiront une fois qu'ils seront engagés dans cette voie. L'humanité n'a jamais raté aucun changement de cap, mais c'est parfois très long. Sur le plan théorique, aucune technologie n'est contraignante. C'est la manière avec laquelle on l'intègre socialement qui est contraignante. Cela devient très compliqué une fois que cette technologie est insérée dans l'organisation sociale. Vous l'apprendrez ainsi même dans les livres et alors ça devient un cercle vicieux. Mais il y a toujours des alternatives. Jadis la terre était plate et au centre de l'univers, jusqu'à ce que Galilée la mis à tourner autour du soleil. Ca n'a pas fait plaisir à tout le monde, surtout aux ecclésiastiques d'alors. Mais il y a toujours des alternatives scientifiques, il faut les chercher, il faut les trouver et les appliquer. C'est comme ça que cela se passe.

Question - Donc ça nous laisse dans l'espoir optimiste d'avoir un futur meilleur et pas totalitaire. Par quel moyen... ?

Réponse - Par tous les moyens...

Présentation de la gamme des ordinateurs adaptables

Carlos Derbez et André Richard

I. Objectifs généraux

Quatre objectifs généraux ont guidé les choix techniques dans la conception des ordinateurs adaptables :
- Compatibilité.
- Convivialité.
- Modularité.
- Parallélisme.

Ces objectifs sont aujourd'hui atteints par l'implémentation prévue des propriétés qui suivent.

II. Présentation des propriétés inhabituelles

La première caractéristique de l'ordinateur adaptable est que la longueur du mot n'est pas fixée une fois pour toutes, mais peut varier de 1 à 255 octets. Il en résulte :
- que la capacité d'adressage est « illimitée »;
- que l'on peut y inclure des processus capables d'opérer d'un coup sur des nombre de 1 à 255 chiffres ou sur des mots de 1 à 255 caractères.

Pour plus de convivialité, les nombres sont d'un seul type. Leur déclaration dans les programmes s'en trouve simplifiée, et leur transmission à des sous-programmes ne présente jamais de difficultés.

Bien que l'ordinateur soit prévu à un fonctionnement séquentiel, il comporte des dispositifs permettant la programmation non procédurale. Les compilateurs pourront ainsi organiser la mise en œuvre de processeurs multiples. De genre déclaratif, les programmes non procéduraux définissent néanmoins des calculs à effectuer purement et simplement le moment venu, et non des relations à explorer tant bien que mal comme en intelligence artificielle.

Pour tenir compte des événements périphériques, il n'y a pas lieu de revenir à la programmation procédurale. L'algèbre historique (Voir "Résumé technique de la machine molle et vocabulaire") pourvoit aux besoins les plus complexes de synchronisation des calculs, sans qu'il soit nécessaire de définir des tâches à la manière habituelle, ni de recourir aux sémaphores classiques. Les commodités nécessaires sont prévues dans l'ordinateur adaptable, qui est bâti sur des concepts nouveaux. Il constitue une étape dans l'évolution de l'ordinateur classique sans pour autant prétendre remplacer les logiciels de l'I.A., qui n'est qu'une importante application.

Le langage de programmation admet la compilation en modules séparés. Les tableaux sont de dimension variable et les types peuvent n'être connu qu'à l'exécution, si bien que les programmes génériques se compilent une seule fois. La récursivité, la réentrance, la « relocabilité », la manipulation des pointeurs et la transmission des procédures comme paramètres sont prévues. Le langage de programmation offre quelques facilités de calcul non procédural. Il permet aussi de réaliser des fonctions globales complexes sur des tableaux et le contrôle de leurs limites. Il permet l'écriture ou l'appel des procédures d'exception, tant du matériel que du logiciel.

Les ordinateurs adaptables sont tous modulaires dans leur construction et ils sont compatibles, aussi bien dans le sens ascendant que dans le sens descendant.

Par modulaire, nous entendons que l'utilisateur peut à tout moment acheter et intégrer davantage de mémoire et de processeurs à sa machine. Si l'utilisateur le fait, c'est pour incorporer davantage de données dans ses programmes ou pour augmenter leur taille et leur nombre ou pour activer davantage de tâches parallèles ou de modules de programme. Ou bien, tout simplement pour accélérer l'exécution par l'attaque en parallèle par plusieurs processeurs de son programme synchronisé, et tout cela sans limite pratique préétablie et sans reprogrammation. Le micro-ordinateur adaptable ne diffère du mégaordinateur adaptable que par la vitesse de ses circuits, qui admettent plusieurs sortes de réalisations. En particulier, le microprocesseur pourrait avoir une partie de ses circuits émulés, pour diminuer son prix de réalisation.

Par compatible, nous entendons que tout programme ayant des ressources suffisantes, peut s'exécuter indifféremment dans un quelconque ordinateur de la gamme, du méga-ordinateur au micro-ordinateur. Cette caractéristique est essentielle si BULL désire faire descendre les prix de ses solutions pour mieux vendre.

