Contribution aux techniques de mise en pages et d'édition par ordinateur : conception et implémentation de L.A.M.P.E., un langage spécialisé

Jean-Charles Profizi

THESE

présentée à

UNIVERSITE SCIENTIFIQUE ET MEDICALE DE GRENOBLE

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

par

pour obtenir le grade de

Docteur de troisième cycle

Spécialité : INFORMATIQUE

Jean-Charles PROFIZI

CONTRIBUTION AUX TECHNIQUES DE MISE EN PAGES

ET D'EDITION PAR ORDINATEUR.

CONCEPTION ET IMPLEMENTATION DE L.A.M.P.E.,

UN LANGAGE SPECIALISE.

Thèse soutenue le 4 juillet 1974 devant la Commission d'Examen

Président : Monsieur N. GASTINEL

Examinateurs :
- Monsieur J.C. BOUSSARD
- Monsieur G. VEILLON
- Monsieur F. PECCOUD
- Monsieur M. BERTHAUD

UNIVERSITE SCIENTIFIQUE
ET MEDICALE DE GRENOBLE

LISTE DES PROFESSEURS

Président : Monsieur Michel SOUTIF
Vice-Président : Monsieur Gabriel CAU

PROFESSEURS TITULAIRES

MM.	ANGLES D'AURIAC Paul	Mécanique des fluides
	ARNAUD Georges	Clinique des maladies infectieuses
	ARNAUD Paul	Chimie
	AUBERT Guy	Physique
	AYANT Yves	Physique approfondie
Mme	BARBIER Marie-Jeanne	Electrochimie
MM.	BARBIER Jean-Claude	Physique expérimentale
	BARBIER Reynold	Géologie appliquée
	BARJON Robert	Physique nucléaire
	BARNOUD Fernand	Biosynthèse de la cellulose
	BARRA Jean-René	Statistiques
	BARRIE Joseph	Clinique chirurgicale
	BENOIT Jean	Radioélectricité
	BERNARD Alain	Mathématiques Pures
	BESSON Jean	Electrochimie
	BEZES Henri	Chirurgie générale
	BLAMBERT Maurice	Mathématiques Pures
	BOLLIET Louis	Informatique (IUT B)
	BONNET Georges	Electrotechnique
	BONNET Jean-Louis	Clinique ophtalmologique
	BONNET-EYMARD Joseph	Pathologie médicale
	BONNIER Etienne	Electrochimie Electrométallurgie
	BOUCHERLE André	Chimie et Toxicologie
	BOUCHEZ Robert	Physique nucléaire
	BOUSSARD Jean-Claude	Mathématiques Appliquées
	BRAVARD Yves	Géographie
	BRISSONNEAU Pierre	Physique du solide
	BUYLE-BODIN Maurice	Electronique
	CABANAC Jean	Pathologie chirurgicale
	CABANEL Jean	Clinique rhumatologique et hydrologie
	CALAS François	Anatomie
	CARRAZ Gilbert	Biologie animale et pharmacodynamie
	CAU Gabriel	Médecine légale et Toxicologie
	CAUQUIS Georges	Chimie organique
	CHABAUTY Claude	Mathématiques Pures
	CHARACHON Robert	Oto-Rhino-Laryngologie
	CHATEAU Robert	Thérapeutique
	CHENE Marcel	Chimie papetière
	COEUR André	Pharmacie chimique
	CONTAMIN Robert	Clinique gynécologique
	COUDERC Pierre	Anatomie Pathologique
	CRAYA Antoine	Mécanique

Mme	DEBELMAS Anne-Marie	Matière médicale
MM.	DEBELMAS Jacques	Géologie générale
	DEGRANGE Charles	Zoologie
	DESRE Pierre	Métallurgie
	DESSAUX Georges	Physiologie animale
	DODU Jacques	Mécanique appliquée
	DOLIQUE Jean-Michel	Physique des plasmas
	DREYFUS Bernard	Thermodynamique
	DUCROS Pierre	Cristallographie
	DUGOIS Pierre	Clinique de Dermatologie et Syphiligraphie
	FAU René	Clinique neuro-psychiatrique
	FELICI Noël	Electrostatique
	GAGNAIRE Didier	Chimie physique
	GALLISSOT François	Mathématiques Pures
	GALVANI Octave	Mathématiques Pures
	GASTINEL Noël	Analyse numérique
	GEINDRE Michel	Electroradiologie
	GERBER Robert	Mathématiques Pures
	GIRAUD Pierre	Géologie
	KLEIN Joseph	Mathématiques Pures
Mme	KOFLER Lucie	Botanique et Physilogie végétale
MM.	KOSZUL Jean-Louis	Mathématiques Pures
	KRAVTCHENKO Julien	Mécanique
	KUNTZMANN Jean	Mathématiques appliquées
	LACAZE Albert	Thermodynamique
	LACHARME Jean	Biologie végétale
	LAJZEROWICZ Joseph	Physique
	LATREILLE René	Chirurgie générale
	LATURAZE Jean	Biochimie pharmaceutique
	LAURENT Pierre-Jean	Mathématiques appliquées
	LEDRU Jean	Clinique médicale B
	LLIBOUTRY Louis	Géophysique
	LOUP Jean	Géographie
Mle	LUTZ Elisabeth	Mathématiques Pures
MM.	MALGRANGE Bernard	Mathématiques Pures
	MALINAS Yves	Clinique obstétricale
	MARTIN-NOEL Pierre	Seméiologie médicale
	MASSEPORT Jean	Géographie
	MAZARE Yves	Clinique médicale A
	MICHEL Robert	Minéralogie et Pétrographie
	MOURIQUAND Claude	Histologie
	MOUSSA André	Chimie nucléaire
	NEEL Louis	Physique du solide
	OZENDA Paul	Botanique
	PAUTHENET René	Electrotechnique
	PAYAN Jean-Jacques	Mathématiques Pures
	PEBAY-PEYROULA Jean-Claude	Physique
	PERRET René	Servomécanismes
	PILLET Emile	Physique industrielle
	RASSAT André	Chimie systématique
	RENARD Michel	Thermodynamique
	REULOS René	Physique industrielle
	RINALDI Renaud	Physique
	ROGET Jean	Clinique de pédiatrie et de puériculture
	SANTON Lucien	Mécanique
	SEIGNEURIN Raymond	Microbiologie et Hygiène
	SENGEL Philippe	Zoologie
	SILBERT Robert	Mécanique des fluides
	SOUTIF Michel	Physique générale

MM.	TANCHE Maurice	Physiologie
	TRAYNARD Philippe	Chimie générale
	VAILLAND François	Zoologie
	VALENTIN Jacques	Physique nucléaire
	VAUQUOIS Bernard	Calcul électronique
Mme	VERAIN Alice	Pharmacie galénique
M.	VERAIN André	Physique
Mme	VEYRET Germaine	Géographie
MM.	VEYRET Paul	Géographie
	VIGNAIS Pierre	Biochimie médicale
	YOCCOZ Jean	Physique nucléaire théorique

PROFESSEURS ASSOCIES

MM.	BULLEMER Bernhard	Physique
	HANO JUN-ICHI	Mathématiques Pures
	STEPHENS Michaël	Mathématiques appliquées

PROFESSEURS SANS CHAIRE

MM.	BEAUDOING André	Pédiatrie
Mme	BERTRANDIAS Françoise	Mathématiques Pures
MM.	BERTRANDIAS Jean-Paul	Mathématiques appliquées
	BIAREZ Jean-Pierre	Mécanique
	BONNETAIN Lucien	Chimie minérale
Mme	BONNIER Jane	Chimie générale
MM.	CARLIER Georges	Biologie végétale
	COHEN Joseph	Electrotechnique
	COUMES André	Radioélectricité
	DEPASSEL Roger	Mécanique des fluides
	DEPORTES Charles	Chimie minérale
	GAUTHIER Yves	Sciences biologiques
	GAVEND Michel	Pharmacologie
	GERMAIN Jean-Pierre	Mécanique
	GIDON Paul	Géologie et Minéralogie
	GLENAT René	Chimie organique
	HACQUES Gérard	Calcul numérique
	JANIN Bernard	Géographie
Mme	KAHANE Josette	Physique
MM.	MULLER Jean-Michel	Thérapeutique
	PERRIAUX Jean-Jacques	Géologie et Minéralogie
	POULOUJADOFF Michel	Electrotechnique
	REBECQ Jacques	Biologie (CUS)
	REVOL Michel	Urologie
	REYMOND Jean-Charles	Chirurgie générale
	ROBERT André	Chimie papetière
	DE ROUGEMONT Jacques	Neurochirurgie
	SARRAZIN Roger	Anatomie et chirurgie
	SARROT-REYNAULD Jean	Géologie
	SIBILLE Robert	Construction mécanique
	SIROT Louis	Chirurgie générale
Mme	SOUTIF Jeanne	Physique générale

MAITRES DE CONFERENCES ET MAITRES DE CONFERENCES AGREGES

Mle	AGNIUS-DELORD Claudine	Physique pharmaceutique
	ALARY Josette	Chimie analytique
MM.	AMBLARD Pierre	Dermatologie
	AMBROISE-THOMAS Pierre	Parasitologie
	ARMAND Yves	Chimie
	BEGUIN Claude	Chimie organique
	BELORIZKY Elie	Physique
	BENZAKEN Claude	Mathématiques appliquées
	BILLET Jean	Géographie
	BLIMAN Samuel	Electronique (EIE)
	BLOCH Daniel	Electrotechnique
Mme	BOUCHE Liane	Mathématiques (CUS)
MM.	BOUCHET Yves	Anatomie
	BOUVARD Maurice	Mécanique des fluides
	BRODEAU François	Mathématiques (IUT B)
	BRUGEL Lucien	Energétique
	BUISSON Roger	Physique
	BUTEL Jean	Orthopédie
	CHAMBAZ Edmond	Biochimie médicale
	CHAMPETIER Jean	Anatomie et organogénèse
	CHIAVERINA Jean	Biologie appliquée (EFP)
	CHIBON Pierre	Biologie animale
	COHEN-ADDAD Jean-Pierre	Spectrométrie physique
	COLOMB Maurice	Biochimie médicale
	CONTE René	Physique
	COULOMB Max	Radiologie
	CROUZET Guy	Radiologie
	DURAND Francis	Métallurgie
	DUSSAUD René	Mathématiques (CUS)
Mme	ETERRADOSSI Jacqueline	Physiologie
MM.	FAURE Jacques	Médecine légale
	GENSAC Pierre	Botanique
	GIDON Maurice	Géologie
	GRIFFITHS Michaël	Mathématiques appliquées
	GROULADE Joseph	Biochimie médicale
	HOLLARD Daniel	Hématologie
	HUGONOT Robert	Hygiène et Médecine préventive
	IDELMAN Simon	Physiologie animale
	IVANES Marcel	Electricité
	JALBERT Pierre	Histologie
	JOLY Jean-René	Mathématiques Pures
	JOUBERT Jean-Claude	Physique du solide
	JULLIEN Pierre	Mathématiques Pures
	KAHANE André	Physique générale
	KUHN Gérard	Physique
	LACOUME Jean-Louis	Physique
Mme	LAJZEROWICZ Jeannine	Physique
MM.	LANCIA Roland	Physique atomique
	LE JUNTER Noël	Electronique
	LEROY Philippe	Mathématiques
	LOISEAUX Jean-Marie	Physique nucléaire
	LONGEQUEUE Jean-Pierre	Physique nucléaire
	LUU DUC Cuong	Chimie organique
	MACHE Régis	Physiologie végétale
	MAGNIN Robert	Hygiène et Médecine préventive
	MARECHAL Jean	Mécanique
	MARTIN-BOUYER Michel	Chimie (CUS)

MM.	MAYNARD Roger	Physique du solide
	MICHOULIER Jean	Physique (IUT A)
	MICOUD Max	Maladies infectieuses
	MOREAU René	Hydraulique (INP)
	NEGRE Robert	Mécanique
	PARAMELLE Bernard	Pneumologie
	PECCOUD François	Analyse (IUT B)
	PEFFEN René	Métallurgie
	PELMONT Jean	Physiologie animale
	PERRET Jean	Neurologie
	PERRIN Louis	Pathologie expérimentale
	PFISTER Jean-Claude	Physique du solide
	PHELIP Xavier	Rhumatologie
Mlle	RIERY Yvette	Biologie animale
MM.	RACHAIL Michel	Médecine interne
	RACINET Claude	Gynécologie et obstétrique
	RENAUD Maurice	Chimie
	RICHARD Lucien	Botanique
Mme	RINAUDO Marquerite	Chimie macromoléculaire
MM.	ROMIER Guy	Mathématiques (IUT B)
	SHOM Jean-Claude	Chimie générale
	STIEGLITZ Paul	Anesthésiologie
	STOEBNER Pierre	Anatomie pathologique
	VAN CUTSEM Bernard	Mathématiques appliquées
	VEILLON Gérard	Mathématiques appliquées (INP)
	VIALON Pierre	Géologie
	VOOG Robert	Médecine interne
	VROUSSOS Constantin	Radiologie
	ZADWORNY François	Electronique

MAITRES DE CONFERENCES ASSOCIES

MM.	BOUDOURIS Georges	Radioélectricité
	CHEEKE John	Thermodynamique
	GOLDSCHMIDT Hubert	Mathématiques
	SIDNEY STUARD	Mathématiques Pures
	YACOUD Mahmoud	Médecine légale

CHARGES DE FONCTIONS DE MAITRES DE CONFERENCES

Mme	BERIEL Hélène	Physilogie
Mme	RENAUDET Jacqueline	Microbiologie

Fait le 30 mai 1972.

A mes Parents

Je tiens à exprimer ma profonde gratitude à:

Monsieur le Professeur Noël GASTINEL, Directeur du Centre Intertuniversitaire de Calcul de Grenoble, pour l'honneur qu'il a bien voulu me faire en acceptant de présider le Jury.

Monsieur le Professeur J.C.BOUSSARD, qui, avec une attention particulièrement bienveillante, a tout au long dirigé mon travail, et, tant par son éminent concours que par ses précieux encouragements, m'a permis de le mener à bien.

Monsieur G.VEILLON, Professeur à l'E.N.S.I.M.A.G., qui a porté un vif intêrêt à cette étude, à ses perspectives et a bien voulu m'accorder une aide très fructueuse et sympathique.

Monsieur F.PECCOUD, Maître de Conférences (I.U.T.B.) qui m'a déterminé à traiter ce sujet et a eu l'amabilité d'orienter dans le meilleur sens mes premières investigations.

Monsieur BERTHAUD, Ingénieur au Centre Scientifique I.B.M., qui, maintes fois mis à contribution, m'a, avec sa compétence particulière dans ce domaine, très efficacement guidé en vue de l'implémentation du langage.

Je voudrais aussi,
Témoigner, ici, ma grande reconnaissance à Monsieur BELLINO qui m'a initié à l'Informatique et, depuis, m'a toujours incité à poursuivre mes efforts,

Dire à Messieurs SEGUIN et LUCAS, qui m'ont fait l'amitié de s'intéresser à mes travaux, combien j'ai apprécié leurs suggestions,

Enfin, rendre l'hommage qu'ils méritent à Mademoiselle C.SALLUSTIO, Monsieur RASOLONJAVOTO et le Personnel du Service reproduction, pour l'empressement et le soin avec lesquels ils ont assumé la réalisation matérielle de ce document.

S O M M A I R E

TABLE DETAILLEE DES CHAPITRES

PREMIERE PARTIE : ORIGINE ET PRESENTATION DE LA SOLUTION PROPOSEE

CHAPITRE 1 : L'Informatique et l'Edition

CHAPITRE 2 : Présentation du langage "L.A.M.P.E." et du "Système M.P.E."

DEUXIEME PARTIE : DESCRIPTION DU LANGAGE ALGORITHMIQUE DE MISE-EN-PAGES ET D'EDITION.

CHAPITRE 1 : Le Manuel de Référence

TROISIEME PARTIE : L'IMPLEMENTATION DU LANGAGE

CHAPITRE 1 : Contrôle des diverses phases du traitement d'un programme d'un programme PL/1-L.A.M.P.E.

CHAPITRE 2 : La pré-compilation(ou Compilation L.A.M.P.E.)

CHAPITRE 3 : Le Système M.P.E.

INTRODUCTION GENERALE

Les révolutions techniques et économiques ont souvent été à l'origine de la transformation de la Société.

Si le XIX^{ème} siècle a été marqué par la révolution industrielle, il semble que la seconde moitié du XX^{ème} sera fortement "imprégnée" de ce que l'on ose déjà appeler la "révolution informatique".

Certes, le ferment de ces mutations réside en l'évolution croissante des connaissances humaines, mais, aussi, en leur transmission sans cesse améliorée.

"La seule vraie révolution, c'est GUTENBERG qui l'a accomplie. C'est lui qui a le plus contribué à répandre le savoir" [AR-4-1].

L'amélioration de la communication entre les hommes peut être considérée comme l'un des traits significatifs de notre époque et comme l'une des applications les plus nobles de la science contemporaine.

Les découvertes récentes apportent, dans ce domaine, une contribution remarquable, en fournissant notamment les moyens les mieux adaptés à la diffusion des masses d'informations, élément essentiel de notre culture et symbole de notre civilisation.

Cependant, ces informations n'ont pas toutes le même intérêt ni le même impact.

Les toutes nouvelles méthodes audio-visuelles sont plus précisément orientées vers la vulgarisation de notions facilement assimilables. Par contre, les méthodes traditionnelles, engendrant des imprimés, ou selon des techniques plus récentes des micro-fiches, sont destinées à fournir des informations dont l'étude nécessite une certaine réflexion et même des consultations fréquentes.

Ces deux techniques se complètent plus qu'elles ne s'opposent.

L'EDITION, au prix d'une mutation, irréversible et profonde, a donc un avenir prometteur.

Pour qu'elle puisse répondre aux exigences actuelles de rapidité et de transmission à longue distance, l'Informatique a dû lui apporter son concours.

Initialement, le recours à cette science nouvelle a été envisagé comme une aide à la composition de textes. Puis, plus récemment, comme le moyen d'assurer directement, au terme de leur "composition", l'impression des articles devant figurer dans des traités scientifiques et techniques. Mais, les solutions jusqu'ic envisagées donnaient à l'ordinateur l'unique possibilité d'être un exécutant simplement docile et conscencieux.

Pour notre part, il nous a semblé que l'évolution de telles techniques devait nécessairement conférer à l'ordinateur un rôle plus important, en ce sens que, tout en le laissant tributaire des décisions du programmeur, la liberté devai cependant lui être offerte de proposer des dispositions efficaces.

C'est dans cette optique que nous souhaitons apporter notre modeste contribution à la résolution des délicats problèmes impliqués par la "Mise-en-pages" et "l'Edition".

Confrontés à une étude vaste et au sujet de laquelle nos connaissances étaient réduites, nous avons volontairement limité nos investigations et décidé de procéder par étapes successives.

Sur le plan théorique, nous nous sommes résolument engagé dans la recherche de solutions assimilant la description d'une mise-en-pages à la ~~t~~ranscription d'un raisonnement algorithmique.

Ce qui nous a amené :

* A donner une importance primordiale aux relations diverses unissant, dans le cadre d'une mise en pages, les composants d'une Edition.
* A établir un subtil compromis entre les considérations qui, dans toute édition, relèvent de l'art et celles qui émanent d'un raisonnement logique.

Sur le plan pratique, nous nous sommes fixé des objectifs précis :

- Concevoir une application des solutions élaborées, se prêtant à des évolutions régulières et prévisibles.

- Mesurer l'efficacité des solutions proposées, face à la variété des difficultés à surmonter, en sensibilisant le plus grand nombre possible d'utilisateurs.

- Rentabiliser rapidement nos travaux, compte tenu des besoins immédiats.

Ce qui nous a amené :

* A délaisser présentement des techniques séduisantes telles que l'interrogation directe ou l'utilisation d'un support graphique ; décidant de ne faire appel à elles, vu l'intérêt certain qu'elles présentent dans notre cas, que lors d'une prochaine étape.

* A envisager l'étude entreprise sous l'angle particulier de l'édition des différents types de données pouvant être traités par un programme en langage évolué.

En effet, les langages évolués, dont la puissance et les possibilités ne cessent de s'accroître, sont de nos jours de plus en plus utilisés.

Cependant, leur aptitude à résoudre fort aisément les problèmes les plus complexes et les plus différents contraste curieusement avec le caractère rudimentaire des méthodes d'Edition qu'ils nous proposent.

Editer des informations traitées dans un programme écrit en de tels langages, revient généralement à décrire, à l'aide d'instructions "format", la composition ligne par ligne des pages à imprimer. Cela, au prix de calculs longs et fastidieux tenant compte du type des informations, de leurs dimensions, et, de l'emplacement qu'il est convenu de leur assigner sur le support de l'Edition.

Cette technique est peu élégante et souffre de deux handicaps :

- Une certaine rigidité dûe au fait que toute modification relative au format du papier d'imprimante, portant sur la disposition préalablement

adoptée ou même s'adressant à l'une seulement des informations assujetties à co-exister sur une même page, conduit inévitablement à repenser la disposition initiale et à réécrire la plupart des instructions.

- Une mise au point délicate, et, d'autant plus malaisée que la présentation envisagée concerne un plus grand nombre de données.

Tant de contraintes, de difficultés, et par suite de temps perdu, ne peuvent qu'inciter à procéder à des Editions sommaires. Ce qui est en contradiction avec la tendance à une bonne présentation des "listings" (désignation conventionnelle des imprimés édités par un périphérique d'ordinateur) qui se manifeste actuellement avec une acuité grandissante.

La pénétration de l'Informatique dans les secteurs d'activité les plus divers confère en effet aux "listings" un rôle prépondérant. Etant, dans de nombreux cas et en gestion tout particulièrement, les témoins incontestés des traitements effectués, ils servent de base à des décisions importantes, passent à ces fins en de nombreuses mains, et se doivent donc d'offrir une présentation claire et logique.

La mise en application des solutions que nous nous sommes proposées d'atteindre va, dans l'immédiat, avoir pour rôle de simplifier la démarche du programmeur en vue de la Mise en pages et de l'Edition des résultats d'un traitement.

Pour y parvenir, nous avons été amené à concevoir d'une part, un dispositif "Logiciel" (ou "Software") et, d'autre part, un langage.

Le dispositif programmé, dénommé système "M.P.E." est destiné à fonctionner en corrélation avec le programme objet, délivré par un compilateur PL/1, et a la charge d'effectuer la Mise en pages et l'Edition d'informations traitées dans tout programme PL/1.

Son originalité consiste à assumer, en lieu et place du programmeur, les tâches nombreuses, longues et fastidieuses que nécessite la prise en considération des dimensions respectives des informations et des pages; puis, à assurer, conformément aux directives qui lui sont dictées , l'Edition envisagée.

Le langage, adoptant PL/1 comme langage hôte, permet à l'utilisateur de communiquer ses désiderata au système précité.

Ces quelques considérations ont visé à donner un rapide aperçu des motivations et de l'esprit de notre étude dont nous détaillerons les divers aspects dans les trois parties que comporte l'exposé ci-après.

La **première partie** dresse un bref bilan des recherches entreprises dans le domaine de l'application de l'Informatique à l'Edition; puis,elle s'attache, en un deuxième temps, à délimiter le champ de nos investigations et à préciser tant la solution adoptée que les concepts qui en autorisent l'application.

La **deuxième partie** s'adresse à l'informaticien auquel est soumise la définition du langage L.A.M.P.E.. Elle concerne également le futur utilisateur auquel est donné un aperçu de la corrélation entre les ressources du langage et le mode de raisonnement proposé.

La **troisième partie**, plus technique, est consacrée aux particularités relevées dans les méthodes d'implémentation du langage, ainsi que dans l'architecture et le fonctionnement du "Système M.P.E.".

P R E M I E R E P A R T I E

<><><><><>

ORIGINE ET PRESENTATION
DE LA SOLUTION ADOPTEE

PREMIERE PARTIE

C H A P I T R E I

L'INFORMATIQUE ET L'EDITION

L'Edition est une opération complexe en elle-même. Nous pouvons répartir en deux catégories les travaux qui assurent sa réalisation.

- Les uns, intéressants et délicats puisque requérant toute l'habileté d'un spécialiste. Tel est le cas de l'ensemble des décisions qui régissent toute Mise-en-pages.

- Les autres, fastidieux puisque routiniers et souvent inefficaces puisque ne satisfaisant pas toujours aux impératifs de rapidité imposés, de nos jours, à toutes les techniques de communication. Tel est le cas de l'ensemble des considérations physiques qui conditionnent la Mise-en-pages.

Ainsi, l'automatisation des travaux systématiques devait nécessairement être envisagée et fournir, à l'Informatique, une occasion nouvelle d'apporter son concours.

I.1. L'INFORMATIQUE MISE AU SERVICE DE L'EDITION

Au vu des résultats acquis au cours des années précédentes, il est possible de distinguer deux aspects dans l'application de l'Informatique à l'Edition :

* L'aspect d'aide à l'Edition dans le cadre d'une imprimerie; aspect qui ne constitue pas le but actuel de notre étude.
* L'aspect d'aide à l'Edition dans le cadre d'un Centre de Calcul; aspect qui a retenu notre attention.

I.1.1. L'Informatique appliquée à l'Edition dans le cadre d'une imprimerie.

En supprimant les tâches systématiques conditionnant toute Edition et confiées jusqu'à présent à une certaine catégorie de personnel, les ordinateurs contribuent à raccourcir de façon très appréciable les délais nécessités par la réalisation, à partir d'un manuscrit, des épreuves destinées à être reproduites en un grand nombre d'exemplaires.

Les ordinateurs interviennent principalement dans la "Composition", la "Présentation" et, depuis peu, dans la "Mise-en-pages" de textes.

Pour illustrer l'intérêt procuré par leur utilisation, nous nous bornerons à souligner leur efficacité dans l'une des tâches qui leur sont dévolues : la "Justification à droite".

Opération consistant à terminer chaque ligne constitutive de l'imprimé toujours par un caractère, jamais par un espace. Ce qui ne peut être obtenu, selon la longueur de la ligne, que par une répartition uniforme des espaces entre les différents mots ou par une troncature judicieuse du dernier mot de la ligne [IB-1-1].

Dans un tel contexte, l'ordinateur joue le rôle d'organe d'interprétation et de commande [TR-1-1].

- Organe d'interprétation puisque, par l'intermédiaire généralement d'un ruban perforé, lui sont soumis, outre le texte qu'il doit traiter, les directives ou "codes fonctions" qu'il doit interpréter.

Interprétation d'autant plus fondamentale que ces directives tradui-

sent les aspirations du programmeur: coordonnées de l'implantation, choix des caractères, déclenchement des alinéas, etc...

- Organe de commande puisque, en vue de la concrétisation du traitement effectué, il communique, par l'intermédiaire d'une bande magnétique, toutes les précisions nécessaires à la mise en oeuvre d'une photo-composeuse ou d'une composeuse électronique. Se trouve ainsi désignée une unité périphérique, pouvant être assimilée à une machine outil, à laquelle incombe le rôle de fournir un film ou "placard papier", directement exploitable par tous les systèmes de reproduction et plus particulièrement par "l'offset".

I.1.2. L'Informatique appliquée à l'Edition dans le cadre d'un Centre de Calcul

L'Edition envisagée dans le cadre d'un Centre de Calcul est axée sur la diffusion de renseignements à l'intention des utilisateurs d'un même système d'exploitation.

L'ordinateur joue alors un rôle moins élaboré. D'autant plus qu'utilisant les imprimantes qui lui sont connectées, il dispose, vu l'uniformité des caractères, de ressources limitées quant à l'aspect visuel que peuvent revêtir les imprimés.

Plusieurs tendances se sont manifestées, témoignant de l'intérêt porté par les Informaticiens à ce problème actuellement en pleine évolution.

- Une de ces tendances fût à l'origine de "systèmes" orientés vers l'édition de textes et adoptant soit le mode d'accès séquentiel, soit le mode d'accès direct.

Tel est le cas de TEXT 360 [IB-1-2], de FORMAT [BE-1-1], de SCRIPT [IB-1-5], dont l'utilisation présida respectivement à la publication des récentes brochures IBM telles que [IB-3-2],[IB-3-3], etc, du livre "Compiler Construction For Digital Computer" [GR-4-1] et de thèses [LE-2-1].

- Une autre tendance fût motivée par le caractère anachronique des instructions d'édition utilisées dans les langages évolués usuels. Instructions d'un niveau élémentaire et dont l'utilisation est à la fois fastidieuse et complexe.

Pour remédier à la perte de temps qu'entraine généralement l'édition des résultats d'un traitement, les programmeurs optent pour les techniques les

plus rapides et les plus simples. Ce qui, en contrepartie, entraine un dépouillement pénible des "listings" et une consommation déraisonnable de papier.

C'est ainsi que des études furent entreprises pour améliorer les techniques d'édition des résultats fournis tant par des algorithmes très divers et librement programmés dans des langages évolués usuels, que par des modules spécialisés.

Selon le cas, il fut donc procédé soit à l'amélioration des instructions d'édition des langages évolués ("Report Section"[IB-1-4],RPG [IB-1-6]),soit à la conception de "packages" chargés de la saisie et de l'ordonnancement des données, de l'exécution de traitements spécialisés (statistiques, gestion, etc..) puis de l'Edition finale.

Tel est le cas des "esquifs" fournis par "AVA-INFORMATIQUE" [AI-1-1] ou des "packages-programmes" proposés par la "SOCIETE D'INFORMATIQUE ET DE SYSTEME" [CS-1-1].

I.2. LES CONSIDERATIONS AYANT PERMIS D'ORIENTER NOTRE ETUDE

Le bilan, volontairement succint, que nous venons de dresser, nous inspire quelques remarques essentielles qui sont à l'origine de la délimitation de notre champ d'investigation.

I.2.1. La complémentarité des aspects offerts par l'application de l'Informatique à l'Edition.

L'aide à l'Edition dans une imprimerie et dans un Centre de Calcul représente deux axes de recherche, volontairement dissociés afin de distinguer la nature des difficultés, mais complémentaires puisque visant tous deux à un même but :

Automatiser le processus de l'Edition en déchargeant le spécialiste de toutes les tâches "mécaniques".

I.2.2. Les résultats les plus significatifs ont été acquis dans l'Edition de textes

- Dans le cadre de l'Imprimerie, les informations traitées sont limitées à un type précis : les textes.

L'accent est mis sur leur "composition" (justification à droite) et sur la souplesse d'utilisation des diverses "polices de caractères" disponibles.

Cela, de façon à ce qu'une telle automatisation ne nuise pas aux traditionnelles possibilités de mise en évidence de certains passages et n'entrave pas le respect des règles régissant l'aspect visuel des imprimés.

- Dans le cadre d'un Centre de Calcul, par contre, ce sont les problèmes de mise en évidence qui sont éludés au profit de ceux que pose l'édition d'informations de types variés.

I.2.3. L'automatisation n'a pas, pour l'instant, maitrisé la Mise-en-pages

Malgré les progrés accomplis au cours de ces dernières années, on ne peut parler actuellement d'automatisation de la Mise-en-pages, opération-clé de toute Edition.

Des solutions tant "matérielles" ("Hardware") [WA-1-1] que "logicielles" ("Software")[TR-1-1] ont permis récemment de surmonter nombre de difficultés engendrées par l'Edition des textes. Par contre,l'Edition d'informations de types divers contraint toujours l'utilisateur d'un système informatique approprié à respecter une démarche classique.

Il est en effet nécessaire :

* En un premier stade, de déterminer, au terme d'un travail long et souvent délicat, les dispositions rigides (ou "Maquettes") auxquelles seront assujettis les composants de l'Edition.

* En un second stade, de codifier plus ou moins aisément les "Maquettes" préalablement établies, afin de communiquer à l'ordinateur les cadrages retenus.

Ceci nécessite d'avoir recours soit à des "Formats" généralisés (à l'échelon de la page), soit à l'adjonction, aux informations sélectionnées,

d'indicateurs de position (technique fort astreignante : quatre bordereaux de perforation différents sont, par exemple, imposés par l'un des systèmes précités).

Cette rigidité est particulièrement nuisible dans le cas où le système informatique utilisé adopte l'accès séquentiel [BE-1-1]. Cas dans lequel il est difficile, sinon impossible, de disposer : titres ou sous-titres de chapître, notes explicatives à la suite de schémas etc...

C'est ainsi que dans le livre de David GRIES [GR-4-1], titres et sous-titres de chapîtres, tableaux, libellés, ont été rajoutés à l'original par des procédés habituels.

I.3. LES OBJECTIFS QUE NOUS NOUS PROPOSONS D'ATTEINDRE

Notre étude est axée sur l'aide que peut apporter l'ordinateur dans la détermination de la Mise-en-pages, sans distinction des types d'informations.

Ainsi, nous nous proposons de confier, au programmeur, le soin de décrire ses désidérata de façon souple et logique; l'ordinateur se chargeant de les interpréter, d'assumer en conséquence les différents travaux nécessaires, de proposer une solution acceptable et de la mettre en application si la décision en est prise.

Notre champ d'investigation peut, de ce fait, se résumer en quelques points essentiels.

I.3.1. Baser la Mise-en-pages sur de nouveaux principes

Fort des enseignements puisés dans les expériences antérieures, il nous a semblé nécessaire d'adopter de nouveaux principes issus de la signification profonde qui est attachée à la notion de Mise-en-pages.

La Mise-en-pages peut, en effet, être considérée comme le moyen de matérialiser, par une implantation différenciée, les rapports logiques existant entre les composants d'une même Edition.

Ainsi est née une nouvelle tendance qui n'est pas sans rappeler celle à laquelle se sont ralliés les chercheurs de l'Université de Montréal dont les travaux actuels ont certains rapports avec les nôtres [VE-1-1].

L'origine de cette tendance ainsi que les concepts nous ayant permis de l'appliquer font l'objet du chapître suivant.

I.3.2. Etablir une méthode de l'Edition

Permettre de décrire une Mise-en-pages sans qu'il soit nécessaire de se préoccuper ni des dimensions des informations, ni de celles des pages servant de support, telle est notre intention.

La conception de notre solution doit dès lors être dominée par une idée-maîtresse : Faire abstraction de la notion de dimensions.

C'est en abolissant, ainsi, la plus lourde des contraintes, que nous ramenons la réalisation de toute Mise-en-pages à l'accomplissement de deux tâches nettement différenciées :

- La description, confiée au programmeur de façon à ne pas nuire à l'influence artistique qui se manifeste dans tout imprimé. Elle se ramène désormais à la traduction de relations logiques entre les composants.

  Ainsi, par hypothèse,

  * les composants d'un imprimé : les chapîtres,
  * les composants d'un chapitre : les textes, les intitulés, les notes, les sous-titres, les tableaux récapitulatifs,

  sont des "entités" assimilables à des groupements organisés d'informations, unies entre elles par des relations logiques.

- La concrétisation, confiée à l'ordinateur auquel incombe, de ce fait, deux rôles essentiels :

* L'ordinateur : outil d'aide à la décision
  En assumant les tâches routinières qui régissent la Mise-en-pages et par suite conditionnent l'Edition, il lui est possible de proposer une solution optimale, compte tenu des directives qui lui ont été fournies.

* L'ordinateur : outil de mise-en-application.
  En respectant la Mise-en-pages qu'il a préalablement déterminée, il lui est possible de réaliser l'Edition effective dans la mesure où l'utilisateur donne son accord.

En proposant,ainsi, une nouvelle répartition des tâches nous envisageons un mode de raisonnement totalement différent de celui présidant, jusqu'ici, à toute Edition par "Ordinateur interposé".

Cette méthode offre l'avantage de guider le programmeur dans sa démarche, quelle que soit la complexité de l'Edition envisagée.

En effet,

- la diversité des liens logiques qu'il est possible d'établir dans le cadre d'une même Edition, conduit le programmeur à traduire ses désidérata sous la forme d'étapes distinctes et de complexité souvent croîssante.

- La considération des liens logiques primant sur les contraintes physiques, ce n'est que lors de la concrétisation des étapes préalablement décrites que le programmeur communique les dimensions du support.

REMARQUES :

- Ces indications justifient le rôle que nous attribuons à l'ordinateur. Il serait impensable de disposer d'une gamme suffisante de "maquettes" précodées pour faire face aux diverses dispositions envisageables.

- Conséquences inéluctables de telles options : les Mises-en-pages proposées par l'ordinateur doivent être en accord avec celles que l'esprit humain aurait pû déterminer, dans les mêmes circonstances, au prix de multiples calculs.
  Pour tendre vers ce but, nous devions :
  * du point de vue théorique, approfondir l'étude des processus conventionnels (objet du chapitre suivant).
  * du point de vue pratique, prévoir des "réglages" (Cf.architecture du Système M.P.E.3ème partie de la Thèse, Chapitre 3).

I.3.3. Elargir la gamme des informations concernées par les opérations de Mise-en-pages

Afin de vérifier l'efficacité des solutions que nous allons proposer, il nous fallait les soumettre à une utilisation fréquente et les appliquer à des types d'information présentant une grande diversité. Nul champ d'application ne pouvait être plus favorable que les données traitées par un langage évolué.

Soit,un tableau rassemblant un nombre important de valeurs identiques. En assurer l'Edition, et par conséquent la Mise-en-pages, impose à tout programmeur d'envisager une succession d'opérations :

a) Définir la représentation; ce qui revient à préciser :
   - la répartition la plus favorable à l'interprétation des valeurs ainsi regroupées.
   - la signification des éléments constitutifs à l'aide de commentaires (intitulés de lignes et de colonnes).
   - la position et la composition appropriée des traits de séparation différenciant les divers sous-ensembles de valeurs.

b) Définir la Mise-en-pages, qui se ramène dans le cas présent à une décomposition homogène en sous-tableaux, dans la mesure où l'intégralité de l'information considérée ne peut être imprimée sur une même page.

   Décomposition délicate car, ne devant pas nuire à l'interprétation de tels résultats, elle nécessite :

   - de tenir compte de l'organisation du tableau initial, de façon à obtenir des sous-tableaux significatifs.
   - de rappeler, pour chaque sous-tableau, les intitulés de lignes et de colonnes.

Il est intéressant de remarquer que de telles opérations de Mise -en-pages peuvent s'avérer bien plus délicates encore.

Supposons simplement que l'intelligibilité des imprimés à obtenir soit conditionnée par la coexistence d'informations distinctes sur la page où figure le tableau ou sur chacune de celles où figure un sous-tableau.

En appliquant à cet exemple les orientations adoptées (Cf §1-3-1 et §1-3-2), il nous est possible, dès à présent, de discerner les nouvelles

attributions du programmeur (description et décision) et celles dévolues à l'ordinateur (interprétation, proposition et réalisation).

Nous pouvons ainsi mesurer les avantages qu'en retirera l'utilisateur et, en revanche, la complexité de la tâche incombant désormais à l'ordinateur.

Outre le respect de l'idée maitresse, cette complexité se trouve donc accentuée par la diversité des composants d'une édition et par la nature des liens qui peuvent les unir dans le cadre d'une mise-en-pages.

I.3.4. Distinguer nettement Données et Traitements

De façon à permettre à une personne autre que celle ayant écrit le programme d'avoir aisément un aperçu du traitement et de procéder sans risques à des modifications, il est essentiel de distinguer données et traitements.

Une telle conception représente un facteur important de souplesse en ce sens qu'un même ensemble de données peut être soumis à des dispositions différentes et, qu'inversement, une même disposition peut être appliquée à des ensembles différents de données.

I.3.5. Considérer l'étude actuelle comme le prélude à des extensions prometteuses

L'ordinateur n'a pas évolué aussi rapidement dans tous les domaines. S'il est doté de moyens exceptionnels en matière de traitement, de stockage et de sélection d'informations de types variés, il souffre d'un handicap: la communication avec le milieu extérieur (saisie de données, communication des résultats). Ainsi, a-t-on recours à l'utilisation de supports tels que cartes ou bandes perforées, clavier de terminal, papier d'imprimante, écran de visualisation alphanumérique ou graphique, etc...

Les travaux actuels ont tendance à rapprocher les moyens de communication homme-machine de ceux existant entre les hommes :

- Communication orale

  * Reconnaissance de la parole (saisie des données).
  * Ordinateur parlant (communication de résultats).

- Communication visuelle

  * Lecture optique (saisie des données).
  * Affichage sur écran de visualisation, ou obtention d'imprimés ayant la même physionomie que ceux obtenus par les procédés évolués de l'imprimerie.(Communication des résultats)[WA-1-1]

C'est dans le cadre de cette dernière orientation que se situe notre étude. Son aboutissement devrait contribuer non pas à mettre l'ordinateur au service de l'imprimerie, comme cela est fréquemment le cas, mais de mettre les techniques avancées de l'imprimerie au service de l'ordinateur.

Grâce à une telle évolution des méthodes de représentation, de Mise-en-pages et d'Edition, il sera possible de considérer l'impression des textes, des tableaux ou des structures de données, la réservation d'emplacements destinés à l'implantation de schémas, comme les résultats d'un traitement décrit à l'aide d'un langage évolué.

L'Edition offrira alors toutes les caractéristiques des imprimés composés par un Editeur et se présentera sous une forme adaptée à une reproduction tant rapide que de qualité (génération d'une bande magnétique destinée à une photocomposeuse).

Le fait de nous intéresser, en un premier temps, à la représentation, à la Mise-en-pages et à l'impression des divers types de données, dont un langage évolué peut assurer le traitement, constitue à maints égards une délimitation intéressante de notre étude :

- Viser à l'essentiel en éliminant tout problème marginal, ou par ailleurs résolu, tels que, respectivement, l'utilisation de "diverses polices de caractères" et la justification à droite.

- Autoriser la mise en application rapide des solutions retenues de façon à évaluer leur efficacité et à recueillir les renseignements indispensables à une évolution prévisible.

I.4. LES ASPECTS FONDAMENTAUX DE LA SOLUTION RETENUE

I.4.1. Le choix d'un langage évolué

Les informations de types les plus variés que nous envisageons de traiter devant être fournies par un langage évolué, PL/I a retenu notre attention.

Plusieurs raisons ont motivé notre choix.

PL/1,

- autorise le traitement d'une grande variété d'informations.
- représente, à notre avis, le langage évolué actuellement implémenté qui s'adapte le plus aisément à des traitements très différents (gestion, calcul scientifique, traitement de caractères etc...)
- a déjà fait l'objet de nombreuses extensions [CR-2-1][LA-2-1] auquel il se prête en raison de l'accès facile aux divers renseignements contenus dans les DOPES-VECTORS.

Par contre, ce choix présente des inconvénients :

- pour l'utilisateur potentiel des dispositifs que nous proposons, puisque nous lui imposons d'avoir une connaissance, pour le moins élémentaire de PL/1.
- pour nous-même , puisqu'il nous a fallu tenir compte, tant pour la définition du langage spécialisé (§I.4.3.) que pour la nature du texte généré par le pré-compilateur, des possibilités nombreuses et par suite contraignantes (du point de vue implémentation) de PL/1.

I.4.2. La conception d'un système spécialisé: "le "système M.P.E."

Le compilateur d'un langage est un organe complexe et difficilement modifiable.

Il ne pouvait donc être question d'envisager l'incorporation dans le compilateur PL/1 de dispositifs assurant la mise en application des nouvelles méthodes d'Edition proposées.

La conception d'un système spécialisé, dénommé "système de Mise-en-pages et d'Edition" ou plus brièvement "Système M.P.E." se présente comme une solution intéressante à plus d'un titre :

- En tant que système totalement indépendant du compilateur, il n'augmente nullement "l'espace mémoire" nécessaire à la compilation d'un programme. Sa mise en oeuvre est déclenchée lors de l'exécution du programme objet.

- En tant que système additionnel, il ne neutralise aucunement les possibilités bien connues du langage.

- En tant que système spécialisé, il regroupe des modules à chacun desquels est dévolu un rôle précis (déterminer la surface libre pouvant supporter une impression, déterminer les découpes à faire subir à des informations composées, etc...).

Une telle organisation minimise la zone mémoire occupée, puisque seuls les modules à mettre en oeuvre, à un instant donné, sont "résidents". Elle présente d'autre part l'avantage de favoriser l'évolution du système puisque toutes modifications ou adjonctions peuvent être aisément réalisées.

Il peut néanmoins sembler surprenant que ce système dans sa version actuelle ne soit pas intéractif.

Il ne s'agit là ni d'une faute de conception, ni d'un refus d'avoir recours à l'aspect conversationnel, étant convaincu des avantages que ce dernier peut nous offrir :

* Prise de décision rapide
* Traduction aisée, grâce à l'utilisation d'un écran graphique, de contraintes physiques relevant de considérations esthétiques.

Mais, il nous fallait concevoir un prototype destiné à être soumis à l'appréciation d'un grand nombre d'utilisateurs. L'orientation, présentement adoptée, est simplement conforme aux objectifs que nous nous sommes fixés dans l'immédiat (Cf §I.3). Nous considérons néanmoins comme inéluctable le recours à cette technique, à plus d'un titre séduisante.

Une telle évolution :

* N'entraînera pas des modifications fondamentales du système M.P.E. puisque l'aspect algorithmique de L.A.M.P.E.(Cf I.4.3.) permet, dès à présent, de prendre des décisions identiques en s'appuyant sur les mêmes considérations.

* Ne pourra être envisagée que lorsque les résultats escomptés auront été fournis par cette version expérimentale.

I.4.3. L'élaboration d'un langage: L.A.M.P.E."

La mise en oeuvre des différents modules qui constituent le "système M.P.E." est régie par des paramètres. Ceux-ci résultent de la codification des directives que le programmeur fournit afin de préciser, conformément à la Méthode d'Edition proposée, la Mise-en-pages et la représentation auxquelles doivent satisfaire les informations sélectionnées.

- Pour faciliter la communication entre le programmeur et le "système M.P.E.", l'idée de définir un langage spécialisé s'est imposée; un pré-processeur se voit confier la tâche de traduire ces directives.

- Pour offrir toute la souplesse de description nécessaire, nous avons été conduit à envisager la conception d'un langage autorisant l'écriture d'algorithmes de Mise-en-pages et d'Edition.

D'où sa dénomination qui résume ses trois caractéristiques essentielles :

- Langage algorithmique :
  Seule la description d'algorithmes donne actuellement la possibilité à l'utilisateur, de traduire l'ensemble de ses requêtes, de tester les configurations proposées, et de faire connaître la nature de son choix.

- Langage de Mise-en-pages :
  Toute Edition est tributaire d'une Mise-en-pages. La détermination de cette dernière est confiée, désormais, au système M.P.E. Le programmeur doit néanmoins décrire, à l'aide de L.A.M.P.E., les diverses contraintes auxquelles doit satisfaire l'implantation des informations sélectionnées, au sein des pages nécessaires à l'Edition.

- Langage d'Edition :
  Le système M.P.E. assure la concrétisation de la Mise-en-pages qu'il a préalablement déterminée, en réalisant l'Edition effective dans la mesure où le programmeur, à l'aide de L.A.M.P.E., lui précise son accord.

PREMIERE PARTIE

C H A P I T R E II

PRESENTATION
DU LANGAGE L.A.M.P.E. ET DU SYSTEME M.P.E.

Langage spécialisé de par les traitements qu'il autorise, langage de commandes de par la destination des directives qu'il permet d'exprimer, L.A.M.P.E. est un langage évolué de par le contexte dans lequel il doit s'intégrer.

Ces trois caractéristiques, ou plutôt ces trois aspects du langage, ont été à l'origine de nombre de difficultés qu'il a fallu surmonter pour essayer de donner à L.A.M.P.E. une conception "saine" et susceptible de ne pas être altérée par les évolutions envisagées.

La présentation des solutions dictées par l'idée maitresse et l'exposé des motivations de tels choix, font l'objet du présent chapitre.

2.1. LA CONCEPTION DE L.A.M.P.E. EN TANT QUE LANGAGE SPECIALISE

Principale difficulté : respecter les deux orientations adoptées :

* Ramener la description de toute édition à l'établissement de liens logiques entre les informations concernées.
* Assurer la description en faisant volontairement abstraction de la notion de dimensions (idée maitresse).

2.1.1. Les techniques de l'Edition, un modèle pour L.A.M.P.E.

Du fait de la spécialisation de ce langage, il nous a paru intéressant, pour le concevoir, de nous inspirer, après les avoir analysés en détail, des méthodes qui président, dans l'Imprimerie, à l'Edition de livres, de rapports ou de tous documents destinés à une grande diffusion.

Schématisant le processus, nous pouvons identifier l'Edition à la réalisation d'un nombre limité d'opérations, biens définies et s'enchaînant logiquement.

En effet, traditionnellement, le manuscrit est confié à un Editeur qui a la charge d'élaborer des "maquettes" ou documents représentant la structure de chacune des pages de l'imprimé final.

Une fois obtenues, ces maquettes sont remises à l'Imprimeur qui effectue, à partir d'originaux, le tirage final.

2.1.1.1. Le rôle de l'Editeur

La constitution des maquettes résulte de la disposition,dans des pages de dimensions généralement imposées, d'éléments fournis par le manuscrit et de natures diverses:textes, tableaux, figures, schémas, illustrations, etc..

Ce serait minimiser énormément le rôle de l'Editeur que de le ramener à une simple opération de disposition. En réalité, une tâche bien plus délicate, nécessitant un esprit d'initiative et même certains dons artistiques, lui incombe.

"L'art de la Mise-en-pages relève des même règles que les autres arts plastiques".(De LABORDERIE)[LA-4-1].

Il nous a été possible de répartir, en trois phases d'égale importance, les travaux conduisant à la conception d'une "maquette".

Nous les désignerons respectivement par les appellations suivantes : REPRESENTATION, COMPOSITION, MISE-EN-PAGES.

a/ La représentation

"Action d'exprimer matériellement à l'aide de moyens graphiques ou plastiques" (Larousse);

En procédant à la représentation d'éléments fournis par le manuscrit, l'Editeur définit l'aspect qu'il convient de leur donner, dans le cadre de l'imprimé, afin de faciliter leur interprétation et de rendre plus agréable leur consultation.

- L'aspect des éléments à éditer peut être régi par des règles.
  Tel est le cas de la représentation des textes pour lesquels nombre d'opérations résultent de l'application de règles précises :

  * Décalage à chaque alinéa
  * Règles de troncature du dernier mot d'une ligne lors de la justification à droite, etc...

- L'aspect des éléments à éditer peut être régi par des considérations de logique et d'esthétique,laissées à la seule initiative de l'Editeur.
  Nous citerons par exemple les deux cas suivants :

  .Représenter un tableau, quelle que soit la nature des informations qu'il regroupe, revient à choisir la répartition la plus "parlante" de ses composants, et à différencier, au moyen de traits, les lignes, les colonnes et leurs intitulés.

  . Représenter un schéma, c'est adapter sa configuration et ses proportions aux dimensions de la page qui supportera son impression.

b/ La composition

"Action de former un tout de différentes parties" (Larousse).

Pour effectuer des compositions, l'Editeur prend en considération les éléments obtenus au terme du traitement précité.

Il soumet ces éléments, en vertu tant du contexte que de la nature des renseignements qu'ils renferment, à des regroupements ou associations logiques destinées à matérialiser l'interdépendance existant entre eux.
Lors de la consultation d'une publication, il nous semble, en effet, normal,

* de voir figurer, immédiatement en dessous d'une illustration, le commentaire qui s'y rapporte,
* de voir figurer dans le texte, afin de renforcer les affirmations qu'il comporte, une figure ou un schéma explicatif,
* de voir figurer un titre avant le texte qu'il identifie.

De tels regroupements n'excluent pas, cependant, la possibilité de mettre en valeur certains des éléments qui le composent.
C'est ainsi que le titre d'un paragraphe peut être souligné et parfois décalé par rapport au texte, qu'une observation particulièrement importante peut être encadrée, etc...

Concrètement, la "composition" est une phase de l'Edition qui permet de définir la position relative, les uns par rapport aux autres, d'éléments très divers, et de conférer un caractère distinctif à certains d'entre eux.

c/ La mise-en-pages

"Action de rassembler des paquets de composition et d'en former des pages" (Larousse).
En réalisant la mise-en-pages, l'Editeur s'acquitte de l'ultime phase au cours de laquelle il adopte l'implantation optimum, au sein du support de l'impression, des éléments ou partie d'éléments, ayant fait l'objet d'une représentation et ayant éventuellement participé à une composition.
Etape fondamentale, s'il en est une, puisque de la coexistence sur certaines pages de plusieurs éléments distincts, de leur mise en évidence, de leur cadrage, de l'aspect harmonieux que revêt leur répartition, dépendent l'intelligibilité et la clarté de l'imprimé.
Prenons pour justifier nos dires, quelques cas probants :

* Le titre d'un livre est généralement rappelé en tête de chacune de ses pages.
* Le début d'un chapitre et son titre sont toujours mentionnés sur une même page.
* Toutes les explications afférentes à la partie d'un texte imprimé sur la page courante sont placées au bas de cette dernière et non en fin de chapitre.

REMARQUE :

Raison d'une telle décomposition.

"Tout ce qui se conçoit bien s'énonce clairement" (Boileau).

Mais, tout ce que l'esprit humain conçoit aisément, ne s'exprime pas aussi aisément dans le langage des ordinateurs.

C'est la raison pour laquelle on est enclin à pousser dans le détail l'analyse de tout processus susceptible d'être confié à une machine et destiné, de ce fait, à être exprimé sous forme algorithmique.

Ainsi, lorsque l'on parle de "Mise-en-pages" on ne fait pas habituellement de distinction entre les deux phases suivantes :

- La phase que nous dénommons "Composition" (qui n'a aucun rapport avec la "composition typographique"), au cours de laquelle est établi un lien logique entre les informations disjointes sans que soient prises en considération les dimensions de la page,

- La phase "Mise-en-pages" proprement dite, au cours de laquelle la disposition effectuée est principalement axée sur l'inclusion dans des limites imposées.

Prenons par exemple, la mise-en-pages d'un chapitre.

Tenant compte des dimensions de la page, l'Editeur dispose, sur la première, le titre du chapitre et la portion de texte qui peut y trouver place. Puis, il poursuit, sur les pages suivantes, l'implantation du contenu du chapitre.

En réalité, simultanément et logiquement, l'Editeur a associé le Titre au chapitre ("Composition") et a étudié la possibilité la plus favorable de disposer cette entité sur des supports de dimensions finies ("Mise-en-pages").

Mais l'imbrication de ces deux étapes est telle que l'esprit humain a tendance, dans l'usage courant, à ne pas faire cette discrimination, car des considérations esthétiques le conduisent à revenir sur la composition lors de la Mise-en-pages et que cela ne constitue pas un algorithme.

2.1.1.2. Le rôle de l'imprimeur

L'Imprimeur nous semble assumer une fonction plus routinière,influencée par l'expérience acquise.

Cette fonction consiste en effet à choisir compte-tenu des maquettes fournies par l'Editeur, et tout en respectant les limites de coût imparties, la trame du papier, la nature de l'encre, les polices de caractères les mieux adaptées à la transcription du manuscrit.

Puis, utilisant les procédés électro-mécaniques ou électroniques mis à sa disposition, l'Imprimeur réalise le tirage.

2.1.1.3. Synthèse

De la brève étude, volontairement schématisée, que nous venons d'exposer,se dégagent deux aspects dont l'intérêt est fondamental pour la suite de notre exposé :

* "L'aspect description" qui regroupe les différents travaux dont s'acquitte l'Editeur.
* "L'aspect traitement" qui s'identifie aux opérations réalisées par l'Imprimeur.

2.1.2. Les techniques d'Edition adoptées par L.A.M.P.E.

Si nous avons essayé de schématiser la technique de l'Edition, c'est afin de fonder la conception de notre langage sur des bases éprouvées et des notions connues. Cela dans un double but :

- D'une part, pour permettre au futur utilisateur de se familiariser plus rapidement avec les nouveaux concepts auxquels L.A.M.P.E. fait appel, en l'incitant à établir une analogie avec des méthodes usuelles.

- D'autre part, afin d'envisager des extensions ultérieures de L.A.M.P.E. et de son champ d'application.

En vertu du parallélisme entre les techniques de l'Edition et les

principes de notre langage, celui-ci présente également deux aspects :

- Un aspect "description"qui permet aux programmeurs de traduire leurs directives concernant représentations, compositions et mises-en-pages.

- Un aspect "traitement" qui assure la concrétisation des descriptions préalablement faites et offre la possibilité de demander la mise en action de l'imprimante ou de différer l'impression en procédant à un stockage sur support magnétique.

2.1.2.1. La nature des informations traitées

L.A.M.P.E. admet PL/1 pour langage hôte.

Toute information pouvant être soumise au traitement d'un programme PL/1, peut être soumise à celui traduit par des instructions L.A.M.P.E.

Afin de mieux fixer les idées, nous rappelons les principaux types de données existant en PL/1 [IB-3-2].

- Les informations simples:
  * Scalaires
  * Chaînes de caractères
  * Chaînes de bits

- Les informations composées,

ainsi désignées du fait qu'elles résultent du regroupement de plusieurs informations simples de natures identiques ou différentes, se répartissent en deux catégories :

  * Tableaux
  * Structures

2.1.2.2. L'aspect description du langage

a/ Assurer la description des "Représentations"

Aucune information ne peut être éditée si une représentation ne lui a été implicitement ou explicitement conférée. A cet effet une distinction existe entre informations simples et informations composées.

- Considérons des informations simples

Que ce soit des scalaires, des chaînes de caractères ou des chaînes de bits, leur aspect lors de l'Edition est essentiellement dicté par leur description initiale (déclaration PL/1). La description d'une "représentation" ne peut avoir, pour elles, qu'un caractère optionnel. Ainsi, pour éditer un nombre fractionnaire, il est possible mais non nécessaire de fixer le nombre de décimales à prendre en considération et les séparer de la partie entière par un point, de pouvoir supprimer les chiffres non significatifs,... etc....

- Considérons des informations composées

Lors de l'édition, la disposition des composants peut différer de celle adoptée lors de la déclaration. De même, à la différence de la configuration-mémoire, une séparation explicite des composants est généralement requise. La description d'une "Représentation" est dans ce cas nécessaire.