Le système d'exploitation sera écrit dans le même langage que celui de l'utilisateur, à quelques instructions privilégiées près.

L'absence de coupure entre les programmes de l'utilisateur et ceux du système d'exploitation rompt la tradition informatique en a matière et facilite la gestion de l'ordinateur. Celui-ci ne connaît ni les « Macrocalls système », ni les « Link », ni les « Load », ni les sémaphores des programmes. Le système d'exploitation est tout petit, sauf pour la gestion des fichiers et l'interprétation du langage de commandes.

Nous avons des idées précises sur les caractéristiques que nous voulons obtenir du système de gestion des fichiers. Celles-ci ont été définis. Cependant, nous considérons ce système comme une application particulière et elle n'est qu'une préoccupation mineure de notre projet.

Pour les applications de l'intelligence artificielle, le langage de programmation offre toutes les ressources nécessaires à la réalisation des divers interprètes en usage, tels celui du PROLOG ou ceux des systèmes experts, car l'I.A. c'est de la programmation au 2ème degré. LISP n'a pas fait l'objet de notre étude, mais il est probable que ses implémentations classiques seront acceptées avec des avantages.

Les performances espérées sont un peu inférieures à celles des ordinateurs classiques, mais ceux-ci sont le plus souvent très spécialisés et trop limités. Dans le cas général, leur compatibilité est aléatoire et leurs possibilités sont restreintes, compliquant inutilement la programmation des applications à vendre chez nos clients. Pour mémoire, rappelons qu'une application écrite en Pascal ou en C est 2 fois plus lente que si elle était écrite en assembleur. Malgré cela, plus personne ne programme en assembleur, sauf exceptionnellement.

III. En quoi les ordinateurs adaptables sont-ils universels ?

Actuellement, il existe sur le marché :

a) Des langages plus ou moins compatibles.
b) Des ordinateurs plus ou moins modulaires
c) Des langages procédures plus ou moins restrictifs.
d) Des systèmes d'exploitation multitâches.
e) Des systèmes d'exploitation multiprocesseurs.
f) Des systèmes d'exploitation multiprocessus.
g) Des langages de gestion.
h) Des langages Temps Réel.
i) Des langages scientifiques à double ou triple précision.
j) Des langages traitant globalement des tableaux.
k) Des méga-ordinateurs.
l) Des micro-ordinateurs.

En général, l'utilisateur cherche un ordinateur possédant une ou plusieurs des qualités allant de a) à 1). L'ordinateur, ou système ou langage qui réunit ces qualités n'existe pas aujourd'hui. Cette lacune est comblée dans la conception de l'ordinateur adaptable, car celui-ci n'est plus un ordinateur spécialisé, comme le sont ceux du marché. Créer des ordinateurs qui remplissent quelques uns de ces critères est le pain quotidien des constructeurs. Les ordinateurs adaptables sont universels vis à vis des besoins informatiques existant aujourd'hui, et une extensibilité existe pour demain. Nous avons cherché à avoir une conception cohérente et générale qui permette de satisfaire la plupart des besoins informatiques. Universalité, cohérence et généralité de conception ! Là réside leur nouveauté.

IV. Quels sont les domaines d'application des ordinateurs adaptables ?

Tous les domaines de l'informatique :

- Temps réel.
- Gestion.
- Calcul Scientifique de Haute Précision.
- Commande des Processus Industriels.
- Télécommunications.
- I.A par interprétation (comme d'habitude).
- Base de Données.
- Bureautique.
- Robotique.
- C.A.O..
- Multiprogrammation.
- Parallélisme.
- Multiprocessing.
- Etc...

V. Où en est le projet de l'ordinateur adaptable ?

Aujourd'hui, nous connaissons :

- La structure matérielle des ordinateurs adaptables.
- La méthode générale de synchronisation en parallélisme.
- La façon de compiler son langage de programmation.

Avec ces connaissances, nous pouvons réaliser un prototype en logiciel sur nos micro-ordinateurs BULL 90-20 ou sur tout autre ordinateur.

A cet effet, nous avons réalisé sur BULL 90-20 l'analyseur syntaxique du compilateur.

Nous nous préparons à réaliser :

- le compilateur
- la simulation des circuits de l'ordinateur adaptable sur BULL 90-20.

VI. Quand verrons nous le premier ordinateur adaptable ?

Si continuons à travailler hors du temps normal de travail, comme nous l'avons fait pendant 22 ans, cela serait trop long (9 ans ?) à moins de trouver un financement qui permettrait de réaliser rapidement (Pour qui ?) ; le financer et le réaliser par nos propres moyens est une possibilité qui n'est pas à exclure. Nous vendrons alors un logiciel au lieu de construire un ordinateur.