Prenons pour exemple (voir Figure 1.2.1.) le tableau conforme à la déclaration PL/I : "DCL TABLEAU (2,4,3) FIXED";"

Son édition impose au programmeur de prendre plusieurs décisions :

- Répartir lignes, colonnes, couches, de façon à en donner une représentation plane.(Le transformer en un tableau bi-dimensionnel).
- Définir les intitulés de lignes et de colonnes permettant de qualifier chaque élément constitutif.
- Implanter des traits de séparation pour clarifier et agrémenter son aspect général.

D'où, l'une des formes finales possibles représentées par la figure 1.2.2.

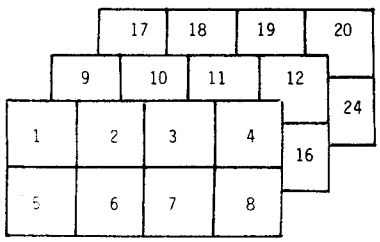


Figure 1.2.1.

	COUCHE-1				COUCHE-2				COUCHE-3			
	CL-1	CL-2	CL-3	CL-4	CL-1	CL-2	CL-3	CL-4	CL-1	CL-2	CL-3	CL-4
LIGNE-1	1	2	3	4	9	10	11	12	17	18	19	20
LIGNE-2	5	6	7	8	13	14	15	16	21	22	23	24

Figure 1.2.2.

b/ Assurer la description des "Compositions"

Disposant d'informations simples et d'informations composées, éventuellement soumises à des directives de représentation, possibilité est offerte par L.A.M.P.E. de définir des regroupements et des mises en valeur de certaines d'entre elles.

- La notion de regroupement renferme, outre l'idée d'association d'informations distinctes, celle de positions relatives des unes par rapport aux autres. Positions que le programmeur doit mentionner avec précision.
  Ainsi, par exemple, un tableau peut, **indifféremment**, être précédé ou suivi du texte précisant la signification des informations ainsi regroupées.

- La notion de mise en valeur s'identifie à l'utilisation d'artifices, traditionnellement utilisés à ces fins, et se traduit en encadrant, soulignant, décalant, l'information considérée.

BILAN ANNUEL DE L'ENTREPRISE

1/ COUT DE FONCTIONNEMENT DES DIFFERENTS SERVICES

EXERCICE CLOS AU 20/12/73.

	PERSONNEL		FOURNITURES	
	SEM-1	SEM-2	SEM-1	SEM-2
DEPT-ELECTRI.	20.000	28.000	85.000	87.000

Figure 1.2.3.

c/ Assurer la description des "Mises-en-pages"

Disposant d'informations simples et d'informations composées, préalablement soumises à des directives de "représentation" et parfois même à des directives de "Composition", procéder à une "Mise-en-pages" consiste à définir leur mise en évidence ou leur simple disposition au sein d'une ou de plusieurs pages.

- La notion de mise en évidence représente l'assignation à des informations au sein des pages, de positions qui, conformément à certains critères visuels, permettent de les signaler particulièrement à l'attention du lecteur.

- La notion de disposition traduit la désignation au système M.P.E., des informations qu'il aura la charge de répartir, au mieux, sur des pages de "listing", guidé en cela par l'importance attribuée à chacune d'elles, dans le cadre de l'édition envisagée.

Pour atteindre ces buts, il nous fallait nous appuyer sur des lois précises. Or :

"En cinq siècles d'imprimerie, on ne trouve que très peu d'observations qui témoignent explicitement d'une connaissance claire des effets de l'imprimerie sur la sensibilité humaine".[MO-4-1].

Ainsi, les règles auxquelles il est possible de se conformer sont elles empiriques. Elles résultent de l'étude des différentes perceptions visuelles consécutives aux variations de disposition de mêmes éléments au sein d'une même page.

Nous avons, en ce qui nous concerne, adopté des principes déduits de ceux régissant actuellement la "Mise-en-pages" de la plupart des imprimés courants (journaux, polycopiés,...etc...).

Organisation des pages

Le support de l'édition est décomposé en deux zones (voir figure 1.2.4) :

- La zone"en-tête" destinée à recevoir toute information caractéristique de l'ensemble de l'imprimé,
- La zone "corps de la page" destinée à recevoir toutes les autres informations.

Organisation du corps de la page

La page se décompose en diverses zones (voir figure 1.2.5) :

- Les "points forts", désignation d'emplacements dont la propriété est de retenir de prime abord l'attention du lecteur. Ils sont destinés à recevoir des informations qui, en raison de leur rôle prépondérant dans le cadre de l'édition envisagée, doivent être distinguées. Les points forts sont répartis aux quatre coins du corps de la page. N'ayant pas tous le même pouvoir attractif, leur classification est conforme à la numérotation incluse dans la figure suivante.
- La "zone banalisée", zone demeurant libre lorsque les informations à mettre en évidence ont été implantées, et au sein de laquelle sont disposées les informations non prépondérantes.

Ainsi,

- METTRE EN EVIDENCE, revient à fixer l'implantation d'une information, ou d'un groupement d'informations (entité), dans l'une des zones privilégiées, en l'occurrence la zone "en-tête" ou les points forts. Dans ce dernier cas, le choix de l'emplacement peut être indifféremment mentionné de façon explicite par le programmeur ou laissé au soin du système M.P.E.

- DISPOSER, revient à implanter des informations, prises séparément ou regroupées (entités), au sein de la zone banalisée. Dans ce cas, c'est le système M.P.E. qui effectue la répartition des informations en fonction de l'importance accordée à chacune d'elle par le programmeur, et exprimée sous la forme d'une priorité relative.

Un principe essentiel régit cette répartition :

De deux entités se faisant suite horizontalement, celle figurant le plus à gauche est considérée comme prioritaire.

De deux entités se faisant suite verticalement, celle placée en haut est prioritaire.

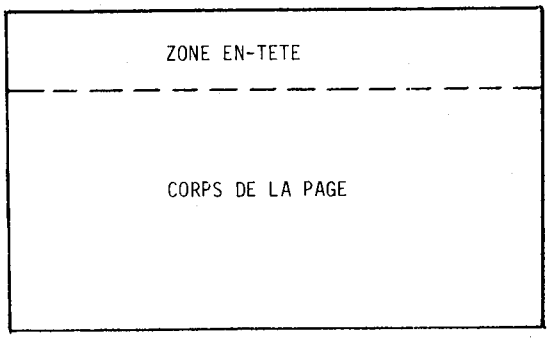


Figure 1.2.4.

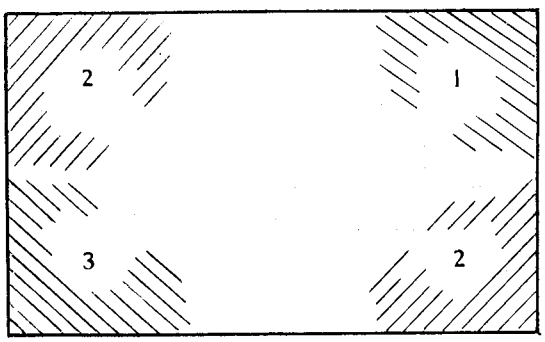


Figure 1.2.5.

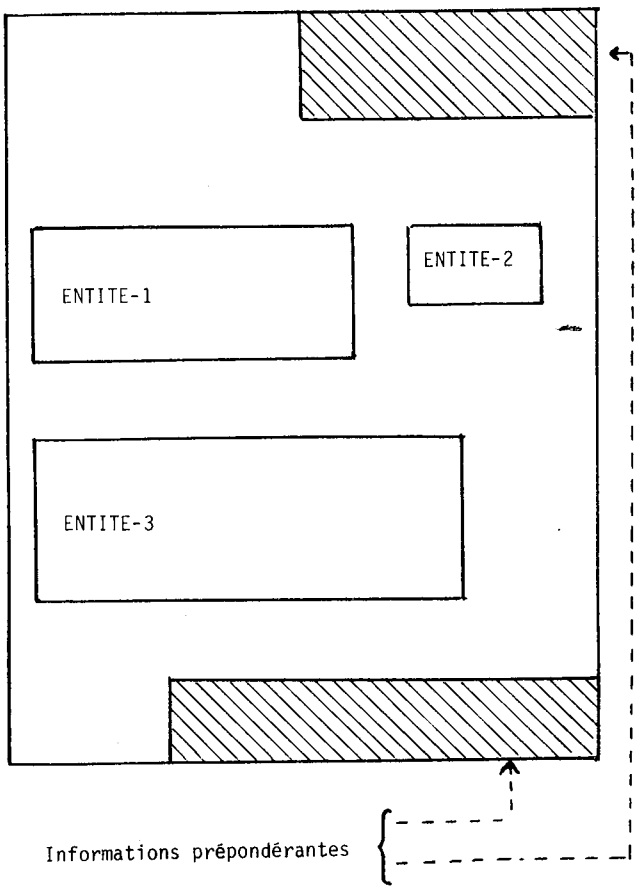


Figure 1.2.6.

Application

Prenons, comme exemple concret, la mise-en-pages imposée par l'Edition d'une facture.

Instinctivement, et en tout premier lieu, on est conduit à faire un bilan des renseignements qui doivent être transcrits afin de conférer à cette pièce officielle un intérêt, tant pour le client que pour le fournisseur.

Doivent donc y figurer :

* Les nom et adresse du client,
* Le montant de la facture et, éventuellement, l'indication de la remise consentie,
* La liste des objets commandés, leur nombre et leur prix,
* Les dates de commande et d'expédition,
* Le numéro d'identification du client nécessité par l'automatisation de la gestion de l'entreprise.

Loin de procéder, ainsi, à une simple énumération, ce sont deux opérations logiques qui ont été successivement réalisées :

* L'association d'informations distinctes mais complémentaires,
* Leur classement en fonction de l'importance qui leur est intuitivement accordée.

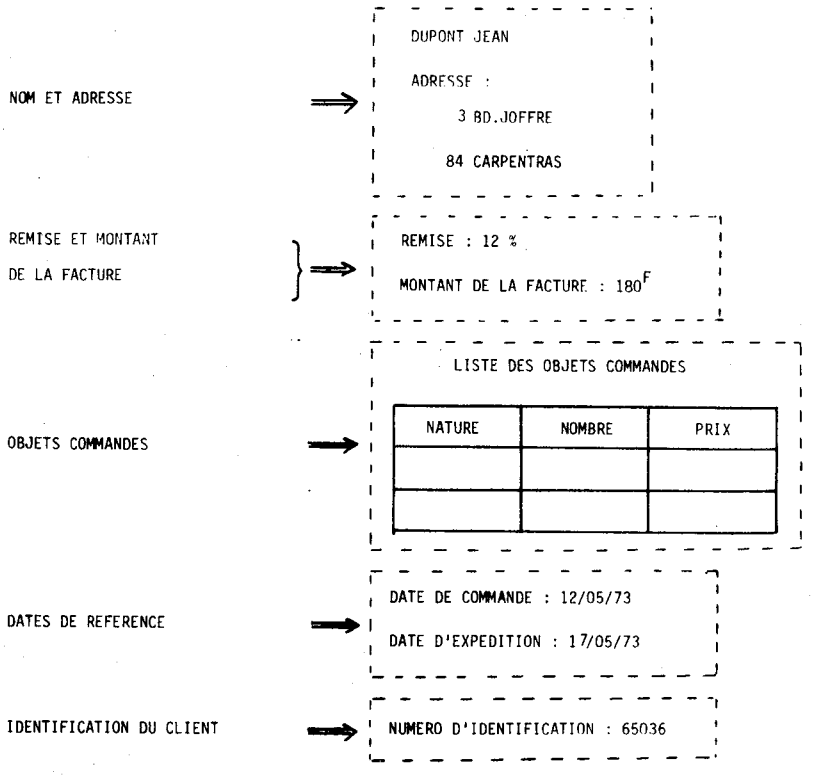


Figure 1.2.7.

En vertu de l'importance de ces entités, dans le cadre de la mise-en-pages, et, en admettant que nous désirions mettre en évidence "nom et adresse", ainsi que "remise et montant", nous obtiendrons l'imprimé suivant :

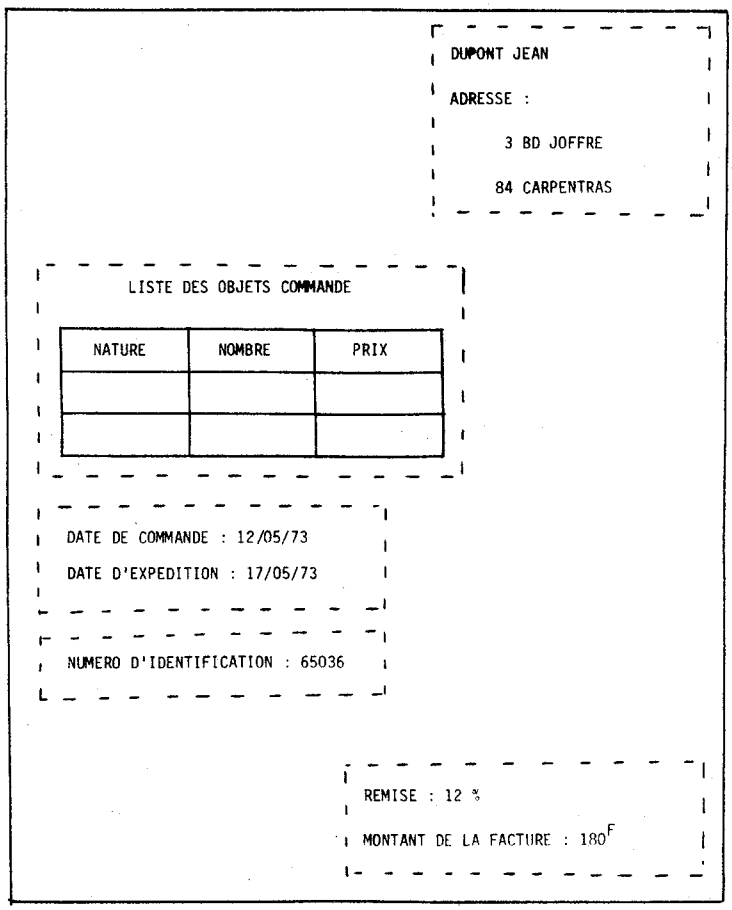


Figure 1.2.8.

2.1.2.3. L'aspect traitement du langage

Sous cet aspect, L.A.M.P.E. permet de déclencher le tirage, en un ou plusieurs exemplaires, des imprimés élaborés au cours des phases antérieures.

Il lui incombe de provoquer la concrétisation des directives de "Représentation", de "Composition" et de "Mise en pages" éventuellement adjointes aux informations sélectionnées.

Cette ultime phase se traduit par la mise en oeuvre du système M.P.E. qui, prenant acte des informations et de leurs directives d'Edition, procède :

* Soit, directement, au moyen de l'imprimante, à l'impression sur une ou plusieurs pages

* Soit, au stockage, sur disques ou sur bandes magnétiques, de façon à différer la génération des imprimés.

2.1.2.4. Synthèse

Les traitements spécialisés, assurés à l'aide de L.A.M.P.E., se trouvent résumés dans le schéma suivant :

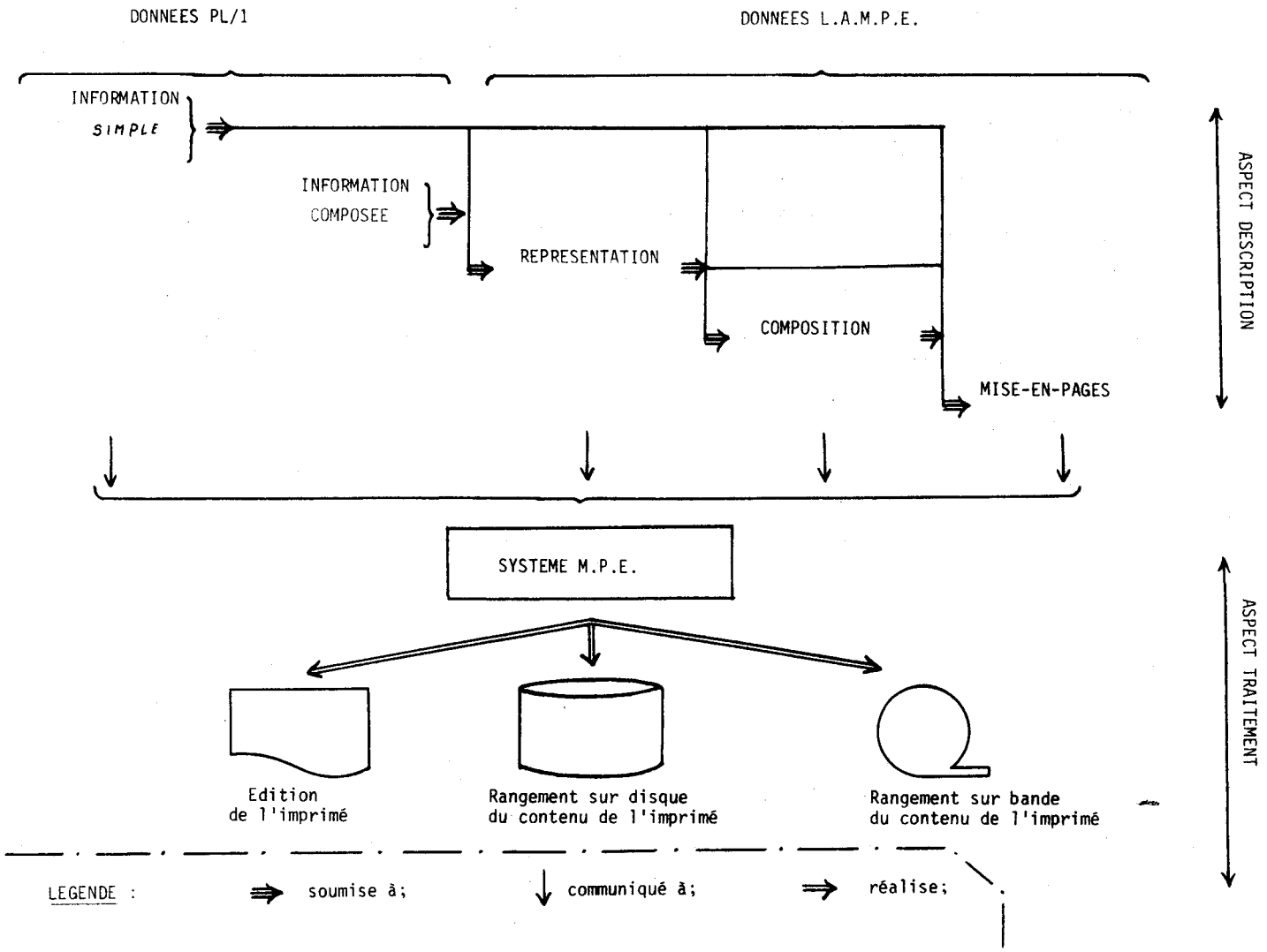


Figure 1.2.9.

2.1.3. Les concepts de base de L.A.M.P.E.

2.1.3.1. Le concept de "Rectangle-unité"

Considérons les deux résultats suivants obtenus à l'issue d'un traitement sur ordinateur

3.1416

	A	B	C	D
L	10	35	42	28
M	53	16	23	12

Figure 1.2.10.

Le premier correspond à l'édition d'une valeur numérique; le deuxième, à celle d'un tableau. L'un comme l'autre est conforme aux principes régissant toute représentation (Cf.§ 2.1.1.1.-a).

Une constatation commune s'impose: ces informations, de natures très différentes, sont toutes deux inscriptibles dans un rectangle.

3.1416

	A	B	C	D
L	10	35	42	28
M	53	16	23	12

Figure 1.2.11.

Toute information de base simple, soumise explicitement ou non à une "Représentation" (Cf.§ 2.1.2.2.a), de même que toute information de base composée, nécessairement soumise à une "Représentation" (Cf.§2.1.2.2.a), est qualifiée d'"Information Editable Simple" et a la propriété d'être inscriptible dans un rectangle.

Envisager l'Edition d'une telle information revient donc à considérer le plus petit rectangle dans lequel elle est inscriptible et dénommé "Rectangle-Unité".

CONCEPT 1

Enoncé : Toute information éditable simple est inscriptible dans un rectangle dénommé "Rectangle-Unité".

REMARQUE 1

L'élaboration d'un Rectangle-Unité résulte de l'application au texte d'une information issue d'un traitement PL/1:

* *D'opérations aptes à lui conférer un aspect conventionnellement admis*
* *D'artifices traditionnnellement utilisés pour améliorer sa représentation.*

2.1.3.2. Le concept de "Rectangle-Entité"

Conformément aux principes régissant toute Composition (Cf.§ 2.1.1.1.b), des informations éditables simples peuvent être indifféremment soumises :

* à des regroupements au sein desquels elles occupent des positions explicitement mentionnées par le programmeur (Cf.§ 2.1.2.2.b),
* à des mises en valeur (Cf § 2.1.2.2.b)

Ces contraintes,

- sont traduites, respectivement, sous la forme :
  * de liens logiques établis entre les informations participant à une Composition et destinés généralement à exprimer la complémentarité de leur signification.
  * d'artifices destinés à améliorer l'aspect visuel et l'intelligibilité de l'imprimé final.
- portent, en vertu du concept-1, sur des rectangles (Rectangles-Unités).

EXEMPLE :

Rectangle-Entité =>

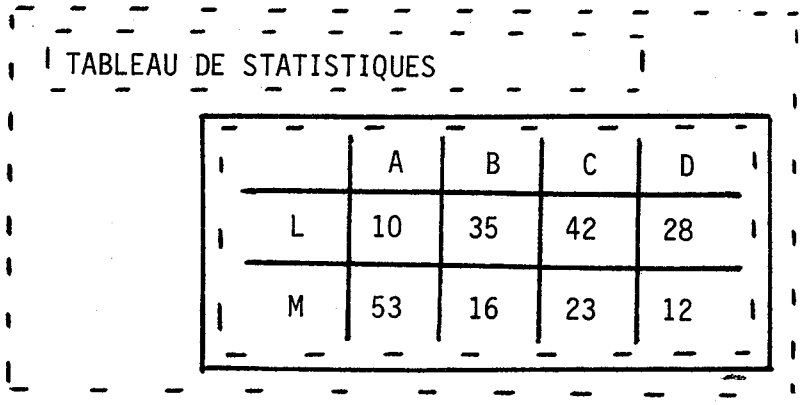


Figure 1.2.13.

Une telle configuration, ou entité, étant composée de rectangles, est également inscriptible dans un rectangle.

Envisager son édition revient à considérer le plus petit rectangle qui puisse la contenir et dénommé "Rectangle-Entité".

CONCEPT-2

Enoncé : Des informations éditables simples, à l'issue d'opérations de regroupement et de mise en valeur, sont inscriptibles dans un rectangle dénommé "Rectangle-Entité".

Schéma : RECTANGLE-ENTITE

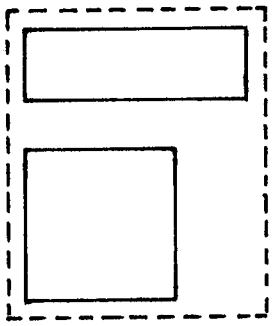


Figure 1.2.14.

N.B. En vertu du concept-2 :

- Un Rectangle-Entité peut être constitué d'un seul Rectangle-Unité. Il s'identifie alors à un Rectangle-Unité.

- Un Rectangle-Entité peut regrouper plusieurs Rectangles-Unités. Les liens qui les réunissent matérialisent la disposition rigide des uns par rapport aux autres (Cf.§1.2.2.2.b) et l'indépendance, de leur association, de tout contexte extérieur.

En conséquence, quelle que soit sa constitution, tout "Rectangle-Entité" peut figurer, au même titre qu'un Rectangle-Unité, au sein d'un autre "Rectangle-Entité".

Cette récursivité permet, aux Compositions, de traduire aisément la plus grande variété possible de configurations.

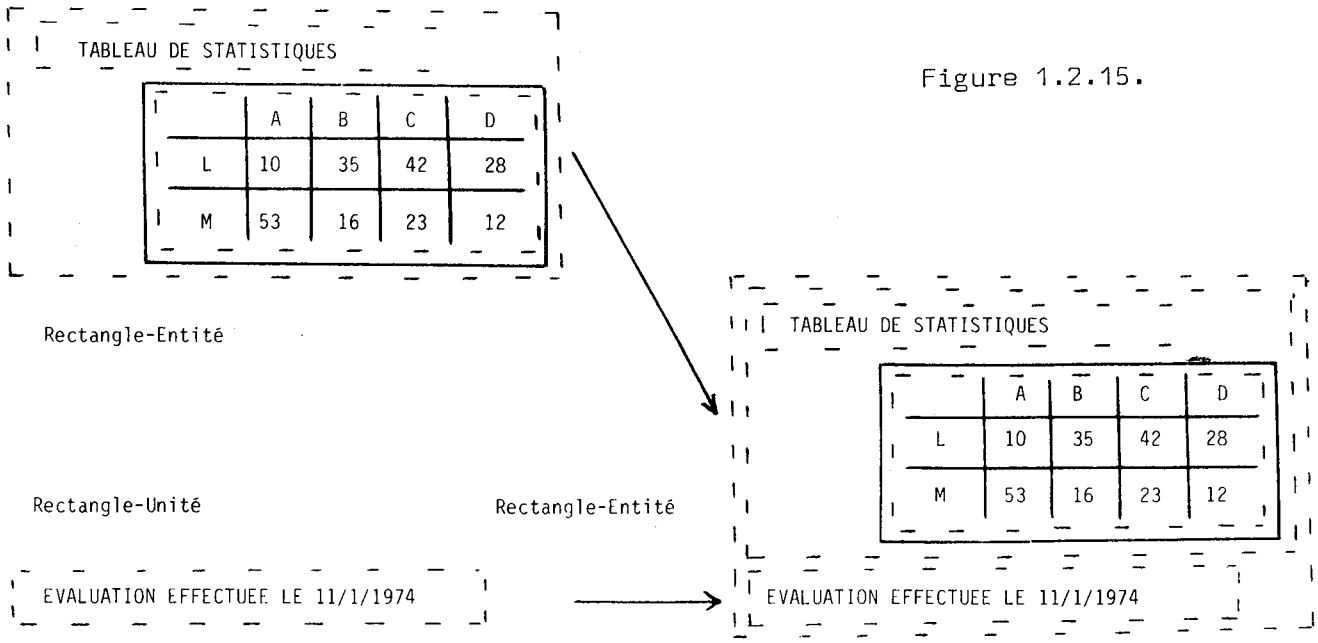


Figure 1.2.15.

REMARQUE 2

L'élaboration d'un Rectangle-Entité résulte de l'application indifféremment à des Rectangles-Unités et à des Rectangles-Entités :

- *D'opérations de composition; lesquelles, conformément à des lois de compositions internes, matérialisent, dans l'esprit de notre étude, les liens impliqués par la phase "Composition".*

- *D'artifices traditionnellement utilisés pour exprimer, dans tout imprimé, la mise en valeur de certains composants.*

2.1.3.3. Les concepts de "Page Virtuelle" et de "Pages-Formelles

Conformément aux principes régissant toute Mise-en-pages (Cf §2.1.1.1. des informations éditables simples ou des entités peuvent être indifféremment soumises :

* à des dispositions, au sein de l'imprimé, basées sur la priorité d'implantation (Cf.§ 2.1.2.2.c),
* à des mises-en-évidence (Cf §2.1.2.2.c).

Ces contraintes,

- sont traduites, respectivement, sous la forme :
  * de liens logiques établis entre les informations participant à la mise en pages et destinés à exprimer l'importance accordée à chacune d'entre elles.
  * D'artifices destinés à améliorer l'aspect visuel et l'intelligibilité de l'imprimé final.

- Portent, en vertu des concept-1 et concept-2 sur des rectangles (Rectangles-Unités et Rectangles-Entités).

Ces rectangles peuvent être, en fonction de la nature des informations qu'ils renferment, de dimensions différentes,et, en fonction de la nature de l'édition envisagée, en nombre variable.

Les contraintes précitées ne sont applicables, conformément à l'idée maîtresse adoptée, que dans la mesure où le support de l'Edition est considéré comme une surface rectangulaire de dimensions non bornées. Nous l'appelons "Page Virtuelle".

Cependant, le support effectif de l'Edition (ou Page-Physique) est, de toute évidence, constitué par une surface rectangulaire de dimensions finies. Une page physique est, en effet, nécessairement inscriptible dans une feuille de "listing" ou éventuellement dans plusieurs d'entre elles prises successivement.

Désignons par "Pages-Formelles" des pages destinées à jouer un rôle de transition entre la "Page-Virtuelle" et les "Pages-Physiques". Elles se caractérisent, à cet effet, par trois propriétés :

- *Elles doivent avoir des dimensions initialement identiques à celles imposées aux Pages-Physiques.*
- *Elles doivent être destinées à contenir, chacune, dans la limite de leurs dimensions, le maximum de Rectangles-Unités et Rectangles-Entités parmi ceux implantés dans une Page-Virtuelle, tout en respectant leur disposition.*
- *Elles doivent se présenter comme des pages "idéales", en ce sens que leurs dimensions sont susceptibles d'augmenter afin d'autoriser, en tout état de cause :*
  - ** Nécessairement, la disposition au moins d'un Rectangle-Unité ou d'un Rectangle-Entité dans son intégralité.*
  - ** Eventuellement, la coexistence de plusieurs d'entre eux explicitement désignés.*

Nous sommes ainsi amenés à distinguer, dans la description d'une Mise-en-Pages, deux étapes complémentaires :

- L'Implantation (fixant la mise en évidence et la disposition),
- La Répartition (fixant la coexistence).

a/ L'Implantation sur une "Page-Virtuelle"

L'Implantation permet d'appliquer, à chaque information participant à une Mise-en-Pages, les contraintes précitées, sans entrer conformément à l'idée maîtresse, dans des considérations de compatibilité de dimensions.