Si BULL, sortant de son indifférence, fait sien ce projet, le délai pourra diminuer beaucoup. Si je travaille tout seul et à temps complet avec la seule assistance de Monsieur André Richard, retraité Bull, il me faudra 4 ans pour réaliser un prototype minimal, nu et non commercial. Ce prototype nu exclura le système de gestion des fichiers, les programmes habituels de base tels le calcul de sinus (X), et la programmation de toutes les fonctions nécessaires, telle le calcul en virgule flottante. A deux ingénieurs maximum, il faudra compter 2 ans et demi pour réaliser le prototype nu. Nous nous intéressons à l'ordinateur, bien plus qu'à son habillage. Réaliser l'habillage pour le rendre commercial exige l'effort de lancement de tout nouvel ordinateur, avec l'avantage cette fois-ci de le faire valablement pour toute une gamme. C'est construire sur des bases solides.

Nous pensons toujours que ce projet est important pour un avenir possible de BULL. L'avenir n'est pas écrit. Il est construit tous les jours par nos actes. Créer un avenir informatique possible pour la maison fait partie de nos responsabilités d'ingénieurs chez notre constructeur national d'informatique.

Remarque : Si éventuellement BULL souhaitait utiliser l'algèbre historique pour ses projets, il faudrait la marier avec un langage existant de programmation. Ceci n'est pas impossible, mais réaliser une telle adaptation demanderait autant d'efforts que créer l'ordinateur adaptable. Je reste donc fidèle au projet de réaliser l'ordinateur adaptable avec l'algèbre historique.

VII. Exemple concret d'utilisation des ordinateurs adaptables

EXEMPLE DU LIBRAIRE.

Un libraire a répertorié ses livres par nom d'auteur, chaque auteur pouvant avoir plusieurs titres. Tous ses livres se trouvent dans le fichier Répertoire, qui est un énorme fichier non « locked ».Le libraire a un vendeur : Paul. Lorsque un client demande à Paul un livre, Paul le cherche et s'il le trouve il le vend, le retire du répertoire, met à jour la comptabilité et édite l'état de la vente.

Le travail du libraire consiste à traduire tous les titres des livres du français à l'anglais en même temps que Paul cherche à vendre les livres. La traduction et la vente de deux livres différents a lieu simultanément. Si un livre est en cours de traduction, il ne doit pas être vendu. S'il est en instance de vente, la traduction est d'abord différée. Elle se fait ensuite seulement si le livre n'est pas vendu. La vente des livres est prioritaire vis à vis de leur traduction.