CONCEPT-3

Enoncé : Des informations éditables simples et des entités, à l'issue d'opérations de disposition et de mise-en-évidence, sont inscriptibles dans un rectangle de dimensions initialement non bornées, dénommé "Page-Virtuelle".

Schéma :

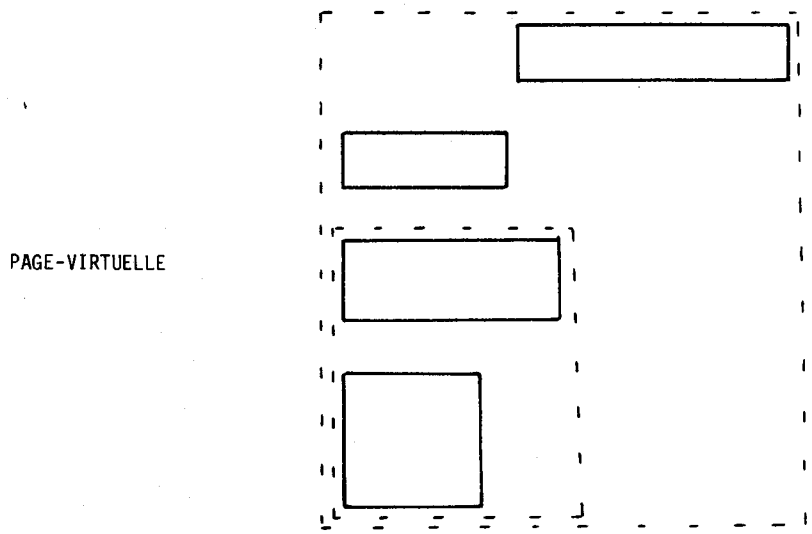


Figure 1.2.16.

Toute édition se ramène à l'impression de pages de dimensions finies (Pages-Physiques).
Il s'agit, dès lors, de transformer un ensemble structuré de rectangles implantés au sein d'un support rectangulaire fictif de dimensions, initialement non bornées, en un ensemble ordonné de surfaces rectangulaires fictive de dimensions initialement fixées.
La répartition permet d'appliquer à des informations ayant préalablement été soumises à des opérations de disposition et de mise en évidence, des restrictions imposant leur coexistence sur une même Page-Physique.
Ce qui revient à établir des liens, entre leur rectangle représentatif préalablement implanté dans une page virtuelle ; il s'agit donc, de façon à ne pas entrer dans des considérations de dimensions, conformément à l'idée maîtresse, à envisager l'élaboration de Pages-Formelles.

CONCEPT-4

Enoncé : Des informations éditables simples et des entités implantées dans une Page-Virtuelle, à l'issue d'opérations de coexistence, sont inscriptibles dans un ensemble ordonné de surfaces rectangulaires, de dimensions initialement imposées, dénommées "Pages-Formelles". Celles-ci sont destinées à contetenir tous les rectangles figurant dans la Page-Virtuelle, disposés dans le même ordre et assujettis aux positions de mise en évidence qui leur ont été assignées.

Schéma :

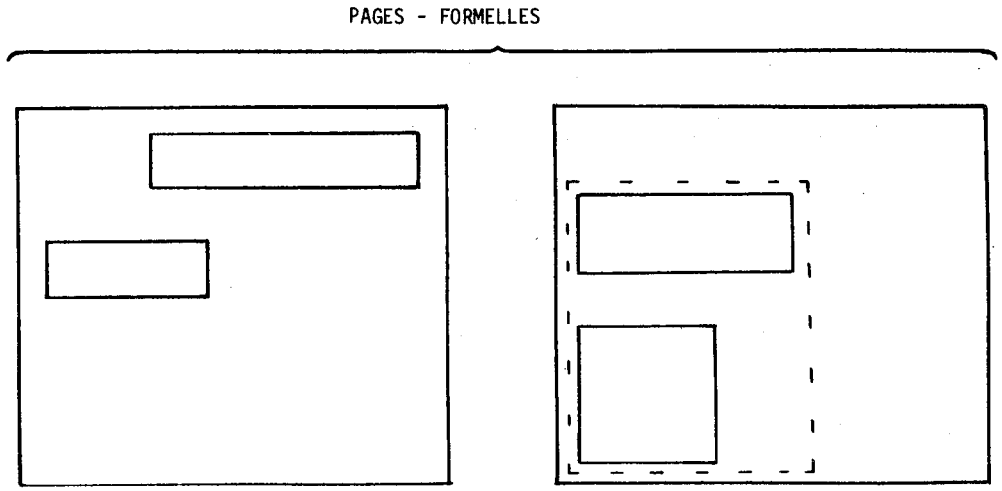


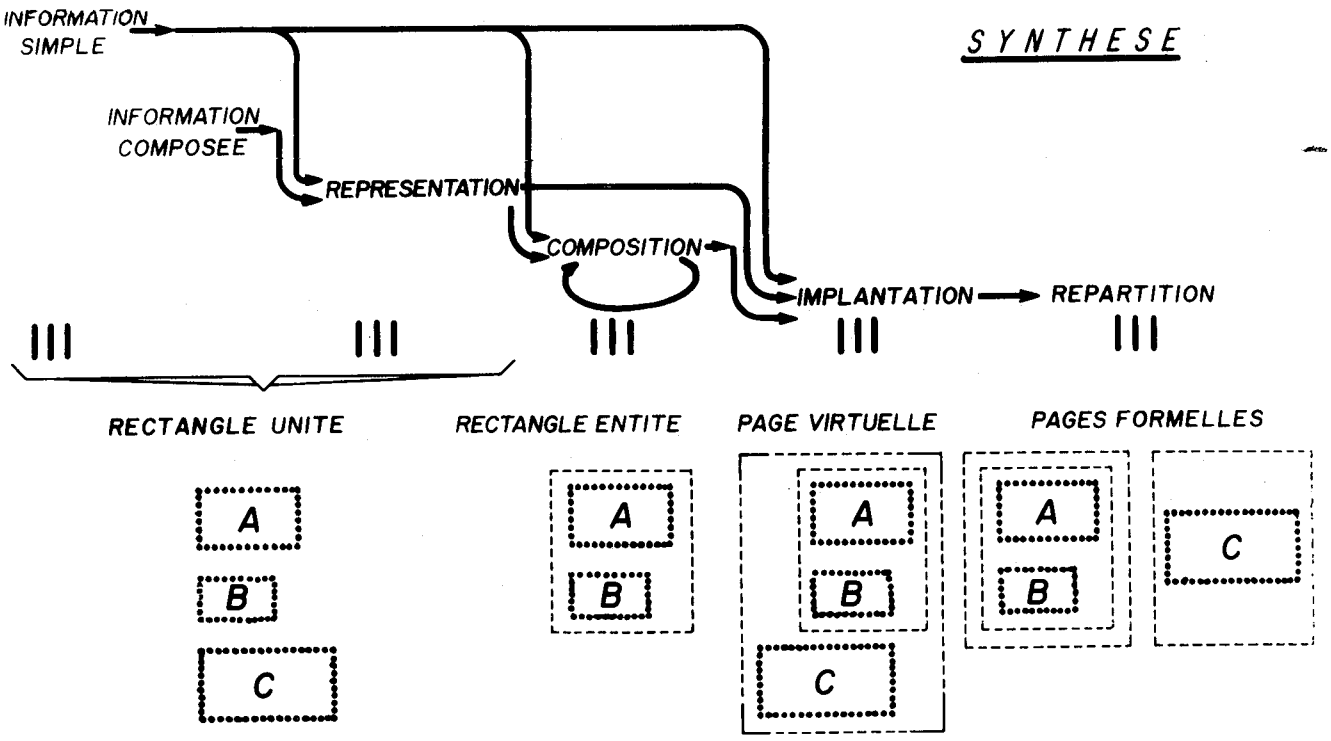
Figure 1.2.17.

L'intérêt de la "Répartition" réside dans le fait qu'elle permet au programmeur d'assujettir certains des rectangles constitutifs d'une Page-Virtuelle à figurer sur une même Page-Formelle.Ce qui entraîne,lors de l'Edition, la coexistence des informations correspondantes sur une même Page-Physique. De telles décisions sont exprimées sous la forme de liens logiques entre les informations concernées.

REMARQUE-4

L'élaboration de Pages-Formelles à partir d'un ensemble structuré de Rectangles-Unités et de Rectangles-Entités résulte:

- *D'opérations de composition; lesquelles, conformément à des lois de compositions internes matérialisent, dans l'esprit de notre étude, les liens impliqués par la phase Répartition.*



LEGENDE : ⟶ : *Soumise à des directives ;* ||| : *S'identifie à ;*

Figure 1.2.18.

2.2. LA CONCEPTION DE L.A.M.P.E. EN TANT QUE LANGAGE EVOLUE

Principale difficulté: Assurer la description des diverses requêtes imposées par une Edition, tout en respectant deux contraintes essentielles :

-L.A.M.P.E. doit être un langage évolué,
-L.A.M.P.E. doit admettre pour traits dominants ceux du langage évolué dont il est l'extension.

2.2.1. Les traits dominants

2.2.1.1. L.A.M.P.E. a une conception conforme à celle de nombreux langages de programmation.

L.A.M.P.E. se présente sous un double aspect :

- Aspect "description des données", lequel implique l'existence de déclarations.

- Aspect "traitement des données" qui implique l'élaboration d'algorithmes conduisant à l'Edition d'informations préalablement déclarées.
  Ce qui motive l'existence d'instructions et de fonctions incorporées.

REMARQUE

Les aspects "description de données" et "traitement de données" matérialisent les aspects "description" et "traitement" mentionnés précédemment (Cf.§2.1.2.2. et §2.1.2.3.).

Ainsi:

- L'aspect description du langage caractérise la description de données à l'aide de déclarations. .

- L'aspect traitement du langage caractérise l'élaboration d'un traitement à l'aide d'instructions dont la mise en oeuvre peut parfois être conditionnée par les résultats émanant d'une fonction incorporée ou d'un traitement marginal.

2.2.1.2. L.A.M.P.E. est une extension d'un langage évolué

La conception d'un langage comme extension d'un langage évolué existant semble actuellement avoir la faveur des spécialistes en "logitiel". Il n'en est pour preuve que le nombre croîssant des langages spécialisés qui adoptent cette solution.

Nous en citerons trois, entre autres, qui, pour avoir adopté PL/1 comme support, ont retenu notre attention :

FORMAC [LA-2-1], P.D.E.L.(A Language for Partial Differential Equations)[CK-2-1] et Compile Time Facilities de PL/1 [SE-3-1]

Plusieurs avantages peuvent être retirés d'une telle orientation, et dans notre cas, nous en distinguerons notamment deux, relatifs à : la définition et à l'utilisation du langage.

a/ Du point de vue définition

Le caractère spécialisé de L.A.M.P.E. est accentué. Corrélativement la syntaxe du langage s'en trouve simplifiée, puisqu'aucune des déclarations instructions ou fonctions incorporées ne fait double emploi avec celles du langage hôte.

En effet,

- en ce qui concerne les données,
  * il incombe aux déclarations PL/1 d'assurer la description et l'élaboration des informations de base.
  * Il incombe aux déclarations L.A.M.P.E. d'assurer la description et l'élaboration des informations éditables.

- En ce qui concerne le traitement, ou plus précisément les algorithmes de "Mise-en-Pages" et "d'Edition",
  * Il incombe aux instructions PL/1 de décrire les algorithmes
  * Il incombe aux instructions L.A.M.P.E. de commander la mise en oeuvre des processus de "Mise-en-Pages" et "d'Edition".

En conséquence, L.A.M.P.E. et PL/1 s'imbriquent à un point tel qu'il n'y a pas, à proprement parler, d'utilisateurs de L.AM.P.E. mais des utilisateurs du langage "PL/1 - L.A.M.P.E.".

C'est cette considération qui nous a porté , par souci d'homogénéité:

- à calquer l'organisation globale des déclarations, instructions et appels de fonctions incorporées L.A.M.P.E. sur celle de leurs homologues PL/1. On utilise, à cet effet, la syntaxe PL/1.

  DECLARATION →MOT-CLE CORPS-DE-DECLARATION ;
  INSTRUCTION →[ETIQUETTE:] MOT-CLE CORPS-D'INSTRUCTION ;
  INDICATEUR-DE-FONCTION-INCORPOREE→ MOT-CLE [(PARAMETRES)];

- à ne pas distinguer, comme cela se pratique dans d'autres extensions de langage (P.D.E.L., Compile Time Facilities de PL/1), les mots-clés par un symbole spécial (% par exemple).

- à adopter, pour les déclarations ainsi que pour les instructions, le même indicatif (mot-clé initial) que celui adopté en PL/1 (DECLARE ou DCL et PUT).

b/ Du point de vue utilisation

Tout programmeur, déjà familiarisé avec l'emploi de PL/1, tirera partie plus rapidement et plus efficacement de ce nouveau langage.

Dans tout programme, déclarations, instructions et appel de fonctions incorporées L.A.M.P.E. figurent au même titre que leurs homologues PL/1.

Elles sont, de ce fait, astreintes, quant à leur emploi, aux règles syntaxiques régissant l'écriture d'un programme PL/1. (Portée des déclarations et structure de blocs, communication des paramètres effectifs, rédaction des instructions conditionnelles).

La rédaction d'un texte L.A.M.P.E., de même que celle d'un texte PL/1 ne respecte pas une forme rigide.

2.2.1.3. L.A.M.P.E. est un langage homogène

Les composants de ce langage sont répartis, en fonction du rôle qui leur est imparti, en différentes classes (indicateurs d'instruction, indicateurs de commande, modes, attributs, compléments d'information) - (Voir annexe 6).

Certaines de ces classes sont familières puisqu'elles existent

dans la plupart des langages de programmation. Par contre, certaines autres, bien que connues, jouent en L.A.M.P.E. un rôle prépondérant.

Il en est ainsi des attributs et des compléments d'information. (Cf.2ème partie : Manuel de Références).

2.2.2. Les "Objets": un sur-ensemble des variables PL/1

Les informations traitées par les instructions L.A.M.P.E., en raison de leur diversité, de leurs propriétés et de leur complexité revêtent une importance particulière.

Pour les différencier des données traitées par les instructions PL/1 nous les dénommons: "objets".

On désigne par "objet" une information ou un groupement structuré d'informations.

Les objets :

* *s'identifient aux variables PL/1 du fait que grâce à l'identification dont ils sont dotés ils participent comme elles à un traitement.*
* *se différencient des variables PL/1 du fait que les traitements auxquels ils sont soumis ont pour unique finalité l'édition.*
* *constituent un sur-ensemble des variables PL/1 du fait qu'ils prennent ces dernières pour base.*

2.2.2.1. L'emploi du terme "d'objet" dans le contexte du langage

Vue la signification qui est habituellement attachée au terme "d'objet", il nous a semblé que son emploi convenait pour désigner certaines étapes du raisonnement présidant à toute édition à l'aide de L.A.M.P.E.

a/ Dans son emploi courant,

le terme "d'objet" est évocateur de l'idée d'un produit fini, concret, qui, à lui seul ou solidairement avec d'autres de ses homologues, à un usage déterminé.

- Définir un "objet", telle est la tâche de son "concepteur" (Artisan, Ingénieur ou autres) à qui il appartient de résoudre les différents problèmes posés par sa réalisation, et aussi, d'exposer la solution préconisée sous une forme nette et précise (plans, rapports, etc...)

- Concrétiser un objet consiste en un travail nécessitant l'emploi d'outils ou de machines-outils appropriées.

- Utiliser un objet revient à lui donner la destination pour laquelle il a été conçu.

Ce qui nous amène à distinguer :

* les objets complémentaires qui ne peuvent remplir à eux seuls une fonction significative (écrou, roue, etc...)
* les objets autonomes (généralement composés) qui remplissent à eux seuls un rôle précis. Ce qui n'exclut pas leur participation possible à la constitution d'objets plus complexes.

b/ Dans notre cas,

le terme d'objet désigne une information ou un groupement structuré d'informations dont le rôle est de servir de base à la réalisation d'une édition.

- Définir un tel"objet" revient à décrire, sous forme d'une déclaration, sa composition.

- Concrétiser un objet revient à mettre en oeuvre, à l'aide des instructions L.A.M.P.E., les "Outils" adéquats dénommés plus précisément "procédures-outils" puisque ce sont des modules de traitement fournis par le système M.P.E.

- Utiliser un objet consiste à faire appel à lui

* Soit pour le faire participer à l'élaboration d'autres objets qui requièrent son concours,
* Soit pour exploiter, en tant que lecteur, les informations fournies par les imprimés qu'il engendre.

2.2.2.2. La classification des objets

Vu la diversité des liens imposés par la description d'une édition et par suite, vu la diversité des types d'objets existant, nous sommes amené à établir une classification précise que résume le tableau figurant à la page II-31'.

a/ Origine des objets

Nous désignons

- par objet propre, tout objet décrit à l'aide d'une déclaration L.A.M.P.E. et dont l'existence est, de ce fait, essentiellement motivée par des traitements afférents à l'Edition.

- par objet extérieur, tout objet décrit à l'aide d'une déclaration PL/1 et, par suite, externe au langage L.A.M.P.E. Ce qui explique l'emploi du qualificatif d'"extérieur".

b/ Nature des objets

Nous désignons

- par objet de base, tout objet servant de "base" à l'Edition envisagée et s'identifiant, de ce fait, aux informations fournies par le programme PL/1.

- par objet complémentaire, tout objet ayant pour unique fonction de fournir, lors de la représentation d'un objet de base, toutes les indications complémentaires requises telles que modèles de données, libellés précisant la signification des informations constitutives de l'objet de base, etc...

- par objet élaboré, tout objet s'identifiant à l'un des quatre rectangles fictifs définis par les concepts de base (Cf.§2.1.3) et résultant, de ce fait, de l'application :

  * à un objet de base, de directives de représentation (Cf § 2.1.2.2.a)
  * à un ou plusieurs objets élaborés, de directives de composition et de mise-en-pages (Cf §2.1.2.2.b et 2.1.2.2.c).

Ainsi :

Un Rectangle-Unité correspond à un objet REPRESENTATION

Un Rectangle-Entité correspond à un objet COMPOSITION

Un Rectangle-Page-Virtuelle correspond à un objet IMPLANTATION
Un Rectangle-Pages-Formelles correspond à un objet REPARTITION

c/ Propriétés des objets

Nous désignons :

- par objet éditable, tout objet dont la concrétisation se traduit par l'Edition directe des informations qui le composent,

- par objet non éditable, tout objet dont la concrétisation ne peut pas se traduire par l'édition directe des informations qui le composent,
  * soit, parce que, comme objet de base, il nécessite, de par sa nature, l'apport de précisions concernant sa représentation (Cf §2.1.2.2.)
  * soit, parce que, comme objet complémentaire, l'Edition de ses informations constitutives n'aurait aucun sens; son rôle étant de contribuer à la représentation d'un objet de base en fournissant, dans ce but, des renseignements complémentaires tels que Formats, Intitulés, etc...

d/ Classe des objets

Nous désignons :

- par objet composé, tout objet résultant de la réunion d'objets simples de même nature.

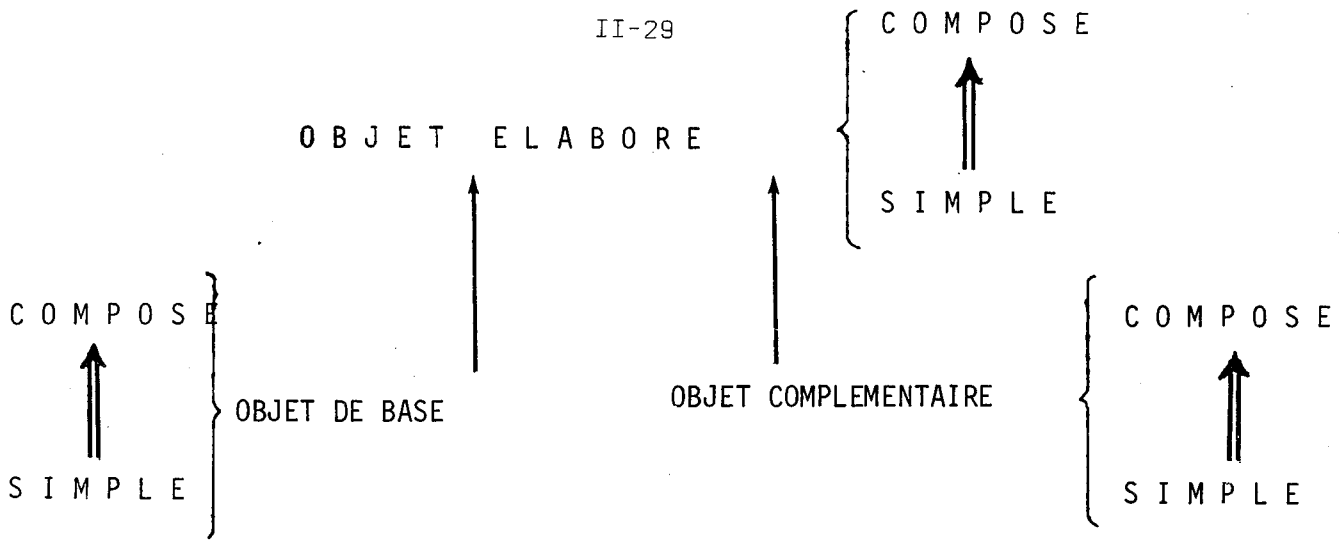
Ainsi :

- Tout objet élaboré composé est construit à partir d'objets élaborés simples
- Tout objet de base composé est construit à partir d'objets de base simples.
- Tout objet complémentaire composé est construit à partir d'objets complémentaires simples.

- par objet simple, le plus élémentaire parmi l'ensemble des objets répondant à une même propriété.

REMARQUE :

En fonction de leurs propriétés et de la compatibilité des liens qu'ils établissent entre leurs composants, la participation possible des objets à la constitution d'un autre est résumé dans le schéma suivant :



N.B. ⇑ signifie : "peut entrer dans la composition de".

↑ signifie : "participe nécessairement à la constitution de".

e/ Dénomination des objets

Nous désignons :

- par objet "Répartition", un objet éditable décrivant, conformément au concept-4 (Cf §2.1.3.3.) la configuration des "Pages-Formelles".

  Pour tout renseignement,

  * sur le rôle de l'objet : Voir §2.1.1.1.c et §2.1.2.2.c
  * sur les propriétés : Voir Manuel d'utilisation, §2.4.2.3.

- par objet "Implantation", un objet éditable décrivant, conformément au concept-3 (Cf. §2.1.3.3.) la configuration d'une "Page-Virtuelle".

  Pour tout renseignement,

  * sur le rôle de l'objet : Voir §2.1.1.1.c. et §2.1.2.2.c.
  * sur les propriétés : Voir Manuel d'utilisation, §2.4.2.2.

- par objet "Composition", un objet éditable décrivant, conformément au concept-2 (Cf. 2.1.3.2.), la configuration d'un "Rectangle-Entité".

  Pour tout renseignement,

  * sur le rôle de l'objet : Voir §2.1.1.1.b. et §2.1.2.2.b.
  * sur les propriétés : Voir manuel d'utilisation, §2.4.2.1.

- par objet "Représentation", un objet éditable décrivant, conformément au concept-1 (Cf.§2.1.3.1.), la configuration d'un "Rectangle-Unité".

  Pour tout renseignement,
  * sur le rôle de l'objet:Voir §2.1.1.1.a et §2.1.2.2.a
  * sur les propriétés:Voir Manuel d'utilisation, §2.3

- par objet "Modèle", un objet non éditable regroupant l'ensemble des indications complémentaires, nécessaires à la représentation d'un objet de base, et telles que :
  * des libellés permettant de préciser la signification des informations constitutives d'un objet de base.
  * des "Modèles de données" destinés, de même qu'en PL/1, à fournir des indications sur l'aspect que doit revêtir chaque information constitutive de l'objet de base.
  * des"facteurs itératifs" permettant de rendre la rédaction plus élégante en évitant la répétition d'indications identiques.

  Pour tout renseignement,
  * sur les propriétés de l'objet:Voir Manuel d'utilisation,§2.2

- par objet "Fichier", un objet non éditable regroupant, après qu'elles aient été sélectionnées, les données destinées à "faire corps" lors de leur Editio
  Un tel objet s'identifie à un bloc d'informations structuré et borné au même titre que les objets de base composés fournis par PL/1.

  Pour tout renseignement,
  * sur les propriétés de l'objet:Voir Manuel d'utilisation §2.1.

2.2.2.3. La description des objets à l'aide du langage

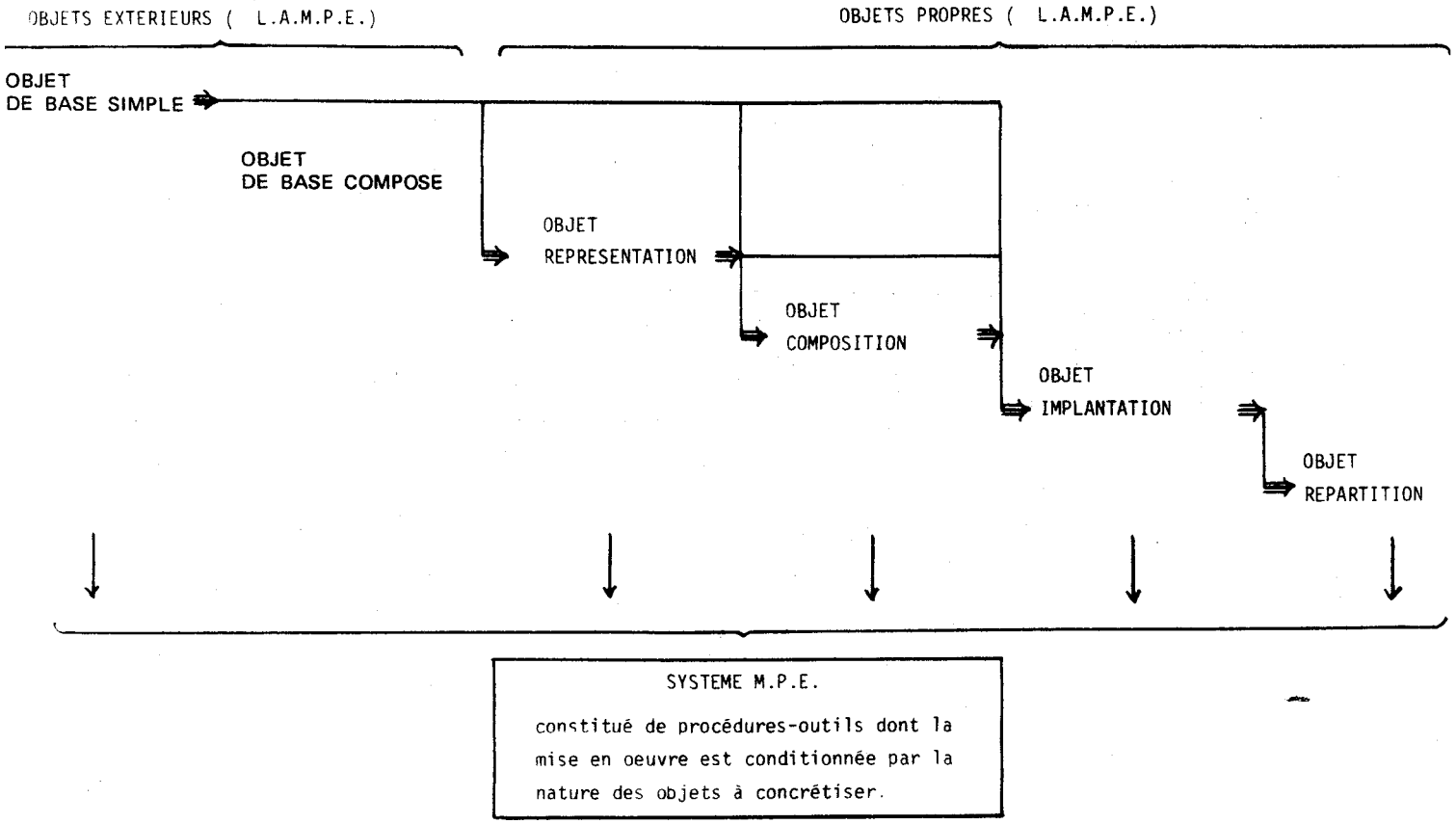
Les objets,qu'ils soient extérieurs ou propre (à L.A.M.P.E.), sont décrits à l'aide de déclarations.

- En ce qui concerne les déclarations d'objets extérieurs, nous renvoyons le lecteur aux brochures PL/1 [IB-3-2].

- En ce qui concerne les déclarations d'objets propres à L.A.M.P.E., nous renvoyons le lecteur, pour une étude plus détaillée, au manuel de référence (deuxième partie).