Pour faire cela, ils sont bien équipés. Leur micro-ordinateur adaptable BULL a été doté pour leurs besoins de trois processeurs. Le libraire et Paul ont écrit et compilé leurs programmes séparément. Leurs modules Traduire et Client ont été écrits, dans la mesure du possible, dans la notation Pascal, car il n'est pas possible d'improviser un langage non procédural.

```
   MODULE TRADUIRE ;

VAR
  H :T_ATTACHE_MODULE ;
FUNCTION LIBRAIRE ; (*Inutile d'indiquer le type, l'ordinateur le connaît*)

CONST
  PAS_DE_LIVRE=T_LIVRE(' ', [1000] ' ') ) ;
                  (* Constante structurée mise à blanc*)
TYPE T_LIVRE=RECORD
         AUTEUR :CHAR 100 ; (* AUTEUR est un mot de 100 caractères *)
         TITRE :ARRAY [1..1000] OF CHAR ;
          END ;

TYPE T_ETAT_FICHIER=RECORD
EoF : BOOLEAN;    (* Indique la fin d'un fichier *)
ERR : NOMBRE;     (* Indique le numéro d'erreur de E/S *)
FICHIER : BOOLEAN; (*Indique si l'on est positionné sur la tête d'un fichier*)
PCF : BINAIRE 64 LEI; (* Contient le point courant dans le Système de
                         Gestion des Fichiers Adaptables. LEI spécifie que la
                         lecture et l'écriture de cette mémoire sont interdites
                         à l'utilisateur *)
      END ;

TYPE T_B=RECORD
          DEM_VENDRE :  EV ; (* EV signifie type événement *)
          VENDRE :  EV ;
          PAS_VENDRE :  EV ;
          VENDU :  EV ;
          PAS_VENDU :  EV ;
        END ;


 VAR

 LIVRE A VENDRE, LIVRE_A_TRAD : T_LIVRE ;
 MESS : CHAR 30 ;

 F :T_ETAT_FICHIER ,
 BUFFER_DEVICE : ARRAY [1..4096] OF CHAR LEI ;

 E :T_B ;   (* Réservation des événements pour CLIENT*)
```

```
OK : BOOLEAN ;
DEM_TRADUIRE : EV ;
TRAD : EV ;
TRADUIRE : EV ;
PAS_TRADUIRE : EV ;
FIN_TRADUIRE : EV ;

X : T_ATTACHE_TACHE ;

SYNCHRO  (* Equations de l'algèbre historique destinées à la synchronisation
            des tâches parallèles *)

OK=LIVRE_A_TRAD.TITRE=PAS_DE_LIVRE.TITRE
   | LIVRE_A_VENDRE.TITRE=PAS_DE_LIVRE.TITRE

                    (*| veut dire « OU »  * )

   | LIVRE_A_VENDRE<> LIVRE_A_TRAD ;

TRAD=DEM_TRADUIRE & OK & NOT E.DEM_VENDRE

                    (* & signifie « AND » * )

   | DEM_TRADUIRE & NOT TRAD sigma pi (E.VENDU | E. PAS_VENDU) ;
     (* sigma et pi sont deux opérateurs de l'algèbre historique *)

TRADUIRE=TRAD & ( OK| E. PAS_VENDU) ;
PAS_TRADUIRE=TRAD & E.VENDU ;

   E.VENDRE=E.DEM_VENDRE & (OK|DEM_TRADUIRE) ;
   E. PAS_VENDRE=E.DEM_VENDRE & NOT E.VENDRE ;


BEGIN
LIVRE_A_TRAD :=PAS_DE_LIVRE ;
LIBRAIRE := ' TRAVAIL ERRONE' ;
MESS :=' ' ;
CT (X,PAUL(LIVRE_A_VENDRE , MESS,E) ) ; (* Créer la tâche PAUL *)
SEEK (*,F,' REPERTOIRE') ;  (* Chercher le fichier REPEROIRE *)
                    ( * Voir « COMMENTAIRES SUR L'EXEMPLE » *)
DOWN_FILE (* , F) ;          (* Fichier fils d'auteurs *)
                    (* Le fichier séquentiel indexé des auteurs a les fichiers
                    séquentiels des titres comme fichier fils *)
```

```
REPEAT
  READ (*, F,LIVRE_A_TRAD.AUTEUR) ;
  IF  F.EoF   THEN  BEGIN
                    IF MESS <>' '
                        THEN LIBRAIRE :=MESS
                        ELSE LIBRAIRE :='BON TRAVAIL' ;
                    RETURN ;
                   END ;
NEXT (*,F) ;
DOWN_FILE (*,F) ;          (* Fichier fils des titres *)

REPEAT
  READ (DEM_TRADUIRE,F,LIVRE_A_TRAD.TITRE) ;
          (* Envoyer le signal TRADUIRE à la synchronisation *)

QUAND TRADUIRE ; (* QUAND lance un processeur à l'exécution des
                    instructions qui suivent, lorsque TRADUIRE se réalise *)
TRADUIRE_ANGLAIS (LIVRE_A_TRAD.TITRE) ;
WRITE(FIN_TRADUIRE,F,LIVRE_A_TRAD.TITRE) ;

QUAND PAS_TRADUIRE | FIN_TRADUIRE ;
LIVRE A TRAD.TITRE :=PAS_DE_LIVRE.TITRE ;
NEXT(*,F) ; (*Recherche du fichier ou de l'article suivant *)

UNTIL  F.EoF
UP_FILE(*,F) ;     (*   Retour au fichier père *)
NEXT (*, F) ;
UNTIL FALSE
END ;      (* Fin de la fonction LIBRAIRE *)

TACHE PAUL (VAR LIVRE, MESS, E) ;
  (* Inutile de spécifier le type : l'ordinateur le connaît *)

VAR
Y : T_ATTACHE_MODULE.
BEGIN
AM (Y,CLIENT(LIVRE,MESS,E) ;   (* Activation du module CLIENT *)
END ;  (* Fin de la tâche PAUL  *)

BEGIN  (* Instruction du module TRADUIRE *)
AM (H,EDIT (LIBRAIRE) ) ;  (* Impression de la réponse *)
END.     (* Fin du module TRADUIRE et fin de la compilation *)
```

```
MODULE CLIENT (   VAR ART, MESS, E : T_B OF TRADUIRE_) ;
(* Ce module s'exécute en parallèle avec le module TRADUIRE *)
PROCEDURE VENDRE (B_SIZE,TYP) ;

CONST
PAS_DE_LIVRE OF TRADUIRE ;

VAR
F : T_ETAT_FICHIER OF TRADUIRE,
   BUFFER_DEVICE :ARRAY [1..4096] OF.CHAR LEI ;

V : T_ETAT_FICHIER OF TRADUIRE,
   BUFFER_DEVICE :ARRAY [1...B_SIZE] OF.TYP LEI ;
         (* La taille de BUFFER_DEVICE et son type sont spécifiés par les
         paramètres de la procédure VENDRE *)

LIVRE : T_LIVRE OF TRADUIRE ;
M,N : T_ATTACHE_MODULE ;

RECOMMANCER : EV ;
FIN_COMPTABILITE :E V ;
FIN_EDIT_ETATS : EV ;

SYNCHRO

   RECOMMENCER = FIN_COMPTABILITE & NOT RECOMMENCER
                                          sigma pi FIN_EDIT_ETATS
                  |FIN_EDIT_ETATS & NOT RECOMMENCER
                                          sigma pi FIN_COMPTABILITE
                   | FIN_EDIT_ETATS & FIN_COMPTABILITE

BEGIN
   MESS :=' ERREUR DANS LE MODULE CLIENT ' ;
ASSIGN (*, V, 'CONSOLE', DKU,3) ;
         (*  LA 3ème visu du type DKU se trouve en 'CONSOLE' *)

ART := PAS_DE_LIVRE ;
SEEK (*,F,'REPERTOIRE') ;
DOWN_FILE(*,F) ;   (* Fichier fils des auteurs *)
SEEK(*,V,'CONSOLE') ;
DOWN_FILE(*,V) ;

REPEAT
   READ (*,V,LIVRE.AUTEUR) ;
   SEEK(*,F,LIVRE.AUTEUR) ;
   READ(*,F,ART.AUTEUR) ;
   NEXT(*,F) ;
   DOWN_FILE(*,F) ;   (* Fichier fils des titres *)
   READ(*,V,LIVRE.TITRE) ;
```

```
   REPEAT
      READ (E.DEM_VENDRE,F,ART.TITRE) ;

      QUAND E.VENDRE ;
      NEXT(*,F) ;
      IF ART<>LIVRE THEN
                  BEGIN
                  EMETTRE E.PAS_VENDU ;
                  IF F.EoF THEN BEGIN
                              MESS :='LIVRE INEXISTANT' ;
                              RETURN ;
                            END ;

   UNTIL ART=LIVRE

   PREVIOUS(*,F) ;
   DELETE(E.VENDU,F) ;

   QUAND E.VENDU
   AM (M,COMPTABILITE (LIVRE,FIN_COMPTABILITE) ;

   QUAND E.VENDU ;
   AM(N,EDIT_ETATS(LIVRE,FIN_EDIT_ETATS) ;
     (* COMPTABILITE, EDIT_ETATS et TRADUIRE s'exécutent en parallèle *)

   QUAND E.PAS_VENDRE | RECOMMENCER ;
   ART.TITRE. := PA_DE_LIVRE.TITRE ;
   NEXT(*,F) ;
   UP_FILE(*,F) ;
 UNTIL FALSE
END ;    (* Fin de VENDRE  *)

 BEGIN
   VENDRE(80,'CHAR') ;
 END.


 MODULE COMPTABILITE (LIVRE, VAR EVENMENT) ;

 BEGIN


 EMETTRE EVENEMENT ;
   END
```

```
MODULE EDIT_ETATS (LIVRE, VAR EVENEMENT) ;

BEGIN

EMETTRE EVENEMENT ;
  END.


MODULE EDIT (RESULT) ;

BEGIN


END
```