2.2.2.4. Les conditions d'utilisation des objets

Si les différents types d'objets existant, de même que l'indication de leurs propriétés, ont pû être rassemblés au sein d'une même classification, les conditions d'utilisation de ces objets peuvent se résumer en un schéma récapitulatif déduit de celui établi au §.2.1.2.4.



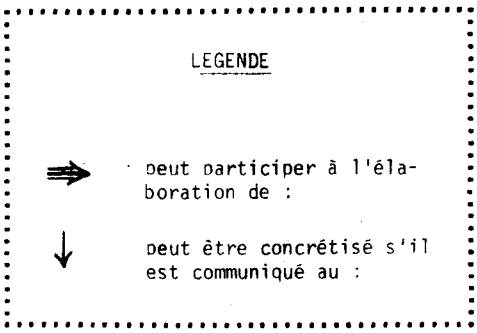


Figure 1.2.19.

CLASSIFICATION DES OBJETS

LES CRITERES DE CLASSIFICATION					DECLARATIONS				Hiérarchisation dans la puissance du langage	
Origine	Propriété	Nature	Classe	Dénomination	Mode	Type	Attribut	Complément d'Information	Degrés	Niveaux
Propre	Editable	Elaboré	Composé	Répartition	Proposition	Restrictive	Restrictif	Identificateur de l'Implantation	degré 4	
				Implantation		Libre	Libre		degré 3	
				Composition		Fixe	Fixe		degré 2	Niveau 2
			Simple	Représentation	Représentation	Arbitraire	Arbitraire	Identificateur d'un objet de base	degré 1	
						Conventionnelle	Linéaire / Tabulé / Structuré			
	Non éditable	de base	Composé	Fichier	Fichier			Identificateur d'une fonction		Niveau 1
		complémentaire	Composé	Modèle	Modèle					
Extérieur	Non éditable	de base	Composé	Tableau	Tableau					
				Structure	Structure					
			Simple	Scalaire	Scalaire					Niveau 0
				Chaîne de caractères	Chaîne de caractères					
				Chaîne de bits	Chaîne de bits					
"	"	Complémentaire	Simple	Modèle de données	Modèle de données					

2.2.2.5. La corrélation entre la nature des objets et la puissance du langage

La gradation dans la puissance du langage est matérialisée par l'emploi des notions de "niveau" et de "degré" (voir tableau p.II.31). C'est ainsi que nous pouvons mettre en évidence les possibilités offertes par L.A.M.P.E.:

* de décrire des objets de complexité croissante en raison de la nature des liens unissant leurs composants,
* d'exprimer, de ce fait, des directives d'autant plus délicates,
* de mettre en oeuvre des "procédures-outils" de puissance appropriée [LE-4-2] .

Plus précisément, les objets traités par le système M.P.E. pouvant être fournis indifféremment par PL/1 et par L.A.M.P.E., c'est de la hiérarchie dans la puissance du langage "PL/1-L.A.M.P.E." dont il nous faut parler.

a/ Niveau 0

Qu'elles appartiennent à PL/1 ou à L.AM.P.E., instructions et fonctions assurent le traitement des informations qui leur sont soumises, mais, en aucune façon, n'entraînent l'élaboration d'informations plus complexes. Il semble, donc, naturel de leur impartir le plus bas niveau du langage.
Il en est de même des déclarations PL/1, puisqu'elles engendrent des objets simplement destinés à subir un traitement, ou à servir de base à L.A.M.P.E. pour la description d'objets plus complexes.

b/ Niveau 1

"Fichier" et "Modèle" sont des objets propres à L.A.M.P.E. directement et essentiellement conçus à partir d'objets extérieurs.
Ils sont donc de complexité plus grande, et leur déclaration requiert, de ce fait, des possibilités de description plus évoluées: celles fournies par le niveau-1 du langage.

c/ Niveau 2

"Représentation","Composition", "Implantation" et "Répartition" sont des objets propres à L.AM.P.E. dont l'existence est motivée par l'aspect et les regroupements logiques des objets qui sont à la "base" de leur conception: les "objets de base".

Ce sont des objets évidemment plus complexes que ceux auxquels ils s'adressent. Leur déclaration implique les possibilités de description fournies par le niveau-2 du langage.

Cependant, tous les objets concernés par le niveau-2, ne traduisent pas des configurations de même complexité. C'est la raison pour laquelle, une divisior plus fine a été envisagée au sein de ce même niveau, sous la forme d'une gradation.

RESUME :

Niveau-2 : Objets composés LA.M.P.E. { Degré-4 : Répartition ; Degré-3 : Implantation ; Degré-2 : Composition ; Degré-1 : Représentation }

Niveau-1 : Objets simples L.A.M.P.E.

Niveau-0 { - Objets PL/1 ; - Instructions ; - Fonctions } PL/1-L.A.M.P.E.

2.3. LA CONCEPTION DE L.A.M.P.E. EN TANT QUE LANGAGE DE COMMANDE

Principale difficulté

L'aspect de langage évolué conféré à L.A.M.P.E. ne doit pas dissimuler sa raison d'être; en l'occurrence, permettre au programmeur d'utiliser efficacement les ressources du "système M.P.E." de façon à ce que l'édition décrite, conformément à l'idée maitresse, puisse être réalisée.

La conception de L.A.M.P.E. en tant que langage de commande implique une corrélation étroite entre ses possibilités de description et les possibilités de traitement imparties au système M.P.E.

2.3.1. La correspondance entre la nomenclature de L.A.M.P.E. et les fonctions du système M.P.E.

Les fonctions essentielles du système M.P.E. réside en deux traitements:

La constitution,

Traitement au cours duquel est paramétrée en mémoire, la

représentation des composants d'une édition et les liens établis entre eux. Ce qui se ramène, en vertu des concepts énoncés, à paramétrer la configuration des diverses surfaces rectangulaires fictives qui peuvent être décrites en L.A.M.P.E. La "Constitution" est régie par l'aspect description du langage puisqu'elle résulte de l'interprétation des déclarations d'objets éditables dont le type symbolise une phase dans la description d'une édition.

La concrétisation

Traitement qui engendre effectivement l'édition et qui se scinde en deux opérations :

* La FORMATION des pages physiques. Ce qui se ramène à paramétrer la configuration de chaque page.
* L'IMPRESSION des pages physiques. Ce qui matérialise la paramétrisation précédemment accomplie.

La "Concrétisation" est régie par l'aspect "traitement" du langage puisqu'elle est tributaire des instructions et également de la mise en oeuvre des fonctions.

2.3.2. La correspondance entre la nomenclature de L.A.M.P.E. et l'architecture du système M.P.E.

L'architecture du système M.P.E. est à l'origine de la nature des indications apportées respectivement par les déclarations et les instructions L.A.M.P.E.

- La CONSTITUTION requiert la mise en oeuvre de procédures-outils spécialisées; ce qui motive la composition du corps des déclarations d'objets éditables à l'aide d'unités syntaxiques du langage dénommées "DIRECTIVES" (Cf Manuel de Référence §1.4.4.)

- La CONCRETISATION requiert la mise en oeuvre d'ensembles de procédures-outils (ou processus) . Ces regroupements ont été nécessités en raison, d'une part, de la modularité volontairement conférée au système M.P.E. et, d'autre part, de la complexité des tâches conduisant à la "Formation" des pages physiques. Les processus ont, en effet, la lourde charge d'effectuer la transition

entre la description logique et l'élaboration effective des pages.

Ces processus entrent dans la composition des deux "dispositifs" impliqués par l'opération de concrétisation et assurant, respectivement, la Formation et l'Impression des pages physiques. La mise en oeuvre de ces dispositifs nécessite l'apport d'indications pratiques (dimensions des imprimés, numérotation des pages, itération, etc...); ce qui motive l'adjonction d'informations complémentaires aux mots-clés représentatifs des instructions et fonctions L.A.M.P.E. (Cf Manuel de Référence §1.6 et 1.7).

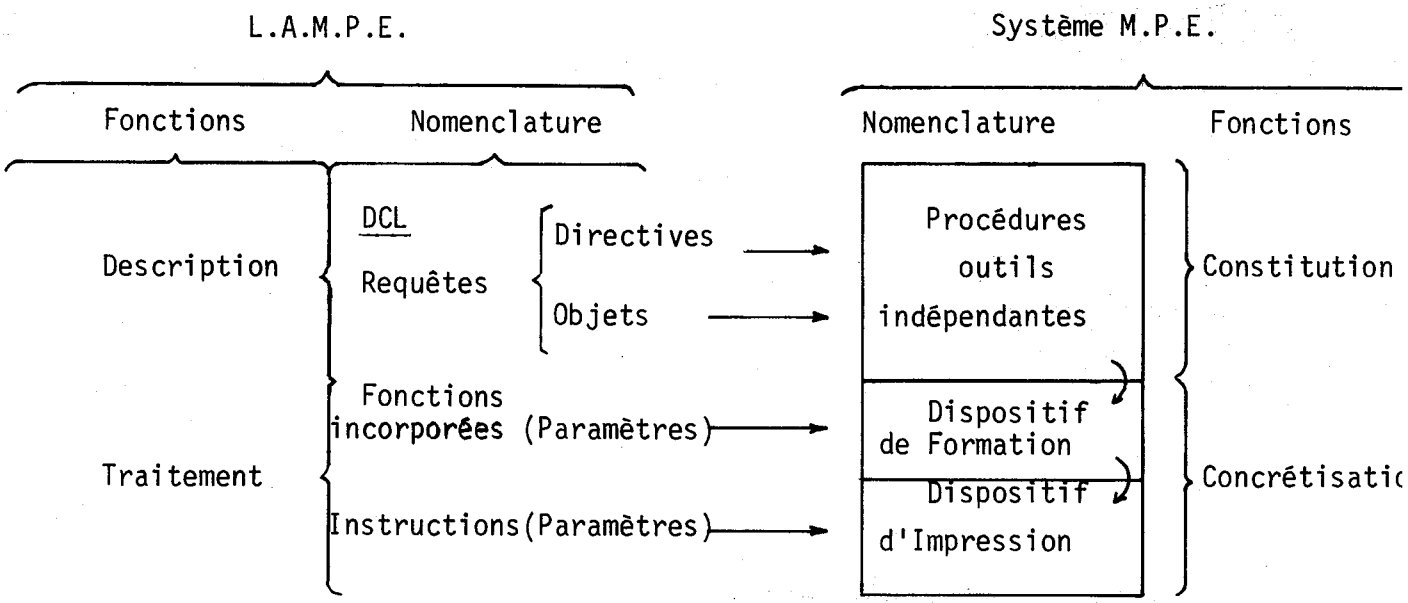


Figure 1.2.20.

2.3.3. La correspondance entre la nomenclature de L.A.M.P.E. et la technique de "formation" des pages physiques utilisée par le système M.P.E.

Une Page-Formelle peut s'identifier,en vertu du concept-4,à une Page Virtuelle. Tel est le cas si les rectangles constitutifs de la Page-Virtuelle confèrent, à cette dernière, des dimensions inférieures ou, au plus, égales à celles dont sont théoriquement dotées les Pages-Formelles.

Une Page-Virtuelle peut s'identifier, en vertu du concept-3, à un Rectangle-entité. Tel est le cas si la Page-Virtuelle est construite à partir d'un seul Rectangle-entité.

Un Rectangle-Entité peut s'identifier, en vertu du concept-2, à un Rectangle-Unité. Tel est le cas si le Rectangle-entité ne comporte qu'un seul Rectangle-unité.

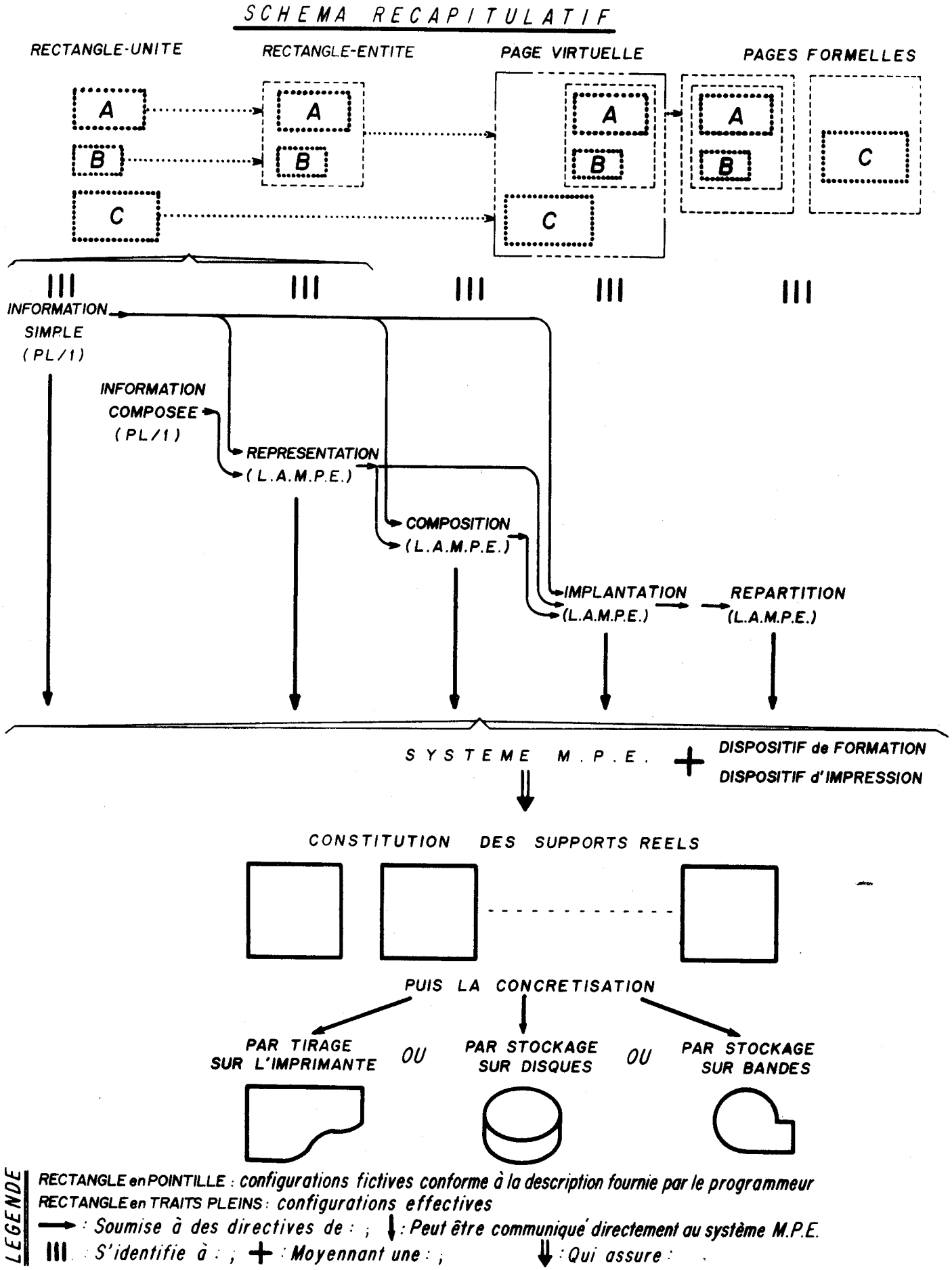

SCHEMA RECAPITULATIF
RECTANGLE-UNITE
RECTANGLE-ENTITE
PAGE VIRTUELLE
PAGES FORMELLES
A
B
C
INFORMATION SIMPLE (PL/1)
INFORMATION COMPOSEE (PL/1)
REPRESENTATION (L.A.M.P.E.)
COMPOSITION (L.A.M.P.E.)
IMPLANTATION (L.A.M.P.E.)
REPARTITION (L.A.M.P.E.)
SYSTEME M.P.E.
DISPOSITIF de FORMATION
DISPOSITIF d'IMPRESSION
CONSTITUTION DES SUPPORTS REELS
PUIS LA CONCRETISATION
PAR TIRAGE SUR L'IMPRIMANTE
OU
PAR STOCKAGE SUR DISQUES
OU
PAR STOCKAGE SUR BANDES
LEGENDE
RECTANGLE en POINTILLE : configurations fictives conforme à la description fournie par le programmeur
RECTANGLE en TRAITS PLEINS : configurations effectives
: Soumise à des directives de : ; : Peut être communiqué directement au système M.P.E.
: S'identifie à : ; + : Moyennant une : ; : Qui assure :

Figure 1.2.21.

En conséquence,

la formation des pages physiques résulte du passage d'un ensemble structuré de rectangles implantés au sein de l'une des surfaces rectangulaires fictives définies par les concepts de base, à un ensemble ordonné de surfaces rectangulaires de dimensions imposées : les "Pages-Physiques".

Les diverses surfaces rectangulaires fictives précédemment décrites (Cf §2.1.3.) correspondent, chacune, à un type d'objet éditable. Il s'ensuit que tout objet éditable, quel que soit son type, peut être concrétisé. Ce qui est conforme au fait que le type d'un objet éditable caractérise une phase de la description d'une édition en précisant les relations logiques établies entre les composants désignés. Il s'agit donc d'un élément autonome qui peut indistinctement soit être mis, lui-même, en relation avec d'autres de ses homologues, soit être concrétisé.

Cette constatation explique la raison pour laquelle l'objet désigné dans une instruction L.A.M.P.E. ne peut être qu'un objet éditable

2.3.4. La méthode employée pour la Formation des Pages-Physiques

Deux méthodes susceptibles d'assurer efficacement l'élaboration des pages-physiques ont retenu notre attention.

2.3.4.1. La méthode de "la fenêtre".

Cette méthode a été utilisée pour assurer la représentation de formules mathématiques, sous leur aspect conventionnel [SI-2-1]. Le principe en est le suivant:

Les formules mathématiques, qui peuvent être, sous leur forme traditionnelle, extrêmement volumineuses, sont élaborées sur un support virtuel rectangulaire, de dimensions non bornées.

L'écran de visualisation, choisi comme support de l'Edition, est considéré comme une fenêtre pouvant, à la guise du programmeur, se déplacer sur le support virtuel.

Un dialogue s'instaure entre l'utilisateur et l'ordinateur fonctionnant, pour les besoins de la cause, en mode conversationnel. Ainsi, ses directives étant traitées immédiatement, l'utilisateur peut-il visualiser la portion de formule qui l'intéresse dans la mesure où celle-ci ne pourrait être contenue en totalité sur l'écran cathodique.

Mais, dans notre cas, cette solution ne pouvait être admise.

En effet, les résultats dont L.AM.P.E. facilite la présentation, ne sont pas destinés à une consultation momentanée.

En conséquence :

- L'écran de visualisation, dont l'utilisation fait partie des extensions prévisibles, ne peut jouer le rôle de support de l'édition. Son aide ne se révèlerait particulièrement efficace qu'en tant qu'écran-témoin permettant au programmeur, d'avoir un aperçu de la disposition finale au sein de chaque page, et, de pouvoir ainsi communiquer de nouvelles directives, le cas échéant.

- Seule la feuille de papier peut constituer, en l'état actuel des techniques, un support valable.

Certes, la page de "listing" pouvait être assimilée, au même titre que l'écran de visualisation, à une "fenêtre". Mais, les informations traitées, étant donné leur nature, auraient été tronquées arbitrairement, dans le cas où leurs dimensions ne leur auraient pas permis de tenir sur une même page. De telle sorte que l'interprétation des pages générées n'aurait plus été possible que par leur juxtaposition, dans le bon ordre, afin de reconstituer intégralement le contenu du support virtuel.

2.3.4.2. La méthode adoptée

En fonction d'un nombre réduit de principes,il est possible pour les processus appropriés du système M.P.E. d'assurer le transfert et l'adaptation des informations à éditer des surfaces rectangulaires fictives décrites par le programmeur à une ou plusieurs pages-physiques.

EXEMPLE :

Considérons la facture (figure 1.2.8.) comme l'image d'une Page Virtuelle.

Les Pages-Physiques étant supposées de dimensions réduites, la mise en oeuvre des processus nous permet d'obtenir les imprimés suivants:

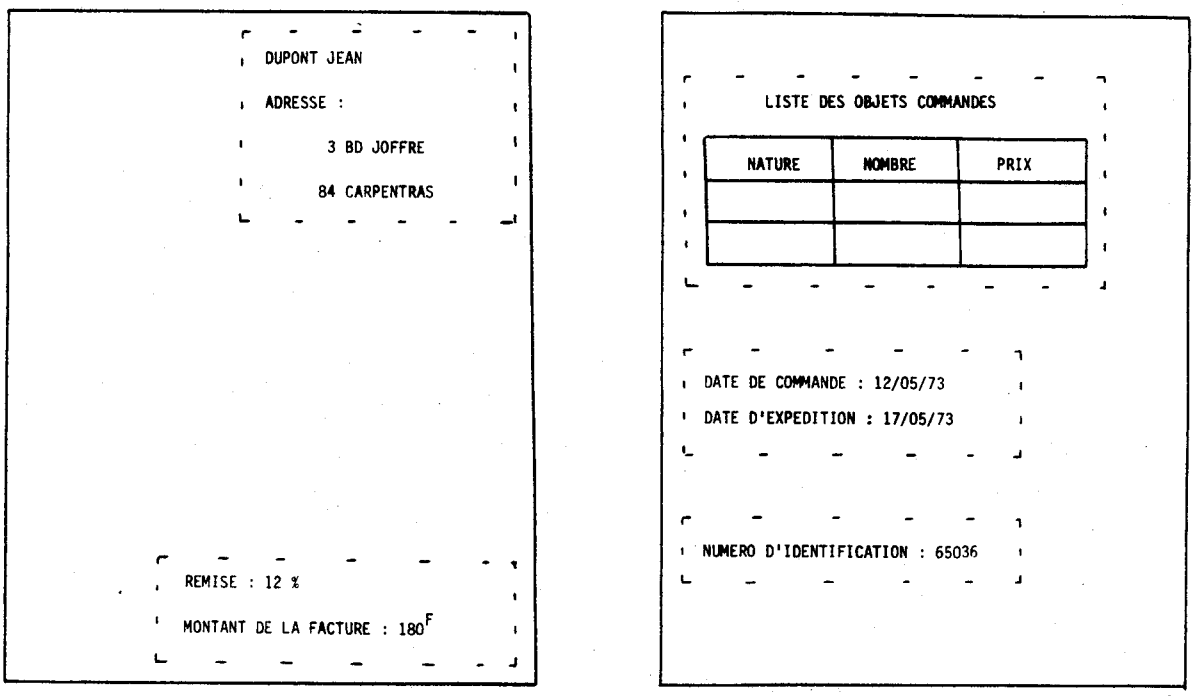


Figure 1.2.22.

Si, grâce à l'opération de Répartition, nous décidions d'imposer la coexistence sur une même Page-Physique:

- de l'entité constituée par la liste des objets commandés,
- de l'entité constituée par le montant et la remise,

les imprimés auraient alors l'aspect suivant :

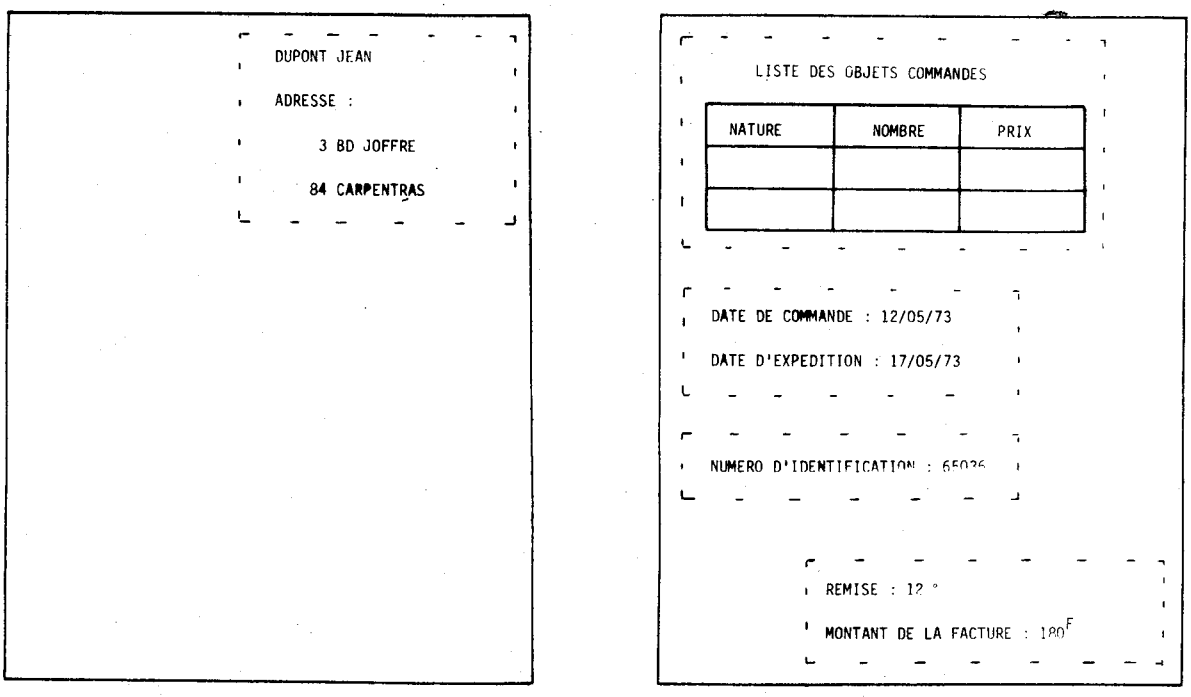


Figure 1.2.23.

Il est à noter que les traitements systématiquement assumés par le système M.P.E. peuvent, en fonction de la nature des données et en fonction de la nature des relations établies entre elles se révéler impossibles ou se traduire par des résultats non satisfaisants.

Dans cette éventualité, la possibilité est offerte au programmeur de tester, à l'aide des instructions conditionnelles PL/1 et des fonctions appropriées L.A.M.P.E. (Cf Manuel de Référence §1.7.) l'opportunité de la configuration des Pages-Physiques. L'impression ne sera donc déclenchée que dans la mesure où le programmeur donne son accord, en connaissance de cause.

Cet aspect algorithmique du langage permet de suppléer, dans la version actuelle, la non-utilisation d'un écran de visualisation graphique.

2.3.5. Les principes régissant la "Formation" des Pages-Physiques.

La formation des Pages-Physiques est confiée à trois processus :

- Le processus d'EVALUATION: il détermine la surface demeurant disponible sur la Page-Physique courante.

- Le processus d'ADAPTATION: il décide de la disposition, après des découpes éventuelles, des informations figurant sur une surface rectangulaire fictive.

- Le processus de DECISION: il retient, en fonction des critères précis, la solution la plus favorable.

L'apport d'indications sur les processus relève de l'implémentation, et nous aurions pû envisager de ne renseigner le lecteur que dans la troisième partie de la thèse.

Néanmoins, avant même de décrire le langage, de même qu'il nous a semblé souhaitable de présenter et de justifier les divers concepts adoptés, de même il nous a semblé opportun d'énoncer, dès à présent, les principes régissant le fonctionnement du processus le plus complexes (le processus d'ADAPTATION).

2.3.5.1. Les principes respectés par le processus d'Adaptation

2.3.5.1.1. Enoncé des principes

Trois principes permettent au processus d'Adaptation de remédier

aux diverses situations qui peuvent se présenter.

N.B. Nous qualifions"d'indissociables" les éléments qui ne peuvent figurer partiellement sur deux pages physiques consécutives.

Principe 1

Les éléments (caractères alpha-numériques ou spéciaux) constitutifs d'un objet de base simple sont indissociables.

Soient les deux informations suivantes :

* "0.314159+01"
* "CHAINE DE CARACTERES"

Il est impossible de répartir, après troncature, chacune de ces deux informations sur deux pages distinctes sans nuire à leur interprétation.

Principe 2

Les éléments (objets de base simples) constitutifs d'un objet de base composé sont dissociables.

Il est couramment admis que des informations telles que des tableaux, des structures, des textes peuvent être décomposés en sous-ensembles cohérents, destinés à être implantés sur des pages successives sans que, pour autant, cette opération ne nuise à leur interprétation.

Principe 3

Les éléments (objets éditables simples) constitutifs d'un objet éditable composé sont dissociables ou non selon la signification attachée à de tels regroupements dans le cadre d'une Edition.

2.3.5.1.2. Corollaires

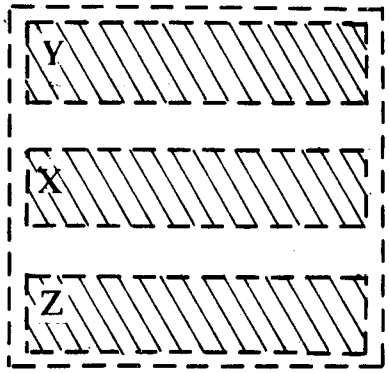
Considérons l'objet Composition correspondant au "Rectangle-Entité" schématisé par la figure 1.2.24.

En vertu du concept 2 (Cf.§2.1.3.2.), supposons que les rectangles X,Y,Z soient des "Rectangles-Unités".

En vertu du concept 1 (Cf.§2.1.3.1.) les rectangles X,Y,Z peuvent

s'identifier à des objets éditables simples construits indifféremment à partir d'une information de base simple ou composée.

Figure 1.2.24.



* Les informations, que chacun des rectangles X, Y, Z renferme, doivent figurer dans leur intégralité sur une même page physique. Ceci conformément à l'indissociabilité caractérisant leur groupement (principe 3), à condition qu'elles soient toutes des informations de base simples (principe 1).

  EXEMPLE :

  MONTANT DE LA FACTURE : 1500.00 FRANCS.
  REMISE CONSENTIE : 12 %.
  TOTAL A VERSER : 1320.00 FRANCS.

  Un tel regroupement perdrait, en effet, sa signification s'il était scindé en deux sous-ensembles répartis sur deux pages successives.

* Les informations que chacun des rectangles-unités X, Y, Z renferme, peuvent ne pas figurer dans leur intégralité sur une même page physique. Ce cas se produit,malgrè l'indissociabilité caractérisant un tel regroupement (principe-3) Si l'un des rectangles supporte la représentation d'une information de base composée (principe-2).

a/ Premier cas

Les informations de base simples se rapportent, de par leur signification, à l'ensemble de l'information de base composée sans distinction.

EXEMPLE :

Supposons que,

X représente un tableau de statistiques (information de base composée

Y " un commentaire précisant la signification de ce tableau,

Z " le nombre d'échantillons à partir duquel ce tableau a pu être obtenu.

Dans la mesure où l'Edition de l'ensemble des informations regroupées en une entité ne peut être réalisée sur une seule page du "listing", la méthode de répartition adoptée est la suivante :

- L'information de base composée est scindée en sous-ensembles cohérents compatibles avec les dimensions de pages.

- Chaque occurrence d'un sous-ensemble de l'information de base composée sur une page du "listing" est accompagnée de l'ensemble des informations qui s'y rapportent.

Page: 1

INSTITUT DE METEOROLOGIE

PRECIPITATIONS MOYENNES PAR REGION

ET PAR SAISON

REGION	HIVERS	PRINTEMPS
PARIS	112	076
RHONE-ALPES	120	108
LANGUEDOC	084	062
MEDITERRANEE	090	053
BRETAGNE	150	116
NORD	132	123

RESULTATS REPRESENTANT LA MOYENNE DE 50 RELEVES EFFECTUES DANS CHAQUE REGION

Page: 2

INSTITUT DE METEOROLOGIE

PRECIPITATIONS MOYENNES PAR REGION

ET PAR SAISON

REGION	ETE	AUTOMNE
PARIS	032	082
RHONE-ALPES	058	073
LANGUEDOC	021	056
MEDITERRANEE	024	058
BRETAGNE	068	098
NORD	072	102

RESULTATS REPRESENTANT LA MOYENNE DE 50 RELEVES EFFECTUES DANS CHAQUE REGION

Figure 1.2.25.

b/ Deuxième cas

Les informations de base simples, bien que concernant de par leur signification la totalité de l'information de base composée, se rapportent plus particulièrement les unes aux premiers, les autres, aux derniers renseignements fournis par l'information de base composée.

Supposons que,

X représente le texte d'un chapitre,

Y représente le titre du chapitre,

Z représente une citation venant à l'appui de l'idée développée dans le chapitre,

le titre du chapitre doit, par convention, être essentiellement mentionné en début du texte et la citation ne devra figurer qu'à la fin.

Dans la mesure où l'Edition de l'ensemble des informations, regroupées en une entité, ne peut pas être réalisée sur une seule page du "listing", la méthode de répartition est la suivante :

- L'information de base composée est scindée en sous-ensembles compatibles avec les dimensions des pages.

- L'occurrence, sur une page du "listing", du premier sous-ensemble de l'information de base composée s'accompagne de l'information qui s'y rapporte.

Il en est de même pour le dernier sous-ensemble.

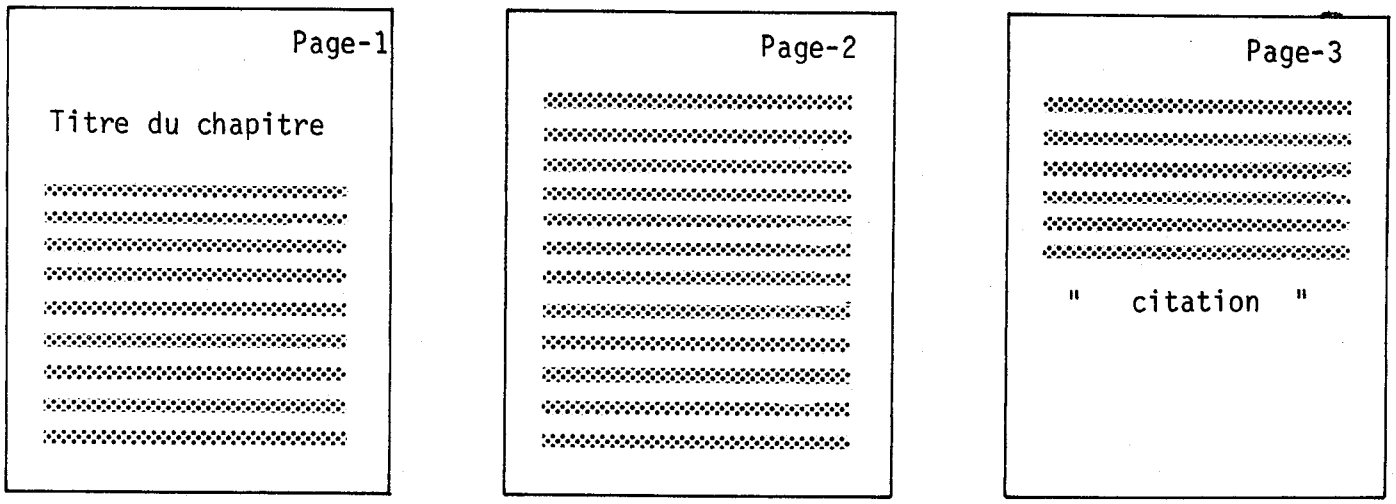


Figure 1.2.26.

2.3.5.2. Les principes respectés par le processus de Décision

Deux critères président à toute décision quant à l'opportunité de la solution à retenir :

* Le nombre de découpes à effectuer sur la représentation de tout objet de base composé.
* Le nombre de pages physiques nécessaires à l'édition envisagée

D'où :

Principe 4

De plusieurs solutions possibles nécessitant un nombre différent de découpes dans les représentations d'objets de base composés impliqués dans l'édition, le processus de décision retient celle se soldant par le nombre minimum de découpes.

Principe 5

De plusieurs solutions possibles nécessitant le même nombre de découpes dans les représentations d'objets de base composés impliqués dans l'édition, le processus de Décision retient celle qui se solde par le nombre minimum de pages imprimées.

Principe 6

De plusieurs solutions litigieuses (même nombre de découpes, même nombre de pages), le processus de Décision retient celle qui offre l'écart le plus faible entre les rapports (largeur/hauteur) des zones éditées sur chaque page-physique.

2.3.6. L'incidence des principes adoptés sur le dispositif de formation des pages-physiq

Pour cet exposé, bornons-nous à développer un exemple général. Considérons quatre rectangles, de dimensions finies, dénommés: I,J,K,L. Admettons qu'ils s'identifient indifféremment à des "Rectangles-Unités" et à des "Rectangles-entités" (Cf §2.1.3.).

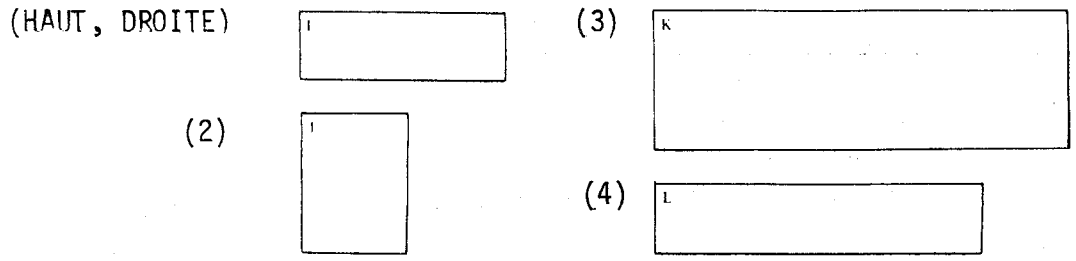


Figure 1.2.27.

N.B. Les indications figurant à gauche des rectangles symbolisent les directives que le programmeur a la possibilité d'assigner, dans le cadre de l'Edition envisagée, aux informations que chaque rectangle renferme (Cf §2.1.2.2.).

Compte tenu de ces directives, la disposition au sein d'une Page-Virtuelle se présente comme l'indique la figure 1.2.28.

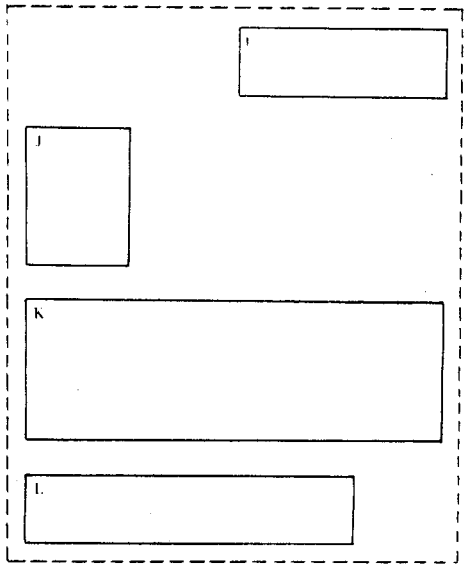


Figure 1.2.28.

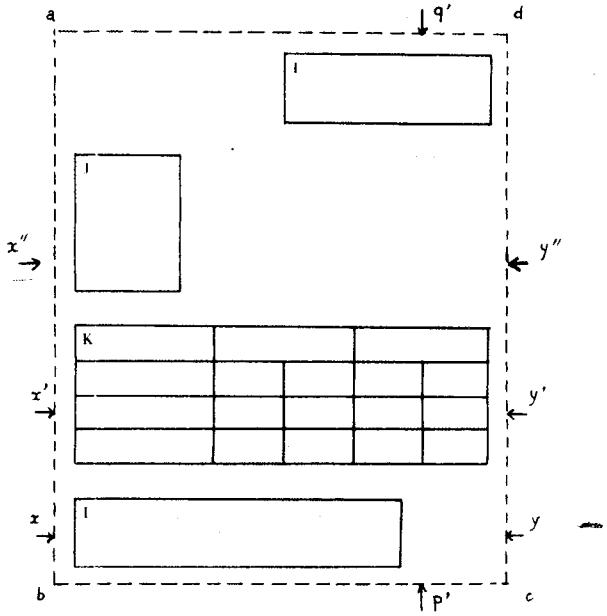


Figure 1.2.29.

Comment va être assumé le transfert de cet ensemble structuré de rectangles vers des supports de dimensions imposées (les Pages Physiques) ?

Le "Processus d'Adaptation", en fonction des dimensions des Pages Physiques, envisage les diverses solutions possibles compte tenu des principes précédemment énoncés.

Le "Processus de Décision" détermine , en vertu des principes admis, la solution paraissant être la plus favorable.

Simulons, sur des cas précis, le fonctionnement de ces processus.

2.3.6.1. La Page-Physique peut contenir de par ses dimensions l'ensemble des rectangles.

Dans le cas où les rectangles I,J,K,L peuvent trouver place sur une même Page-Physique, l'élaboration de l'imprimé ne pose aucune difficulté. Son aspect peut être identique ou non à celui de la Page-Virtuelle (voir figure 1.2.28). Ses dimensions pouvant être ou non proportionnelles à celles admises par la Page-Virtuelle une fois disposés les rectangles I,J,K,L (voir respectivement figure 1.2.30 et figure 1.2.31).

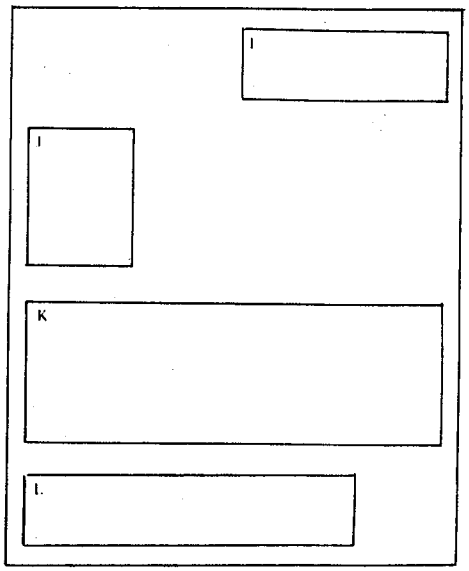


Figure 1.2.30.

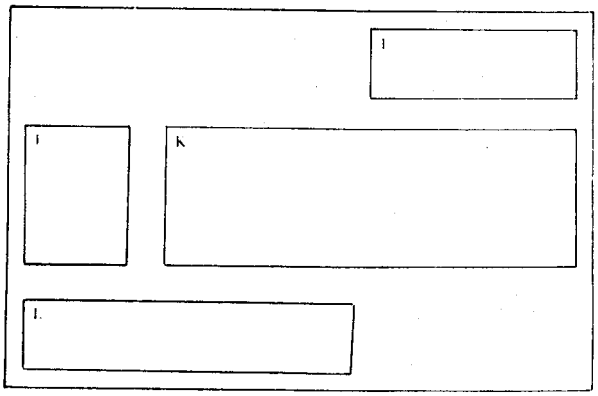


Figure 1.2.31.

2.3.6.2. La Page Physique ne peut contenir de par ses dimensions l'ensemble des rectangles.

Dans la mesure où l'ensemble des rectangles I,J,K,L ne peut coexister sur une même page, l'élaboration des imprimés est plus délicate. Pour justifier l'enchainement des opérations, précisons le type des informations constitutives des rectangles.

Supposons que :

I et L	= Rectangles-Unité	Représentatifs d'un objet de base simple
K	= Rectangle-Unité	Représentatif d'un objet de base composé
J	= Rectangle-Entité	

L'aspect de la page virtuelle est précisé par la figure 1.2.29. Admettons pour les besoins de l'exposé, afin de minimiser le nombre de solutions envisageables, que les Pages-Physiques aient des dimensions qui ne peuvent être supérieures à celles admises par la Page-Virtuelle une fois les rectangles I,J, K,L disposés.

2.3.6.2.1. Première éventualité

Le rectangle "L" ne peut être implanté sur la page où figurent les rectangles préalablement communiqués.

Il en est ainsi des Pages-Physiques assimilables, par exemple, au rectangle a x y d mentionné dans la figure 1.2.29.

Il y a interruption du traitement assumé par le processus d'Adaptation.

L'interruption du traitement assumé par le processus d'Adaptation est provoqué par un "Rectangle-Unité" renfermant la représentation d'un objet de base simple.

En vertu du principe -1 le rectangle concerné est rangé sur la page suivante (voir figure 1.2.32).

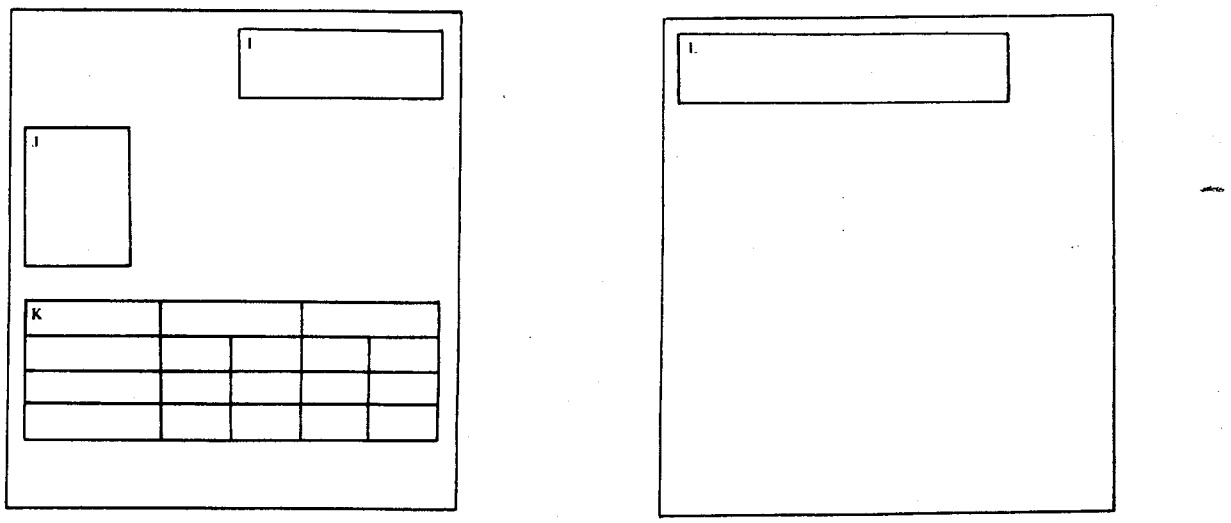


Figure 1.2.32.

2.3.6.2.2. Deuxième éventualité

Le rectangle K ne peut être implanté sur la page où figurent les rectangles préalablement communiqués.

Il en est ainsi des Pages-Physiques ayant, par exemple, pour dimensions celles des rectangles a x'y'd ou a b p'q' mentionnés sur la figure 1.2.29.

L'interruption du traitement assumé par le processus d'Adaptation est provoqué par un "Rectangle-Unité" renfermant la représentation d'un objet de base composé.

En vertu du principe-2, est alors assumée une découpe logique de l'objet de base composé en sous-ensembles autant que possible symétriques.

a/ Dans le cas où le support de l'Edition est constitué de pages telles que a x' y'd :

Première solution

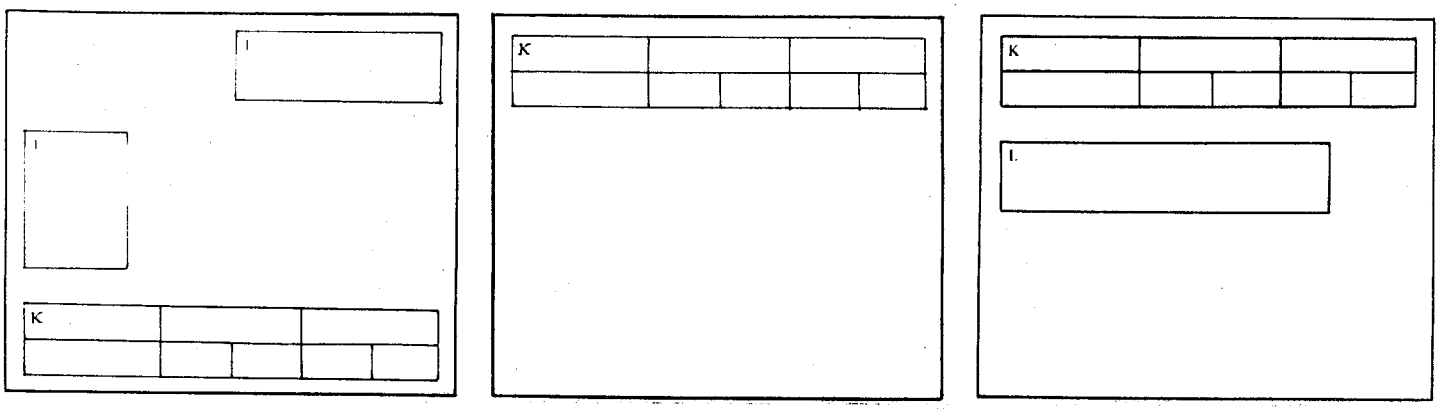


Figure 1.2.33.

Deuxième solution

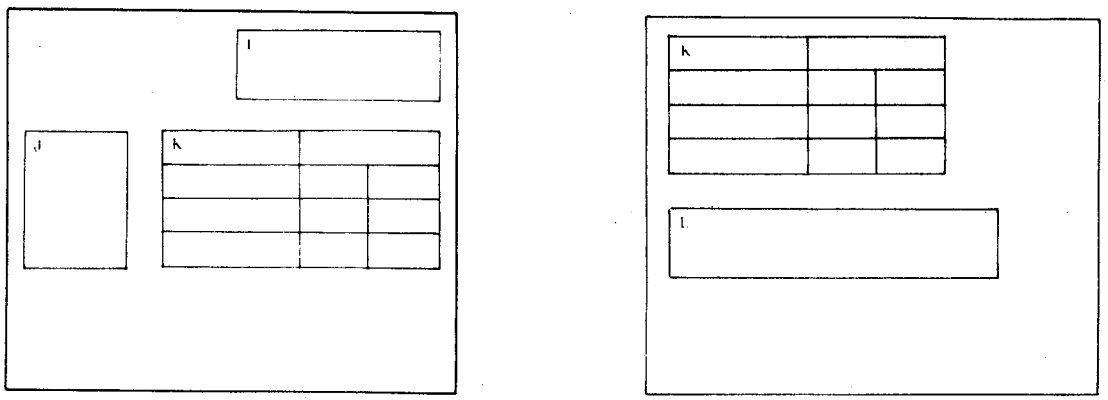


Figure 1.2.34.

Troisième solution

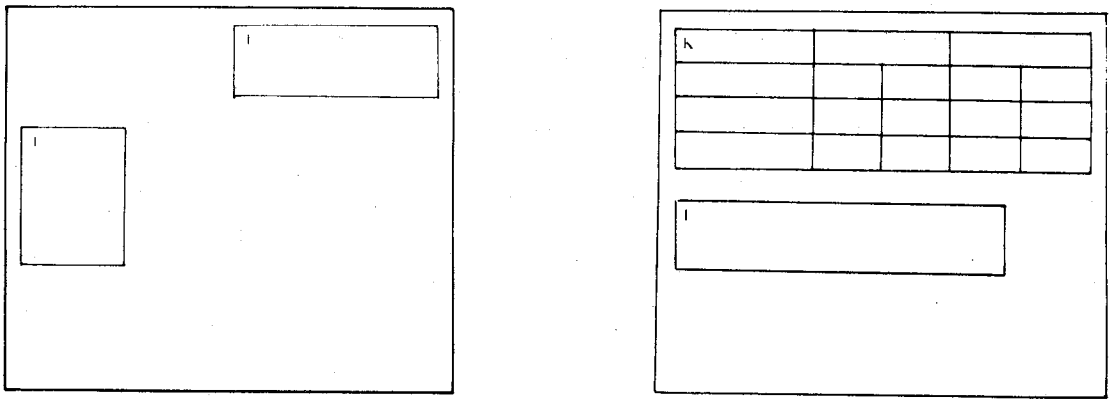


Figure 1.2.35.

En vertu du principe-4, le processus de Décision considère comme satisfaisante la troisième solution.

b/ Dans le cas où le support de l'édition est constitué de pages telles que a b p' q' :

Première solution

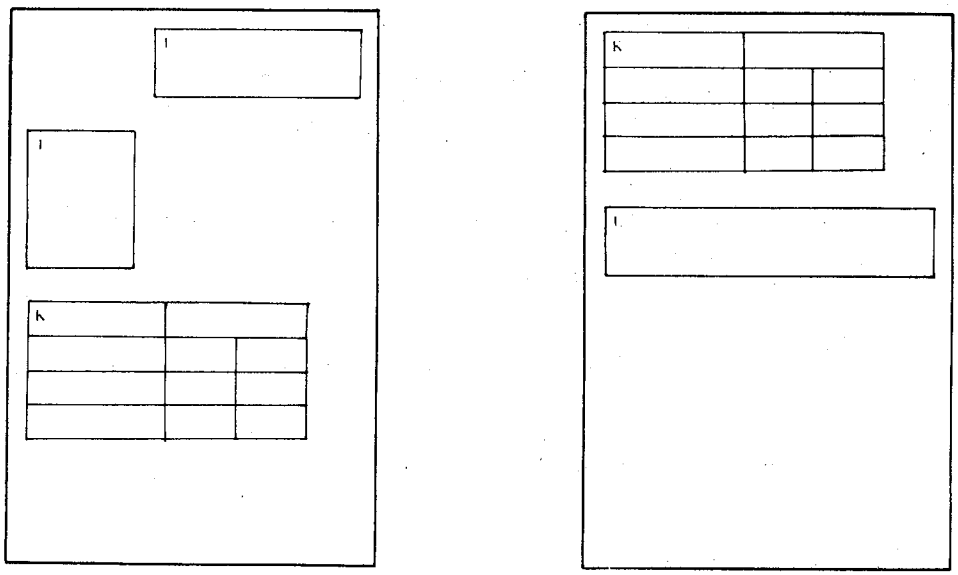


Figure 1.2.36.

Deuxième solution

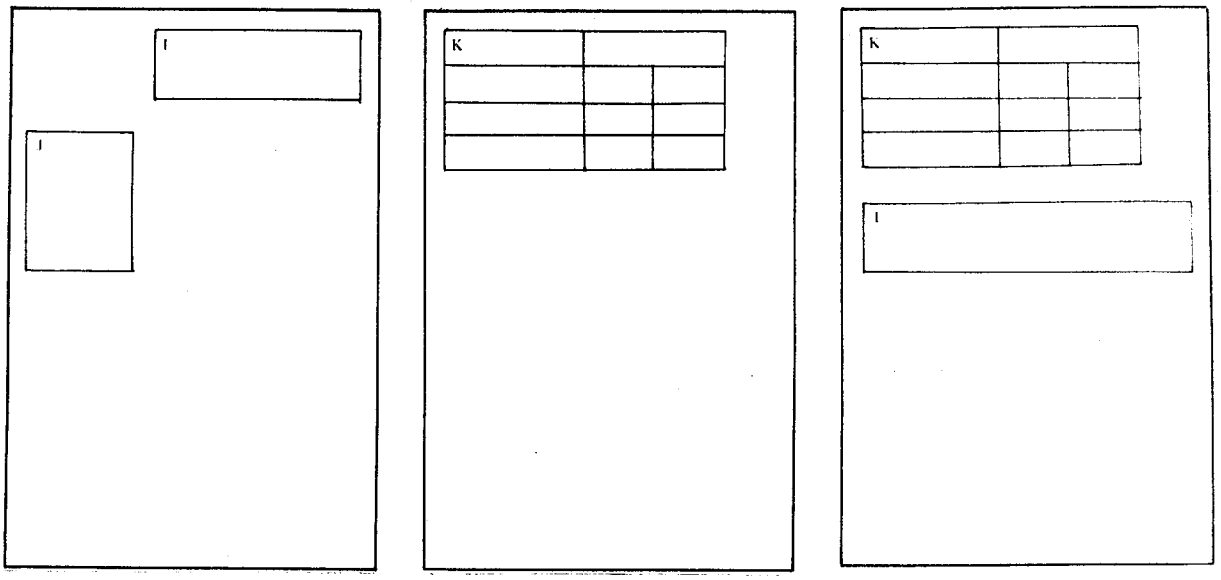


Figure 1.2.37.

En vertu du principe-5, le processus de Décision considère comme satisfaisante la première solution.

2.3.6.2.3. Troisième éventualité

Le rectangle J ne peut coexister sur la page où figurent les rectangles préalablement communiqués.

Il en est ainsi des Pages Physiques ayant, par exemple, pour dimensions celles des rectangles a x" y" d (voir figure 1.2.29).

L'interruption du traitement assumé par le processus d'Adaptation est provoquée par un "Rectangle-Entité".

La nature des informations constitutives du Rectangle-Entité motive alors l'action entreprise.

Considérons que le rectangle J réunisse trois Rectangles-Unités J-1, J-2, J-3.

a/ Supposons que les Rectangles-Unités renferment chacun la représentation d'une information de base simple.
En vertu des principe-1 et principe-3, le Rectangle-Entité J est un groupement insécable d'informations, elles-mêmes insécables. Nous sommes confrontés au même problème que dans la première éventualité (Cf §2.3.6.2.1.)

b/ Supposons que J-1 et J-3 renferment la représentation d'un objet de base simple, alors que J-2 concerne un objet de base composé, en l'occurrence un tableau.
La Page-Virtuelle se présente comme l'indique la figure 1.2.38 ci-après.

- En vertu du principe-1, la décomposition de l'objet de base composé doit être envisagée.

- En vertu du principe-3, l'indissociabilité du regroupement doit être respectée compte tenu de la nature des liens qui unissent J-1 et J-3 à J-2.

Admettons que les Pages-Physiques aient pour dimensions celles du rectangle a x"y" d mentionné sur la figure 1.2.38 .

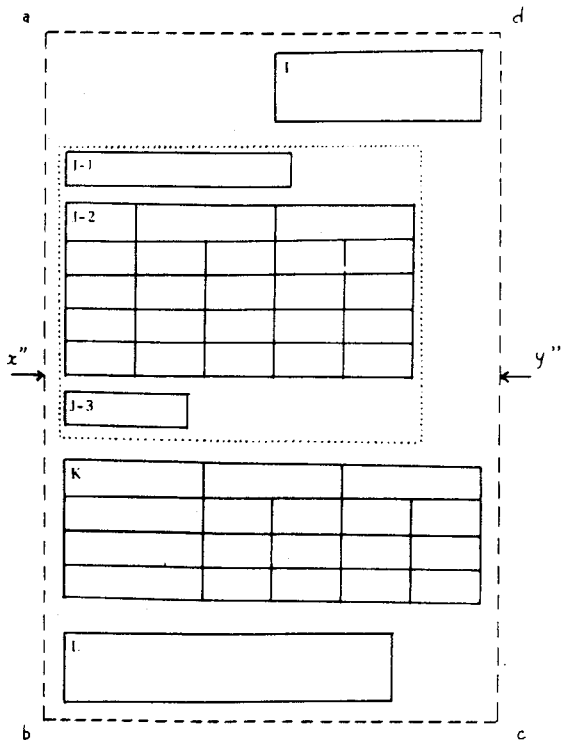


Figure 1.2.38.