VIII. Commentaire sur l'exemple

Ces commentaires ont pour seul but d'éclairer la signification des programmes précédents.

D'abord, quelques mots sur l'organisation du système de gestion des fichiers adaptables : SGFA

Dans le SGFA, tout article peut contenir, soit des données, soit la tête d'un nouveau fichier fils (séquentiel ou séquentiel indexé). Prenons un court exemple du REPERTOIRE du libraire :

```
REPERTOIRE
                    °
                        BALZAC
                            °          EUGENIE GRANDET
                                       LE PERE GORIOT
                                       LE LYS DANS LA VALLEE
                    °
                    SAINT EXUPERY
                            °          LE PETIT PRINCE
                                       VOL DE NUIT
                                       TERRE DES HOMMES
```

('' ° '' dénote la tête d'un fichier)

En partant de la tête du fichier, la clef « REPERTOIRE » nous amène à un article qui est la tête du fichier fils d'auteurs. En faisant DOWN_FILE sur ce fichier, le pointeur courant d'article se place sur le premier article : « BALZAC ». Si je lis le deuxième article, j'accède à la tête du fichier de tous les titres des livres de BALZAC. Je fais alors DOWN_FILE pour accéder au premier article du fichier des titres des livres de BALZAC. READ me permet de lire cet article :

« EUGENIE GRANDET ». NEXT m'amène à l'article suivant que je lis par READ. C'est le titre « LE PERE GORIOT ». Si je fais READ NEXT maintenant, je lis l'article suivant : « LE LYS DANS LA VALLEE ». PREVIOUS me placerait sur l'article précédent : « LE PERE GORIOT » et si je tente de lire l'article qui suit « LE LYS DANS LA VALLEE », SGFA me répond EoF.TRUE (End of File).

L'interface entre le SGFA et l'utilisateur est faite par la zone des mémoires utilisées comme deuxième paramètre dans les instructions SGFA. Par exemple NEXT (, F) demande au SGFA de se positionner sur l'article suivant et la réponse du SGFA peut être F.EoF=TRUE ou F.EoF=FALSE selon que NEXT trouve la fin des articles du fichier ou encore un autre article. F est la zone des mémoires de l'interface.

Revenons au fichier de notre exemple. Si en lisant le fichier des titres des livres de Balzac nous faisons UP_FILE, le pointeur courant du fichier retourne à la tête de ce fichier. Si je fais alors READ_NEXT, je lis le nom d'auteur suivant « SAINT EXUPERY ». D'une manière analogue à la monnaie à la précédente, j'accède à tous les titres des livres de Saint Exupéry et ainsi de suite à tous les auteurs et à tous les titres des livres. Remarquons que le fichier des auteurs est séquentiel indexé tandis que les fichiers des titres des livres sont séquentiels. A partir d'un positionnement sur la tête de fichier d'auteurs, SEEK (,F, « SAINT EXUPERY ») ; amène directement le pointeur courant du fichier sur l'article contenant « SAINT EXUPERY ».

Ces remarques suffisent pour comprendre les accès au fichier SGFA des modules TRADUIRE et CLIENT.

En plus de la déclaration des types, des constantes et des variables, on définit des événements par des expressions de l'algèbre historique.

Pour la synchronisation, disons que les programmes s'arrêtent par exemple lorsqu'une opération d'Entrée/Sortie n'est pas encore terminée. Une fois que cette opération est terminée, le système envoie les événements DEM_TRADUIRE ou E.DEM_VENDRE par exemple à la synchronisation. Celle-ci répond par les événements TRADUIRE, PAS_TRADUIRE, VENDRE ou PAS_VENDRE. Lorsqu'un ou plusieurs de ces événements sont réalisés, les programmes reprennent immédiatement. Cette reprise a lieu aux QUAND activés par ces derniers événements et qui figurent dans les programmes. L'activation des QUAND simultanés permettent aux tâches de s'exécuter en parallèle et en toute sécurité.

Par ailleurs, les ordres READ, DOWN_FILE, UP_FILE, WRITE, NEXT, PREVIOUS et READ_NEXT s'exécutent de façon asynchrone. Notons par exemple READ (*, ...) ; la combinaison de deux instructions : READ asynchrone et QUAND FIN_READ, qui permet la reprise de la tâche à la fin de la lecture. Notons par READ (« Evénement »,.....) ; l'ordre de lecture explicitant que la reprise du travail n'aura lieu qu'au QUAND utilisant « Événement ». Il va de même pour les autres instructions précitées.

IX. Programmation adaptable et coûts

Nous savons que le coût du logiciel d'une application informatique installée chez le client représente près de 85% du coût total, et que ce coût ne baisse pas, mais au contraire, il tend à augmenter. Si nous voulons être compétitifs, il est logique de chercher à faire diminuer ce coût.

Mais comment ? Dans l'ordinateur adaptable nous le faisons par deux facteurs conjugués :

- par l'utilisation d'opérandes dont la capacité ne semble pas avoir de limites y compris pour l'adressage et dont on peut ajuster la capacité à volonté ;
- par la présentation déclarative, et non pas procédurale, des programmes.

Ces deux facteurs combinés donnent une souplesse remarquable à l'écriture des programmes adaptables. Disons le clairement :
programmer l'ordinateur adaptable est plus simple que programmer les ordinateurs du commerce. C'est aussi plus systématique. Cette simplicité est une conséquence de son universalité.

Dire que c'est plus simple n'est après tout qu'une appréciation, car je pourrais affirmer que c'est plus simple d'aller sur la Côte d'Azur en passant par la Vallée du Rhône plutôt que par les Alpes et être contesté par ceux qui ne connaissent que la route Napoléon.

Donnons quelques exemples informatiques concrets.

Si nous écrivons C := A+B ; ou A, B et C sont des tableaux, nous réalisons plus simplement l'opération que si nous additionnons les éléments un à un. C'est convivial et facile à exprimer.

Si j'écris D := 123456789098765 * 3,1416 ; c'est bien plus simple que d'écrire un programme en triple précision combinant une horrible conversion, car je l'exprime dans une seule instruction. C'est convivial et facile à exprimer.

Si nous calculons des expressions complexes telles les racines d'une équation algébrique par le calcul procédural classique, le programme est plus long, plus difficile à écrire et plus apte à l'erreur que si nous écrivons ce calcul en programmation non procédurale.

Si je fais du parallélisme avec des sémaphores, je ne dispose pas de méthode formelle pour combiner leur action, ce qui rend un parallélisme complexe quasiment impossible à programmer. Par sa méthode formelle, l'algèbre historique de l'ordinateur adaptable permet l'expression des parallélismes complexes.

Que dire des tableaux limités à 60000 éléments seulement en Pascal MS-DOS ou en DPS6 ? Avoir des tableaux sans limites à priori facilite aussi la programmation et

rend possibles et conviviales bien des applications.

Que dire encore de la limitation à une seule tâche de MS-DOS ? Mais ce n'est fait que pour de très nombreuses applications rudimentaires !

L'ordinateur adaptable a d'autres propriétés ignorées des systèmes classiques, qu'il serait trop long d'énumérer. Elles ont toutes le même but : éloigner les limites et rendre plus souple et plus agréable la programmation, grâce à une meilleure expressivité.