- Si les informations contenues dans J-1 et J-3 se rapportent à l'ensemble de celles contenues dans J-2 (Cf §2.3.5.1.2-premier cas).

Première solution

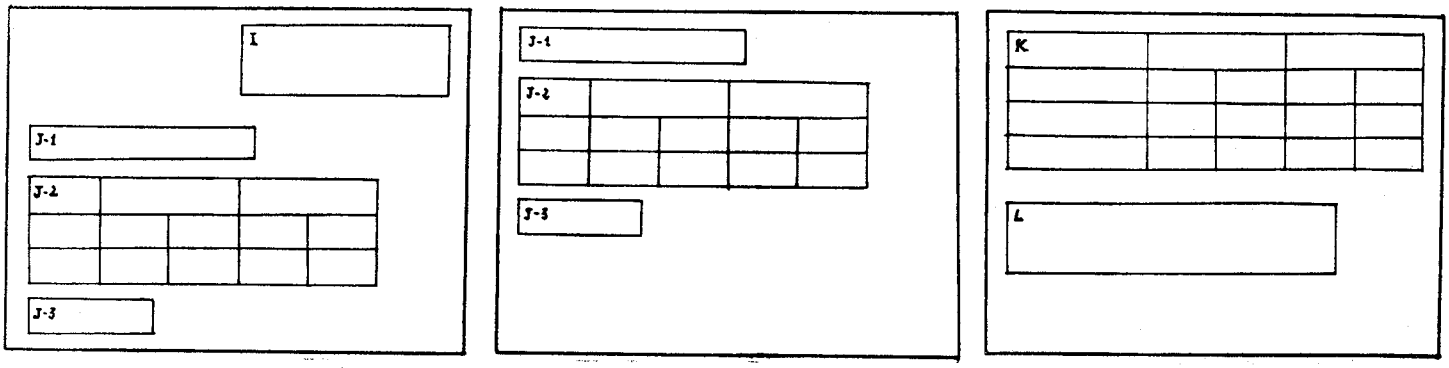


Figure 1.2.39.

Deuxième solution

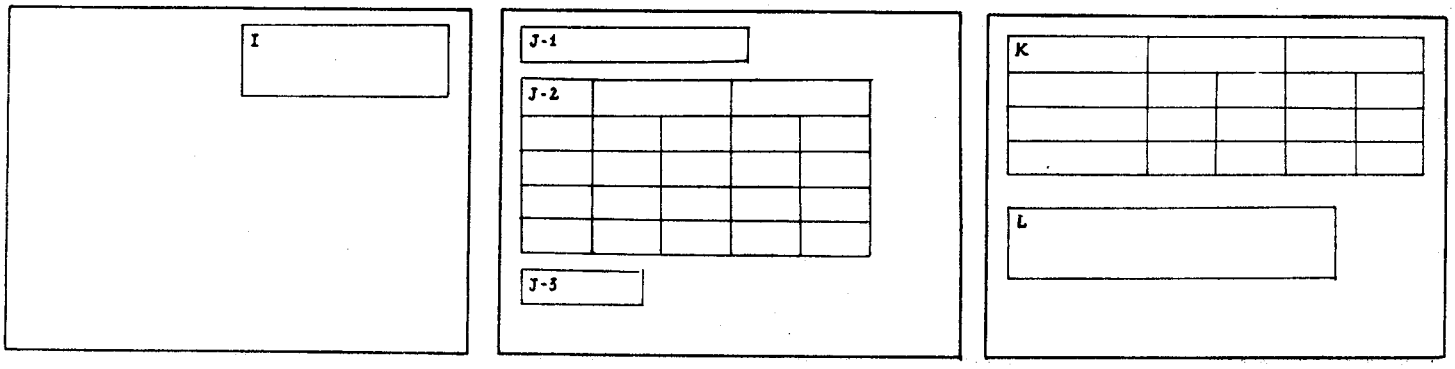


Figure 1.2.40.

En vertu du principe-4, le processus de Décision retient la deuxième solution.

- Si les informations contenues dans J-1 et J-3 se rapportent à celles de tête et de fin de l'objet de base composé (Cf §2.3.5.1. 2-deuxième cas).

Première solution

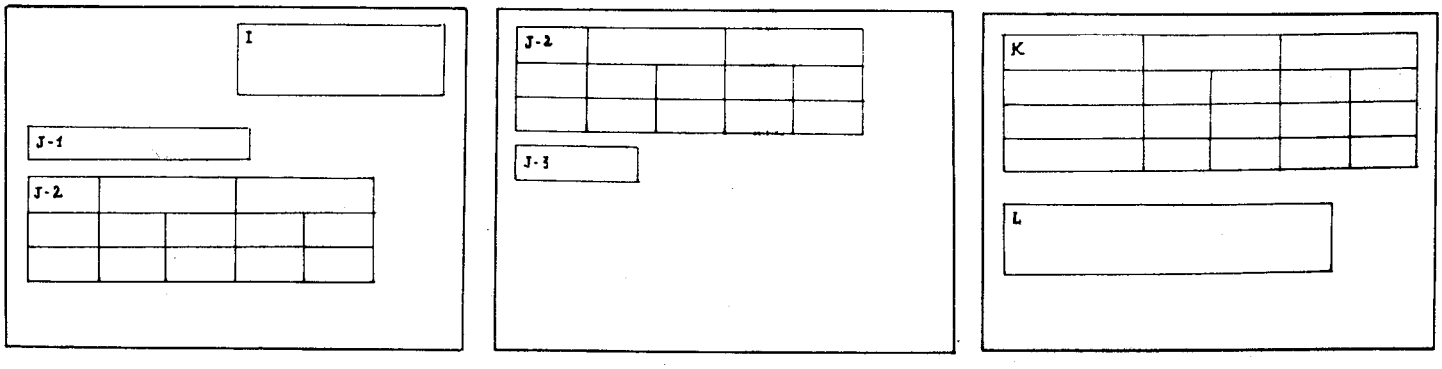


Figure 1.2.41.

Deuxième solution

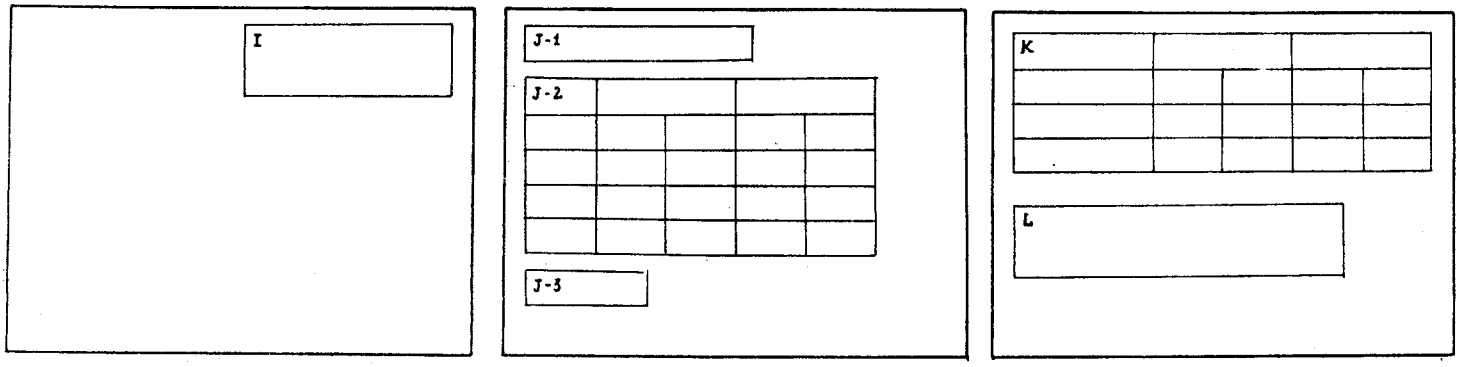


Figure 1.2.42.

En vertu du principe-4, le processus de Décision retient la deuxième solution.

REMARQUES

* Le fonctionnement des processus d'Adaptation et de Décision est identique quelle que soit la nature de l'objet éditable à concrétiser. Dans le cas où il s'agit d'un objet "Répartition", les contraintes physiques alors décrites ne font que minimiser le nombre de solutions envisageables.

* Vu la complexité des problèmes soulevés par une telle automatisation, une restriction, qui ne nuit aucunement à la souplesse de description autorisée, est imposée à l'utilisateur de L.A.M.P.E. :

  parmi les objets éditables simples figurant dans un objet "composition", seul l'un d'entre eux, au plus, peut fixer la représentation d'un objet de base composé.

D E U X I E M E P A R T I E

<><><><><>

DESCRIPTION DU LANGAGE ALGORITHMIQUE

DE MISE-EN-PAGES ET D'EDITION

DEUXIEME PARTIE

C H A P I T R E I

LE MANUEL DE REFERENCE

L.A.M.P.E. est conjointement un langage évolué et un langage de commandes. A ce titre,sa syntaxe, certes conforme à celle de bien des langages évolués, présente certaines particularités.

La manuel de référence se propose de les révéler tout en présentant l'organisation générale du langage et en explicitant la signification, le rôle, et les modalités d'utilisation de ses différents composants.

Remarques préliminaires

Pour la clarté de l'exposé :

a/ La grammaire du langage est fournie en annexe. Dans le courant du chapitre seules certaines règles seront mentionnées. Leur rôle explicatif explique leur forme parfois différente de celle mentionnée en annexe.

- Membres gauche et droit d'une règle sont séparés par le symbole " → "
- La règle est terminée lorsque, sur la ligne suivante, figure le membre gauche d'une nouvelle règle.
- Tout élément d'une règle est séparé par un ou plusieurs espaces.
- Les mots-clés et symboles de base sont soulignés afin de matérialiser leur fonction d'élément terminal.
- Le trait vertical placé entre deux éléments successifs traduit une alternative.
- $[\ ]^{X}$ délimitent un ou plusieurs éléments dont l'occurrence est, soit facultative, soit effective un nombre de fois précisé par la valeur de l'exposant.
- $\{\ \}^{X}$ délimitent un ou plusieurs éléments dont l'occurrence est effective une fois ou un nombre de fois précisé par la valeur de l'exposant.

Ainsi :

Pour x=Ø $[\ ]^{X}$= occurrence 0|1 fois; $\{\ \}^{X}$ = occurrence 1 fois
Pour x=n $[\ ]^{X}$= occurrence 0|n fois; $\{\ \}^{X}$ = occurrence n fois
Pour x=1|...|n $[\ ]^{X}$= occurrence 0|1|...|n fois; $\{\ \}^{X}$= occurrence 1|...|n fois.
Pour x=* $[\ ]^{X}$= occurrence 0|1|...|∞ fois; $\{\ \}^{X}$= occurrence 1|...|∞ fois

en donnant à "∞" la signification: "nombre non limité".

b/ Les exemples seront rédigés conformément aux conventions suivantes :

* Tout mot-clé est représenté par une chaîne de caractères en majuscules et soulignés.
* Un nom de classe est représenté par une chaîne de caractères en majuscules.
* Tout identificateur est représenté par une chaîne de caractères en minuscules.

1.1. LES ELEMENTS CONSTITUTIFS FONDAMENTAUX DU LANGAGE

1.1.1. Les unités syntaxiques

1.1.1.1. Les caractères

Pour écrire les programmes dans le langage L.A.M.P.E. nous disposons de la même gamme de caractères que dans le langage PL/1 :

* Les caractères alphabétiques
* Les caractères numériques
* Les caractères spéciaux

1.1.1.2. Les identificateurs

a/ Un identificateur est représenté par une suite de caractères alphanumériques commençant obligatoirement par une lettre de l'alphabet romain ou par l'un des caractères "@", "#", "$".
Le nombre de caractères, ainsi juxtaposés, ne peut excéder 31. Tout identificateur est précédé et suivi par un délimiteur (Cf §1.1.1.3.) dont fait partie le caractère blanc. Ce dernier ne peut donc être inclus dans un identificateur. Si, pour des raisons de clarté, on veut matérialiser un espacement au sein d'une telle chaîne de caractères, on utilise le seul caractère spécial dont la présence est autorisée: le caractère "souligné".

b/ Certains identificateurs sont dotés, dans le cadre du langage, d'un rôle précis. On les dénomme "Mots-clés".
En vertu de leurs attributions, ils peuvent être répartis en plusieurs catégories :

* Mot-clé introduisant une instruction.
* Mot-clé introduisant une déclaration.
* Mot-clé traduisant une fonction incorporée.
* Mot-clé traduisant un mode.
* Mot-clé traduisant un attribut.
* Mot-clé traduisant un indicateur de position.
* Mot-clé traduisant un délimiteur de portée.
* Mot-clé traduisant un opérateur de mise en relation.

- A tout mot-clé L.A.M.P.E. est adjoint une abréviation dont l'emploi simplifie la rédaction des programmes (voir annexe-6).

- Les mots-clés ne sont pas des mots réservés du langage. Ils peuvent être utilisés, par le programmeur, comme des identificateurs ordinaires.

1.1.1.3. Les délimiteurs

Les délimiteurs sont des caractères spéciaux :

. , : ; ( ) + - * / ! ' '

Ils servent à délimiter Identificateurs et Mots-clés constitutifs de tout programme L.A.M.P.E.

1.1.2. Les composants du traitement

Nous désignons par "composants du traitement" les divers types de combinaison, conformes à la grammaire du langage et associant mots-clés, identificateurs, littéraux, délimiteurs et commentaires de façon à traduire les désidérata du programmeur.

Nous distinguons :

a/ Les composants globaux du traitement communs à la plupart des langages évolués,

* Les Déclarations (Cf.1.2.)
* Les Instructions (Cf.1.6.)
* Les Fonctions incorporées (Cf.1.7.)

b/ Les composants partiels du traitement qui, en raison de la spécialisation du langage, sont dotés d'une importance particulière.

* Les Compléments d'information (Cf.§1.3.)
* Les Requêtes (Cf.§1.4.)
* Les formes structurées (Cf.§1.5.)

1.2. LES DECLARATIONS

Les déclarations ont pour but de donner au compilateur des indications sur l'objet à traiter.

Elles assurent, en respectant l'idée maîtresse et les concepts de base de L.A.M.P.E., la description des objets propres au langage :

- Objets éditables :
  - * Représentation
  - * Composition
  - * Implantation
  - * Répartition

- Objets non éditables :
  - * Modèle
  - * Fichier

Les déclarations ont un rôle essentiel et délicat dans notre langage;

Rôle essentiel, puisqu'elles sont chargées de décrire les diverses combinaisons d'informations dont les imprimés ne sont que la concrétisation.

Rôle délicat, en raison d'une part de la diversité des informations PL/1 à traiter, et d'autre part de la souplesse d'utilisation requise.

1.2.1. La syntaxe des déclarations

DECLARATION	→DECLARE	(1)
	IDENTIFICATEUR MODE [ATTRIBUT] [COMP._INF.] ;	(2)
	CORPS_DE_DECLARATION	(3)
	FIN [IDENTIFICATEUR] ;	(4)
MODE	→{REPRESENTATION | PROPOSITION |MODELE | FICHIER}	
ATTRIBUT	→{FIXE |LIBRE| RESTRICTIF |LINEAIRE| TABULE |	
	STRUCTURE | ARBITRAIRE | PERMANENT}	
COMP_INF.	→ (REF_OBJET)	

a/ L'initialisateur de déclaration (1)

Par analogie avec PL/1, L.A.M.P.E. adopte comme initialisateur de déclaration le mot-clé "DECLARE",plus couramment représenté par son abréviation DCL.

b/ L'en-tête de déclaration (2)

Délimité par un point-virgule, l'en-tête de toute déclaration se compose :

- d'un Identificateur,
- d'un Mode, représenté par un mot-clé du langage. Sa présence est indispensable puisqu'il permet de fixer le type de l'objet. Les modes sont des noms de la langue française.

  Ainsi :

  . Le mode FICHIER caractérise un objet "FICHIER"
  . Le mode MODELE caractérise un objet "MODELE"
  . Le mode REPRESENTATION caractérise un objet "REPRESENTATION"
  . Le mode PROPOSITION caractérise un objet composé éditable.

- d'un Attribut, représenté par un mot-clé du langage et permettant de préciser la signification du mode. Les attributs sont des adjectifs de la langue française.
- d'un Complément d'information, permettant de désigner, dans la mesure où le mode l'implique, l'unique objet dont est déduit celui présentement déclaré (Cf.§1.3.)

EXEMPLE :

"DCL aediter REPRESENTATION TABULE (tableau_pl_1);..."
"DCL mise_en_pages PROPOSITION RESTRICTIVE (implantation)..."

c/ Le corps de la déclaration (3)

La description proprement dite de l'objet en cours de définition a pour siège le corps de la déclaration.

Elle doit être en accord avec le mode adopté.

- Dans le cas où le mode (REPRESENTATION, PROPOSITION) stipule que l'objet est éditable, le corps de la déclaration est composé de requêtes (Cf.§ 1.4.).

- Dans le cas où le mode (FICHIER, MODELE) stipule que l'objet est non-éditable, le corps de la déclaration se présente sous l'aspect d'une hiérarchie d'informations dite "Forme Structurée" (Cf.§ 1.5.).

d/ Le délimiteur de déclaration (4)

C'est par l'association des mots-clé "FIN" (délimiteur de portée) et du caractère ";" (délimiteur) que l'on marque le terme d'une déclaration.

REMARQUE : Au délimiteur de portée "FIN" peut être adjoint l'identificateur mentionné dans l'en-tête de la déclaration.

EXEMPLE : "DCL objet MODE ; ..... ; FIN objet ;"

De même que dans les procédures PL/1, cette indication est une mesure de sécurité. Elle permet, en effet, dans le cas où le corps d'une déclaration est syntaxiquement incorrect, de remédier aux difficultés de compilation et même de valider la déclaration en cause, dans la mesure où une correction est envisageable.

1.2.2. L'initialisation de l'objet déclaré.

Tout objet est simultanément décrit et initialisé.

a/ Si l'objet a pour durée de vie (Cf §1.2.4.) celle du bloc où figure sa déclaration, celle-ci inclut l'initialisation. Ce qui est assimilable à la présence de l'attribut "INITIAL" dans une déclaration PL/1.

b/ Si l'objet a une durée de vie contrôlée, l'initialisation est conjuguée à l'allocation. Ce qui est assimilable à la présence de l'attribut "INITIAL" dans une instruction ALLOCATE de PL/1.

1.2.3. L'affectation de dimensions à l'objet déclaré

Les déclarations définissent indifféremment des objets de base, des objets complémentaires et des objets éditables.

Les objets de base composés, propres ou extérieurs, sont caractérisés par leurs dimensions et par leur structure.

En L.A.M.P.E.,

* Les dimensions de l'objet sont fournies par des indications figurant dans l'en-tête de la déclaration.
* La structure de l'objet est décrite dans le corps de la déclaration.

1.2.3.1. La notion de "Dimension"

Lors de la déclaration d'un objet de base composé, les dimensions précisent le volume des données constitutives du dit objet.

Fixer les dimensions d'un objet de base composé, c'est adjoindre à l'identificateur de ce dernier, au titre de complément d'information, dans le cadre de la déclaration, une liste de scalaires ou de couples de bornes (deux scalaires séparés par ":").

Syntaxe : DIMENSIONS → DIMENSIONS [, DIMENSION]*
DIMENSION → [BORNE :] BORNE
BORNE → SCALAIRE

EXEMPLE : "DCL union (10) FICHIER ...;
:
FIN union;"

1.2.3.2. La notion de "Sélection"

Lors de la participation d'un objet de base composé à un traitement, les sélections permettent d'extraire des sous-ensembles cohérents de données du dit objet.

Sélectionner des données constitutives d'un objet de base, c'est adjoindre à l'identificateur de ce dernier, au titre de complément d'information, une liste de valeurs repérant les composants ou les sous-ensembles de composants à extraire.

Syntaxe : SELECTIONS → SELECTION [,SELECTION]*
SELECTION → [BORNE :] INDICE | ([BORNE :]
INDICE {,[BORNE:] INDICE}*)
INDICE → SCALAIRE | *

REMARQUES :

a/ Lors du traitement d'un objet de base composé, à chacune de ses dimensions peuvent correspondre des sélections.

Exemple : Soit la déclaration : DCL tableau (10,2:5,3) CHAR(20);
On peut avoir lors du traitement tableau((1:4,7:10)2:5,1:3)

b/ De même qu'en PL/1, il est possible en L.A.M.P.E. de sélectionner l'intégralité des données correspondant à une dimension. Au lieu d'utiliser un couple de bornes, on fait appel au caractère "*".
Outre la simplification d'écriture, il est ainsi possible de ne pas tenir compte des fluctuations que pourrait subir la dimension concernée. Un objet de base composé extérieur peut, en effet, être alloué plusieurs fois successives avec, pour chacune de ses dimensions, des valeurs différentes.

c/ A la différence de PL/1, une dimension peut, en L.A.M.P.E., se déduire d'une sélection. Ce qui nous amène à compléter la syntaxe, précédemment donnée, relative aux "dimensions" et à dire :
DIMENSION →{[BORNE:] BORNE|(SELECTION)}
Application :

* dimensions
"DCL union (10) FICHIER...;...; FIN union;"
* sélections
"DCL ... REPRESENTATION TABULE (tableau(5:10,*,(1:6,8:15))";
* dimensions déduites de sélections
"DCL données ((1:10,15;20,25:*))FICHIER ...;...; FIN;"

1.2.4. La portée des déclarations et la durée de vie des objets

1.2.4.1. La notion de portée

Tout objet L.A.M.P.E. est connu dans le bloc qui contient sa déclaration ainsi que dans les blocs intérieurs à celui-ci.

1.2.4.2. La notion de durée de vie

La durée de vie d'un objet L.A.M.P.E. varie en fonction de son type.

a/ L'objet de base a pour durée de vie celle du bloc où figure sa déclaration. La zone mémoire est allouée à chaque entrée dans le bloc et libérée à la sortie du bloc. La déclaration d'un tel objet est, de ce fait, assimilable à une déclaration PL/1 dotée de l'attribut "AUTOMATIC".

b/ L'objet complémentaire et les objets éditables ont tous une durée de vie contrôlée par le programmeur. L'allocation des zones mémoires est conditionnée par l'instruction L.A.M.P.E. Les déclarations de tels objets sont assimilables aux déclarations PL/1 dotées de l'attribut "CONTROLLED".

1.2.5. L'aspect dynamique des déclarations

La description de tout objet propre à L.A.M.P.E. (qu'il s'agisse d'un objet de base, d'un objet complémentaire ou d'un objet éditable) doit pouvoir être influencée par des renseignements fournis au cours de l'exécution du programme.

Ces renseignements pouvant apparaître, de même qu'en PL/1, soit explicitement soit implicitement, une distinction fondamentale s'impose alors entre l'aspect dynamique implicite et l'aspect dynamique explicite.

1.2.5.1. L'aspect dynamique implicite

Comme son qualificatif le laisse entendre, l'aspect dynamique implicite est pris en charge par le système "M.P.E." et ne nécessite aucune intervention du programmeur.

Tel est donc le cas d'un objet éditable simple dont les dimensions sont implicitement identiques à celles de l'objet de base dont il décrit l'aspect.

Prenons pour exemple, la Représentation d'une variable composée PL/1 allouée dynamiquement.

```
DCL  tab_pl_1 (*,*) FIXED CONTROLLED ; /* VARIABLE PL/1 */

     =
     =
DCL  tableau REPRESENTATION TABULE (tab_pl_1 (*,*));
 :
 :
FIN  tableau ; /* DEFINITION DE L'ASPECT DE LA VARIABLE PL/1 */

     —
     —
     —
ALLOCATE tab_pl_1 (5,10) INIT (...) ; /* ALLOCATION PL/1 */
INDICATEUR_D'INSTRUCTION_L.A.M.P.E. (tableau);/*CONCRETISATION*/

     =
     =
     =
ALLOCATE tab_pl_1 (8,16) INIT (...);
INDICATEUR_D'INSTRUCTION_L.A.M.P.E.(tableau);
```

Toute déclaration concernée par l'aspect dynamique implicite s'apparente à une déclaration PL/1 dotée de l'attribut "VARYING"; en effet, il est possible à une variable PL/1 de type "chaînes de caractères" d'admettre implicitement pour longueur, celle de la "constante chaîne" qui lui est affectée [IB-3-1].

1.2.5.2. L'aspect dynamique explicite.

Seuls les objets composés peuvent donner lieu à une déclaration révélant un aspect dynamique explicite.

1.2.5.2.1. La communication dynamique des dimensions.

Tout objet de base composé extérieur ou propre à L.A.M.P.E. est caractérisé par ses dimensions ainsi que par l'ordonnancement et la nature des informations qui le constituent.

Il en est ainsi en L.A.M.P.E. pour l'objet "Fichier" (Cf §2.1.). Un tel objet représente un groupe d'informations en nombre invariable dans le bloc du programme où figure sa déclaration.

Ainsi, l'affectation dynamique des dimensions, si elle est requise, ne peut avoir lieu que lorsque l'on entre dans le bloc d'instructions qui contient la déclaration. Ce qui est en accord avec la notion de durée de vie d'un objet (Cf.§1.2.4.2.).

EXEMPLE :

```
N = I * J + 3; M = N + 5 * K;
BEGIN;
 .DCL ensemble((1:N,M:*))FICHIER PERMANENT (selection(zone));
:      1 CHAR(3),
:      1/*identifications et valeurs*/ (3),
:          2 (CHAR(10), FIXED BIN(31,0));
:FIN ensemble;
```

1.2.5.2.2. La paramétrisation

A la différence des objets composés de base, les autres objets composés propres à L.A.M.P.E. se représentent par un groupe d'informations invariables dans le cadre d'un même bloc du programme.

Il est nécessaire pour l'objet "MODELE" et souhaitable pour les objets "COMPOSITION", "IMPLANTATION" et "REPARTITION" que puisse être précisée dynamiquement leur constitution.

A ce titre, ils sont qualifiés d'objets "paramétrables".

Chaque participation des objets paramétrés à un traitement implique la communication des informations requises.

Paramétrer un objet composé, autre qu'un objet de base, c'est adjoindre à son identificateur, dans l'en-tête de la déclaration, au titre de complément d'information, une liste d'identificateurs séparés par des virgules et dénommés "paramètres formels".

Le format général des déclarations tel qu'il a été précédemment décrit s'en trouve modifié et adopte la forme définitive suivante :

```
DECLARATION          → DECLARE
                       IDENTIFICATEUR [(LISTE_PARAMETRES)] MODE
                           [ATTRIBUT][COMP._INF.][,LISTE_SPECIFICATIONS]
                       CORPS_DE_DECLARATION
                       FIN [IDENTIFICATEUR];
LISTE_PARAMETRES     → PARAMETRE [, PARAMETRE]*
LISTE_SPECIFICATIONS → SPECIFICATION [, SPECIFICATION]*
```

a/ Intérêt de la paramétrisation

La paramétrisation représente l'une des souplesses de description offerte par L.A.M.P.E. Elle présente plusieurs avantages :

- Respecter l'allocation dynamique des objets composés PL/1.

L'édition à l'aide de L.A.M.P.E. de tout objet de base composé nécessite l'élaboration d'objets "Représentation" et "Modèle" (Cf §2.2. et 2.3.)
La paramétrisation permet à ce dernier de s'adapter aux dimensions initialement connues de l'objet de base auquel il s'adresse et d'en respecter les fluctuations éventuelles au cours du traitement.
Considérons à titre d'exemple la séquence suivante :

```
DCL tab_pl_1(*,*) FIXED CONTROLLED;
    =
    =
DCL aadjoindre (m,n) MODELE,...;
 :
FIN aadjoindre;
DCL tableau REPRESENTATION TABULE (tab_pl_1 (*,*));
 :  DIRECTIVE aadjoindre (DIM(tab_pl_1,1),DIM(tab_pl_1,2));
 :
FIN tableau;
```

Ainsi, chaque fois que l'objet Modèle est impliqué dans la Représentation d'une version de "tab_pl_1", son appel nécessite la communication, comme arguments, de deux valeurs numériques déterminées en fonction des dimensions courantes de l'objet de base considéré.
Dans l'exemple présent, nous utilisons les fonctions standards "DIM" de PL/1 pour connaître les dimensions courantes de "tab_pl_1". La directive L.A.M.P.E. appropriée (Cf §1.4) permet de passer ces valeurs, en argument, au Modèle "aadjoindre".

- Permettre de généraliser la description d'un objet éditable composé

Définir un objet éditable composé revient à appliquer des directives précises à des objets éditables pouvant participer à sa composition.
Grâce à la paramétrisation, les directives peuvent porter sur des objets éditables désignés dynamiquement.

EXEMPLE : "DCL aediter (parm_I) PROPOSITION, ...;
DIRECTIVE parm_I ;...;
FIN aediter;"

Ainsi, la concrétisation de cet objet ou sa participation à un traitement ne peut être envisagée sans que soit explicitement mentionné (paramètre effectif) l'objet non précisé lors de la déclaration (paramètre formel).