Cette expressivité est surtout appréciable lors de l'écriture des applications non banales de l'informatique, comme le montre le court exemple du libraire dans notre note « Présentation de la gamme des ordinateurs adaptables ». Ces ordinateurs permettront par leur expressivité, de repousser le mur de la complexité.

L'expressivité dans l'implémentation des programmes a une conséquence dans leur conception. Ce qui se conçoit clairement s'exprime facilement et vis et versa. La qualité de l'expression dans la programmation facilite la qualité dans la conception du logiciel, car les deux sont intimement liés. S'exprimer plus facilement, c'est concevoir plus facilement, concevoir en moins de temps ou concevoir plus de complexité.

Programmer plus simplement ne peut donc que réduire le coût de réalisation d'une application donnée. De combien ? C'est variable de cas à cas, mais la réduction des frais de conception, d'implémentation et de maintenance n'est pas à négliger. Avec des fichiers adaptables et conviviaux en plus, on peut espérer réduire le coût du tiers et parfois même de la moitié.

Prenons maintenant en compte la compatibilité. Elle fait baisser considérablement le coût du logiciel produit tant par nous que par des sociétés extérieures utilisant notre matériel.

Donnons un exemple réel :

DOAS, le projet sur lequel je travaille, a été réalisé en double : DOAS6 et DOAS7, faute de compatibilité DPS6 et DPS7. Le projet exigea environ 70 années homme pour être livrable aux clients. Bull aurait pu, avec deux ordinateurs adaptables différents, épargner presque la moitié, soit 30 année homme, pour obtenir le même résultat.

Nous comptons donc sur l'ordinateur adaptable pour améliorer significativement l'efficacité de la compagnie et pour permettre avec le même effort de réaliser mieux, plus et plus loin, face à ses clients et à ses concurrents.

Résumé technique de la machine molle et vocabulaire

Carlos Derbez et André Richard

I. Déclaration de variables

Types : Événement, Booléen, Caractère, Nombre, Adresse.

Capacité des variables, des paramètres et de constantes logés dans un mot.

Caractères : de 1 à 255.
Nombres : de 1 à 500 chiffres.
Adresses : illimitées.
Booléens : 1 caractère.
Événements : 1 caractère.

La capacité de la variable est spécifié à la programmation, ou bien au lancement.

II. Compatibilité

Tous les programme sont compatibles, tant en ascendant qu'en descendant, si les ressources sont suffisantes.

III. Modularité logicielle

Les bibliothèques de programmes sont prévues.
Les procédures peuvent être transmises.
Les procédures d'exception existent.

IV. Système d'exploitation

La machine molle n'a pas de système d'exploitation, mais on peut programmer le sien ou bien utiliser une bibliothèque.

V. Performances

La machine molle est plus lente mais bien plus facile à programmer que l'ordinateur classique. C'est le prix à payer pour atteindre la convivialité.

VI. Programmation

La machine molle se programme comme une sorte de Pascal temps réel. Ses données sont empilées dans un arbre de mémoires, ou chaque branche est allouée à une tâche en pile, et chaque pile est scindée en deux : une partie constante et une partie variable. C'est cette dernière qui rend molle la machine.

VII. Le code est généré par les instructions typiques du Pascal

- affectation;
- comparaisons;
- expressions;
- tableaux;
- structures;
- si...alors...sinon...;
- case-of;
- procédures (avec un nombre illimité de paramètres);
- fonctions (appelés par nom ou par valeur);
- récursivité;
- réentrance.

Le code est fractionné en segments qui sont synchronisés (pour les problèmes temps réel) par les relations de l'algèbre de faits. Elle utilise l'opérateur temporel « puis » pour atteindre une synchronisation sans faille.

VIII. Matériel

On peut ajouter sans limites de la mémoire. On peut calculer des tableaux de toute dimension. On peut réagir à toute situation temps réel ou de multiprogrammation. On peut aussi ajouter sans limites des blocs de calcul, ce qui accélère considérablement le traitement dans certains cas. On peut ajouter des périphériques sans limites.

IX. Extensibilité

Tant la machine molle elle même que ses langages de programmation peuvent être étendus, si on le veut.

X. Mécanismes nouveaux

Le fonctionnement de la machine molle met en œuvre simultanément quatre mécanismes :

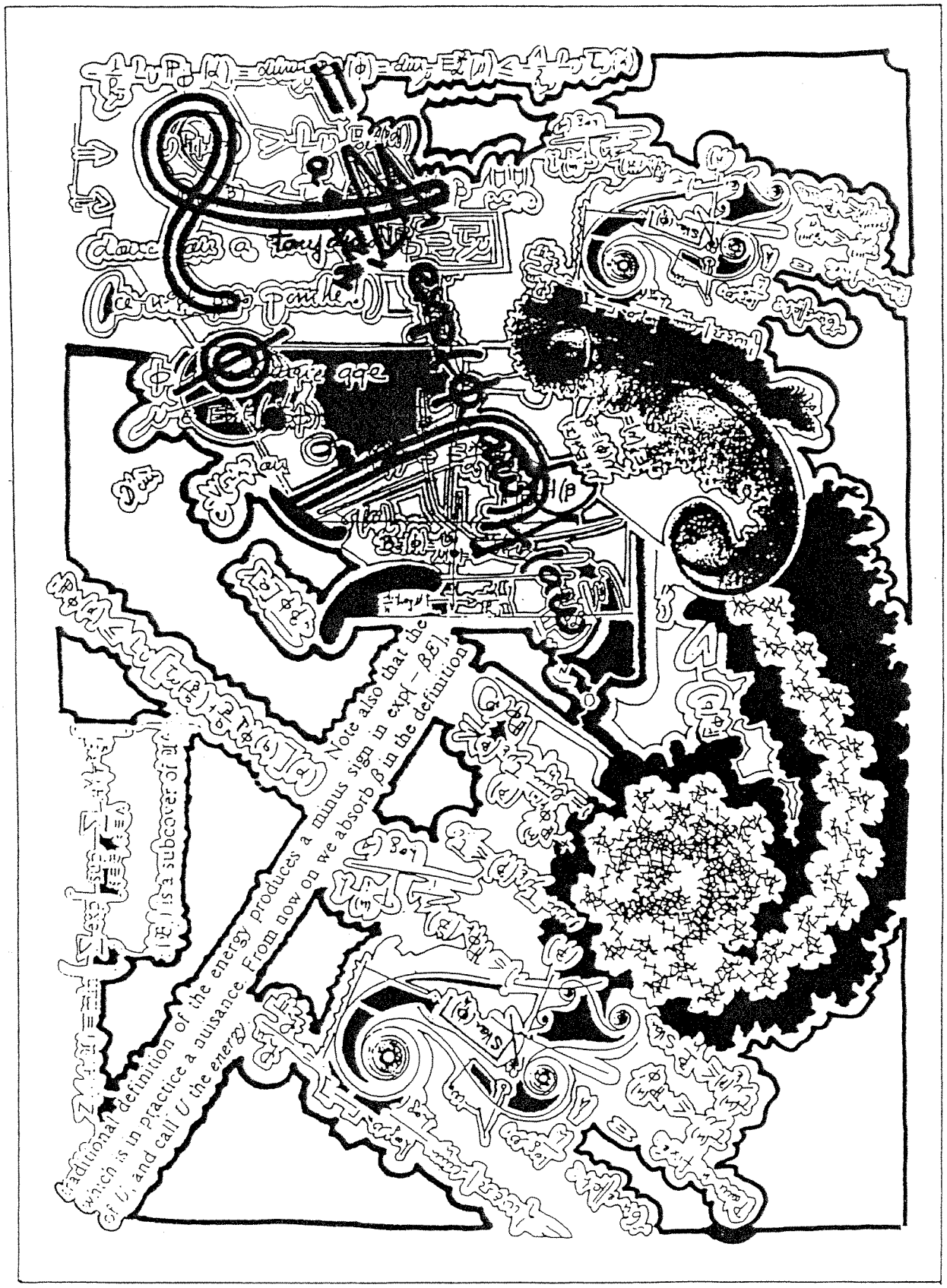
- Un mécanisme séquentiel classique, utilisant une pile.
- Un mécanisme nouveau d'allocation de mémoires, de longueur variable, réalisant la mémoire adaptable ou « molle ».
- Un mécanisme nouveau de synchronisation des séquences de code mettant en œuvre l'algèbre de faits. Celle-ci permet aussi la conception unifiée et formalisée d'un produit informatique. L'algèbre de faits conduit à une conception sans erreur du besoin au produit informatique.
- Un mécanisme nouveau de « calcul à la première demande » permettant l'exécution efficace de programmes non procéduraux dégageant le programmeur de tout souci relatif à l'ordre des calculs.

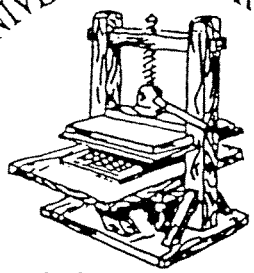
XI. Vocabulaire

Ces textes ont été rédigés à différentes époques, très éloignées les unes des autres, dans des circonstances particulières. Ainsi, plusieurs mots peuvent se référer au même objet :

- Algèbre de faits, algèbre historique, informatique dynamique.
- Machine molle, ordinateur adaptable, système adaptable, la machine.
- Nous disons aussi que l'informatique molle est générée par la machine molle (machine molle + algèbre de faits).

VOUZZAVEDIBISAR



SERVICE REPROGRAPHIE - UNIVERSITÉ DE PROVENCE - CENTRE ST. CHARLES -

3, place Victor Hugo
13331 MARSEILLE cedex 3