Ce qui se traduit par :

{ INDICATEUR_D'INSTRUCTION_L.A.M.P.E. (aediter(objet));
ou
{ DIRECTIVE aediter (objet);

b/ Rôle des spécifications

Tel que nous venons de définir leur rôle, les paramètres peuvent indifféremment représenter des objets éditables ou des objets de base simples (scalaires, chaînes de caractères).
Compte tenu de cette diversité, la présence des paramètres, dans l'en-tête d'une déclaration, doit nécessairement être explicitée (tant pour le programmeur que pour le compilateur).
Tel est l'intérêt des "spécifications".

Il appartient aux spécifications de préciser pour chacun des paramètres formels, d'une part son type, et d'autre part, dans le cas où il est représentatif d'un objet éditable lui-même paramétré, les paramètres dont dépend ce dernier.

- Les paramètres représentatifs d'objets éditables sont spécifiés par le mode correspondant (PROPOSITION, REPRESENTATION)

- Les paramètres représentatifs de valeurs numériques, du fait que le mode de ces derniers est défini en PL/1 et non en L.A.M.P.E., sont spécifiés par le mot-clé EXTERIEUR.

EXEMPLES :

"DCL aadjoindre (parm_1 , parm_2) MODELE,(parm_1, parm_2) EXTERIEUR;"
"DCL aediter (parm_1, parm_2, parm_3) PROPOSITION FIXE,
parm_2 EXTERIEUR, (parm_1, parm_3) REPRESENTATION;

- Les objets éditables passés en argument peuvent eux-mêmes être paramètres.

EXEMPLE :

```
"DCL objet (B , E , F , G , C , D , H ) PROPOSITION FIXE,
  .       (B(C , D) , E(F , G , H)) PROPOSITION,
  .        F( G )REPRESENTATION,(C , D , G , H) EXTERIEUR;
  .       <corps_de_la_déclaration>
 FIN  objet;
```

REMARQUES :

Dans l'exemple précité :

- La composition de l'en-tête de la déclaration est en accord avec les règles régissant la composition et l'utilisation des compléments d'information. (Cf § 1.3.3.)
- Une autre fonction confiée aux parenthèses fait son apparition, en l'occurrence; celle bien classique de la mise en facteur. Ici la mise en facteur concerne les modes attribués à plusieurs des paramètres.

1.2.6. Les aspects classiques des simplifications d'écriture

Le lecteur n'ayant pu acquérir à ce stade de l'exposé, toutes les notions auxquelles font appel les exemples illustrant les paragraphes suivants, nous sommes convenus dans toutes les applications citées :

* de substituer à certaines descriptions leurs significations
* de réserver la description exacte aux exemples correspondants mentionnés dans le "Manuel d'utilisation".

1.2.6.1. Les littéraux

- En PL/1, toute constante (représentant une information simple) est traitée sans avoir été préalablement déclarée. C'est un "littéral".

Il s'agit d'une valeur dotée,par défaut, d'un mode précis: ce qui explique les différentes formes possibles :

- Une suite de chiffres.
- Deux suites de chiffres séparées par un point.

} Valeurs de mode "FIXED DECIMAL" se distinguant par la nature du facteur de précision.

- Une suite de caractères délimitée par des "quotes"

} Valeur de mode "CHAR".

- En L.A.M.P.E., tout littéral PL/1 est traité au même titre que les objets de base extérieurs, de même mode, explicitement déclarés. D'où l'équivalence des deux séquences suivantes :

```
DCL libelle CHAR (50) VAR INIT ('CECI EST UN LIBELLE');/*DECLARATION PL/1*/
INDICATEUR_D'INSTRUCTION_LAMPE (libelle) ;
ou
DIRECTIVE libellé ;
```

```
INDICATEUR_D'INSTRUCTION_LAMPE ('CECI EST UN LIBELLE') ;
ou
DIRECTIVE     'CECI EST UN LIBELLE' ;
```

REMARQUE

Par souci d'homogénéité, à l'image de PL/1, L.A.M.P.E. fait fréquemment usage de littéraux.

Tout caractère ou chaîne de caractères essentiellement destiné à être édité (caractère constitutif d'un trait, commentaire, intitulé, etc...) figure généralement dans le texte L.A.M.P.E. comme littéral.

1.2.6.2. Les déclarations implicites

- En PL/1, toute variable simple peut être traitée sans avoir été implicitement déclarée; simplification restrictive puisque seules les variables simples peuvent en bénéficier.

Ainsi par convention :

* Tout identificateur commençant par l'une des lettres I,J,K,L, M,N, est considéré comme étant le nom d'une variable arithmétique de mode "REAL FIXED BINARY".

* Tout identificateur commençant par une autre lettre est considéré comme étant le nom d'une variable arithmétique de mode "REAL FLOAT DECIMAL".

- En L.A.M.P.E., de telles variables sont traitées au même titre que des objets de base simple, de même mode, explicitement déclarés.

REMARQUE

Par souci d'homogénéité, à l'image de PL/1, L.A.M.P.E. adopte le principe de déclaration implicite pour certains objets simples (objets "Représentation" définissant l'aspect d'un objet de base simple).

En effet, la représentation de tout objet de base (simple ou composé, (Cf §2.3.) requiert l'apport d'indications fournies par un objet "Modèle" (§2.3.). Ce dernier regroupe, notamment, des "Modèles de données", bien connues de tout utilisateur de PL/1, et assurant en L.A.M.P.E. les mêmes fonctions : fixer, lors de l'édition, la répartition des caractères constitutifs de toute information simple.

Pour tout objet de base simple destiné à être édité, sans que son aspect ait été explicitement défini (absence de la déclaration de mode REPRESENTATION appropriée), le pré-compilateur adopte, par défaut, une représentation standard et par conséquent, un "Modèle de données" type.

C'est cette possibilité qui implique le caractère optionnel déjà signalé (Cf première partie §2.1.2.2.), de la Représentation et de la déclaration qu'elle implique.

D'où l'équivalence des deux séquences suivantes qui concourent à l'édition de la valeur du même objet de base simple dénommé "variable_pl_1".

```
DCL aadjoindre MODELE ; F(10,2) ; FIN;
DCL aediter REPRESENTATION LINEAIRE (variable_pl_1);
  .          <appel de l'objet "aadjoindre"
  .           et description de la Représentation>;
  .
FIN;
   INDICATEUR_D'INSTRUCTION_L.A.M.P.E. (aediter);
   ou
   DIRECTIVE aediter ;

   INDICATEUR_D'INSTRUCTION_L.A.M.P.E. (variable_pl_1) ;
   ou
   DIRECTIVE variable_pl_1;
```

1.2.6.3. Les simplifications relatives aux étapes régissant la description d'une Edition.

La forme traditionnellement utilisée pour décrire certains traitements se révèle parfois bien lourde eu égard à la simplicité du cas envisagé.

- En PL/1, pour y remédier, on a recours à des simplifications d'écriture dont l'une est illustrée par l'exemple ci-après dans lequel les deux séquences sont identiques.

```
contrôle   : FORMAT (F(3,2) , E(10,8)) ;
PUT EDIT ( valeur_pl_1)  (R(contrôle)) ;

PUT EDIT (valeur_pl_1) (F(3,2) , E(10,8));
```

De telles facilités ne peuvent cependant être considérées comme des simplifications qu'en fonction des circonstances de leur utilisation.

Ainsi, dans l'exemple précédent, la forme simplifiée retenue se révèlerait moins intéressante que l'utilisation classique d'une instruction "Format" si l'information "valeur_pl_1" devait être éditée plusieurs fois dans le même programme conformément au même Modèle de données.

- En L.A.M.P.E., des simplifications d'écriture analogues existent et leur intérêt par rapport à la forme normale est fonction, de même qu'en PL/1, des conditions d'utilisation.

Il en est ainsi pour les objets Représentation, Composition et Implantation.

a/ L'élaboration d'un objet Représentation est inutile si un objet de base simple n'est utilisé qu'une seule fois dans le programme.
A ce titre, les deux séquences suivantes sont équivalentes.

```
DCL  complément MODELE; (F(3,2)) ; FIN;
DCL  aediter REPRESENTATION LINEAIRE (variable_pl_1);
         <appel de l'objet "complément"
          et description de la Représentation>
FIN;
 INDICATEUR_D'INSTRUCTION_L.A.M.P.E. (aediter);
 ou
 DIRECTIVE aediter ;
```

```
INDICATEUR_D'INSTRUCTION_L.A.M.P.E. (variable_pl_1 (F(3,2)));
ou
DIRECTIVE  variable_pl_1 (F(3,2));
```

b/ L'élaboration d'un objet Composition est inutile si les objets éditables qui doivent le composer sont essentiellement destinés à participer à la constitution d'une Page Virtuelle (matérialisation d'un objet Implantation).

Ainsi, les deux séquences suivantes sont équivalentes :

```
DCL entite_1 PROPOSITION FIXE ;
    <requête décrivant l'entité_1> ; FIN ;
DCL entite_2 PROPOSITION FIXE ;
    <requête décrivant l'entité_2> ; FIN ;
DCL aediter  PROPOSITION LIBRE ;
             DIRECTIVE_1 entité_1 ; DIRECTIVE_2  entité_2 ; FIN ;
```

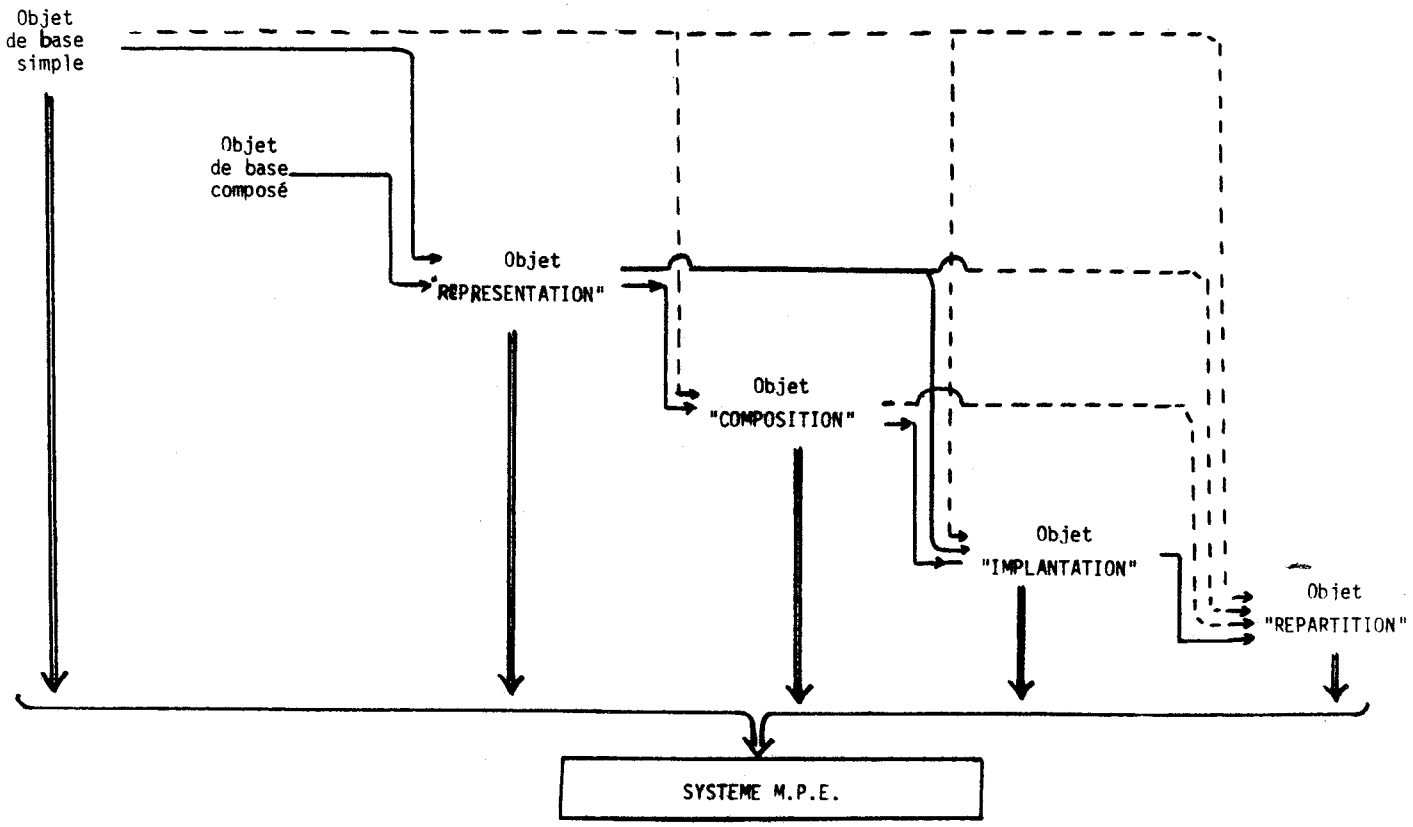
```
DCL aediter  PROPOSITION LIBRE ;
             DIRECTIVE_1 <description de l'entité_1> ;
             DIRECTIVE_2 <description de l'entité_2> ; FIN aediter;
```

c/ L'élaboration d'un objet Implantation est inutile si les objets éditables qui doivent le composer sont essentiellement destinés à être répartis dans des pages Formelles (matérialisation d'un objet Répartition).

Ainsi, les deux séquences suivantes sont équivalentes.

```
{ DCL etape_1 PROPOSITION LIBRE;
{     DIRECTIVE_1  objet_a  ; DIRECTIVE_2  objet_b; DIRECTIVE_3 objet_c; FIN;
{ DCL cefini PROPOSITION RESTRICTIF (etape_1);
{     <requête indiquant la coexistence des objets objet_a,objet_b,objet_c>;FIN;
```

```
{ DCL cefini PROPOSITION RESTRICTIF;
{     <requête indiquant la coexistence de:{-DIRECTIVE_1 objet_a }
{                                          {-DIRECTIVE_2 objet_b } FIN cefini;
{                                          {-DIRECTIVE_3 objet_c>}
```



- - - → : " " " " " " des Simplifications des descriptions.

——→ : " " " " " " du Principe de traitement de L.A.M.P.E.

↓ : peut être concrétisé directement par communication au Système "M.P.E.".

figure 2.1.1.

1.2.7. La désignation d'un objet déclaré

Utiliser un objet revient à le soumettre, en le désignant par son identificateur (sauf si c'est un littéral) aux actions spécifiées par le mode d'une déclaration, par les directives ou par les instructions.

En raison de l'aspect dynamique des déclarations, des dimensions qui peuvent y être mentionnées et des simplifications d'écriture, la désignation d'un objet implique souvent l'adjonction à son identificateur d'un complément d'information. Les précisions ainsi fournies peuvent représenter respectivement des arguments,des sélections de données de base,ou un Modèle de données.

1.2.8. Le schéma récapitulatif

(Voir page suivante).

1.3. LES COMPLEMENTS D'INFORMATION

Le complément d'information est formé d'une liste de membres (ou éléments) séparés par des virgules, placés entre parenthèses et constitués par :

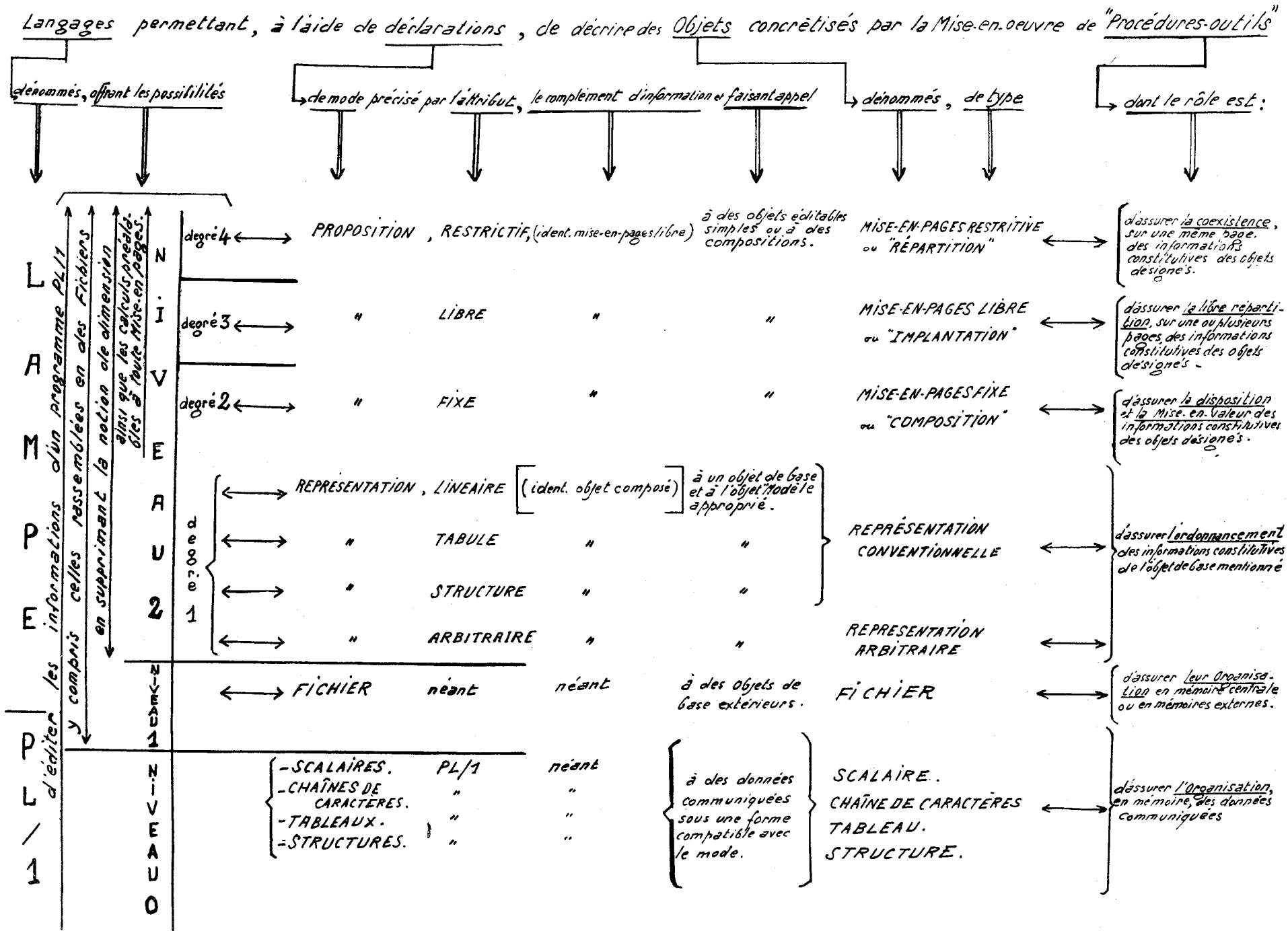
- des Mots-Clés.
- des Identificateurs (éventuellement qualifiés(Ex: struc.élément_1)
- des indices ou des couples de bornes (EX: "3:N")
- des Littéraux (chaînes de caractères, valeurs entières ou décimales)
- des Astérisques.

1.3.1. Le complément d'information peut se rapporter à un mot-clé du langage

* Exemple-1 : "IMPLANTER (ENTETE) a ;"
  Signification: implanter comme en-tête d'une page l'objet désigné par l'identificateur "a".

* Exemple-2 : "SOULIGNER ('.') valeur;"
  Signification: souligner à l'aide d'un trait composé de points l'objet désigné par l'identificateur "valeur".

Dans le cadre de l'EDITION, corrélation entre les niveaux de Langage, les Objets, leur Déclaration, les Outils nécessaires.



* Exemple-3: "DCL aediter REPRESENTATION (inf_1) ..."
  Signification: "aediter" designe un objet dont l'élaboration résulte de l'application de directives de représentation à l'objet de base désigné par son identificateur "inf_1".

1.3.2. Le complément d'information peut se rapporter à un identificateur.

Il représente alors en fonction du contexte :

- Des paramètres formels

  * Exemple: "DCL aediter (parm_1 , parm_2) PROPOSITION ...;..."
    Signification: "aediter" désigne un objet dont la composition fait appel à des objets dont deux d'entre eux, provisoirement désignés par les identificateurs "parm_1" et "parm_2" seront communiqués en cours de traitement.

- Des paramètres effectifs

  * Exemple: "DISPOSER aediter (inf_1,'MONTANT DE LA FACTURE');"
    Signification: Disposer les informations constitutives de l'objet "aediter" dont nous indiquons dynamiquement deux des composants inconnus lors de sa déclaration.

- Des Modèles de données

  * Exemple: "DISPOSER" valeur (F (3.2));"
    Signification: Disposer, conformément au "Modèle de données" mentionné, les informations constitutives de l'objet désigné par l'identificateur "valeur".

- Des sélections

  * Exemple: "DCL aediter REPRESENTATION TABULE
    (tab_pl_1(3,4:8,*));"
    Signification: "aediter" désigne un objet dont l'élaboration résulte de l'application de directives de représentation à un sous-ensemble de l'objet "tab_pl_1".

- Des dimensions

  * Exemple: "DCL union (20) FICHIER PERMANENT (...);...;FIN;"
    Signification: "union" désigne un objet de base composé,
    propre à L.A.M.P.E., et regroupant vingt enregistrements"

1.3.3. Normalisation

Les règles de corrélation et d'imbrication, ci-après énoncées, suffisent à régir l'utilisation et l'organisation des compléments d'information.

Règle de corrélation

Tout complément d'information, de même que tout attribut, se rapporte à l'information précédente la plus proche, autre qu'un attribut ou qu'un élément d'un complément d'information disjoint.

Application

EXEMPLE-1 : "DCL objet REPRESENTATION TABULE (tab_pl_1) ;"

Le complément d'information, fournissant le nom de l'objet extérieur (tab_pl_1) à soumettre à une représentation, se rapporte au mode REPRESENTATION et non à l'attribut TABULE.

EXEMPLE-2 : "PUT LAMPE (aediter) (CARACTERE (60), LIGNE (30));"

Le complément d'information, fixant les dimensions du support de l'édition se rapporte au mot-clé LAMPE et non à l'objet "aediter" qui figure en tant que complément d'information disjoint.

Par contre, en vertu de la même règle, le littéral "30" indiquant le nombre de lignes qui doivent figurer dans la hauteur du support de l'édition, se rapporte, en tant que complément d'information imbriqué, au mot-clé "LIGNE".

Un complément d'information est dit imbriqué si l'information à laquelle il se rapporte est un membre d'un complément d'information non disjoint.

EXEMPLE : "DCL aediter REPRESENTATION TABULE (tab_pl_1 (*,*,3));..."

Règle d'imbrication

L'information à laquelle se rapporte un complément d'information imbriqué peut être indifféremment un mot-clé ou un identificateur à condition que ce dernier ne soit pas l'un des membres d'un complément d'information s'adressant lui-même à un identificateur.

EXEMPLE : "DCL aediter (entité_1(parm_1,parm_2),entité_2)PROPOSITION FIXE,..." est illicite.

1.4. LES REQUETES

Les requêtes ont pour but de décrire la constitution d'un objet éditable en précisant à l'aide de directives et d'opérateurs de mise en relation la disposition de ses composants.

Syntaxe des requêtes

REQUETE	→ COMMANDE DELIMITEUR_REQUETE
COMMANDE	→ [ETIQUETTE] INDICATEUR_COMMANDE OPERANDES DELIMITEUR_COMMANDE
ETIQUETTE	→ IDENTIFICATEUR :
INDICATEUR_COMMANDE	→ DIRECTIVE [RESTRICTIONS]
RESTRICTIONS	→ ({CHAINE_CARACTERES | INDICATEUR_POSITION [, INDICATEUR_POSITION]})
INDICATEUR_POSITION	→ MOT_CLE (cf § 1.4.3.)
DELIMITEUR_COMMANDE	→ FIN [IDENTIFICATEUR]
DELIMITEUR_REQUETE	→ ;
OPERANDES	→ (Cf §1.4.2.)

1.4.1. Les délimiteurs de commande

A tout indicateur de commande doit correspondre un "délimiteur de commande". Représenté par le mot-clé "FIN", il est destiné, comme son nom l'indique, à délimiter la portée d'un indicateur de commande (ou Directive).

1.4.2. Les opérandes

Nous désignons par "opérande" tout objet ou sous-ensemble d'un objet soumis, dans le cadre d'une commande, à l'action spécifiée par la directive. Tout opérande est donc représenté :

* soit par l'identificateur d'un objet éditable,
* soit par l'identificateur d'un objet non éditable,
* soit par un littéral (Cf §1.2.6.1.)

REMARQUE-1

Tout identificateur d'objet, figurant en tant qu'opérande dans une commande, peut être accompagné d'un complément d'information (Cf.1.3.2.).

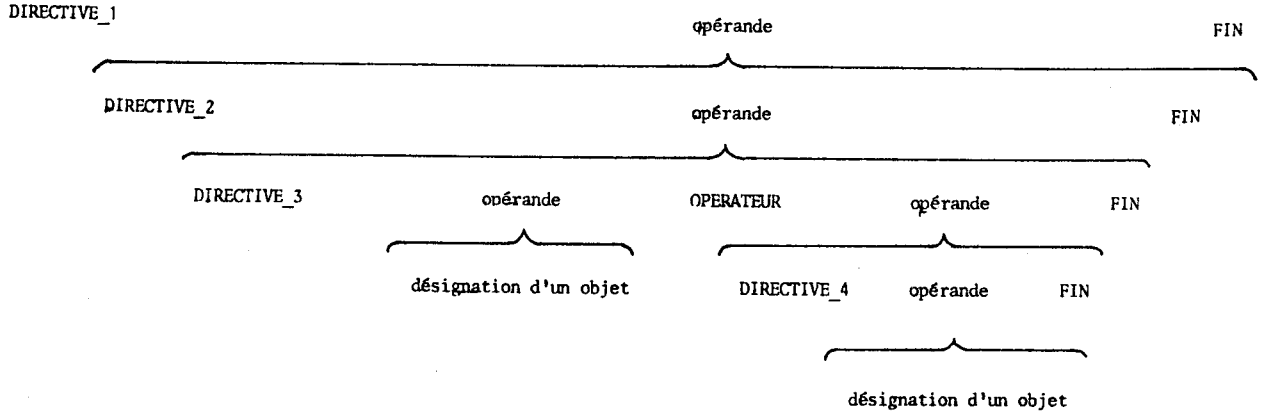
REMARQUE-2

Plusieurs objets peuvent être soumis individuellement et successivement à l'action d'une même directive. Leurs désignations sont séparées par des opérateurs de mise en relation qui matérialisent les liens logiques établis entre les objets considérés.

EXEMPLE : "DIRECTIVE operande OPERATEUR operande ... operande FIN"

REMARQUE-3

Un ou plusieurs objets peuvent être soumis aux actions successives de plusieurs directives. La commande, dans laquelle leurs désignations figurent, est alors considérée comme un opérande soumis à l'action de la Directive précédente



COROLLAIRES

a/ Syntaxe

OPERANDES :OPERANDE [OPERATEUR_DE_MISE_EN_RELATION OPERANDE]*
OPERANDE :{DESIGNATION_D'UN_OBJET | LITTERAL | COMMANDE}

b/ Il nous faut souligner l'analogie existant entre les commandes et les boucles "DO" des langages évolués. Ainsi, les opérandes d'une commande imbriquée sont soumis à l'action de la directive de la commande courante, puis à l'action de la directive de la commande englobante, etc...

1.4.3. Les opérateurs de mise en relation

Nous dénommons "Opérateurs de mise en relation" les trois mots-clés "ET", "PUIS", "AVEC" destinés à matérialiser les liens logiques qui doivent être établis entre deux objets.

Il s'agit d'opérateurs diadiques.

- ET et PUIS sont dénommés : "opérateur de position"
- AVEC est dénommé : "opérateur de liaison".

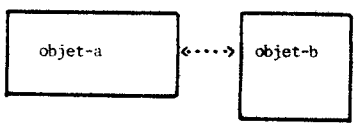
1.4.3.1. Les opérateurs de position

Les opérateurs "ET" et "PUIS" permettent d'indiquer la position, les uns par rapport aux autres, des objets entrant dans la composition d'un objet éditable.

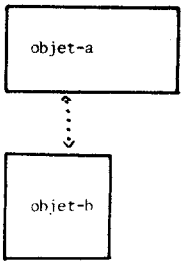
- "ET" indique que les deux opérandes qu'il concerne se font suite latéralement
- "PUIS" indique que les deux opérandes qu'il concerne se font suite verticalement.

EXEMPLE :

"A ET B"



"A PUIS B"



En règle générale, l'opérateur de position fixe toujours la position de l'objet courant par rapport à l'objet préalablement mentionné.

1.4.3.2. L'opérateur de liaison

L'opérateur "AVEC" permet d'indiquer que les informations de deux objets doivent être implantées sur une même page,sans pour cela fixer leur position relative (Cf Manuel d'utilisation §2.4.2.3.).

1.4.4. Les directives

Nous dénommons "Directives" des mots-clés du langage destinés à traduire la nature de l'action à laquelle doivent être soumis les objets entrant dans la composition d'un objet éditable.

Les directives sont toujours des verbes de la langue française pris à l'infinitif (DISPOSER, IMPLANTER,....)

Toute directive peut requérir, en fonction de sa signification, l'emploi d'un complément d'information (restriction), en l'occurrence :

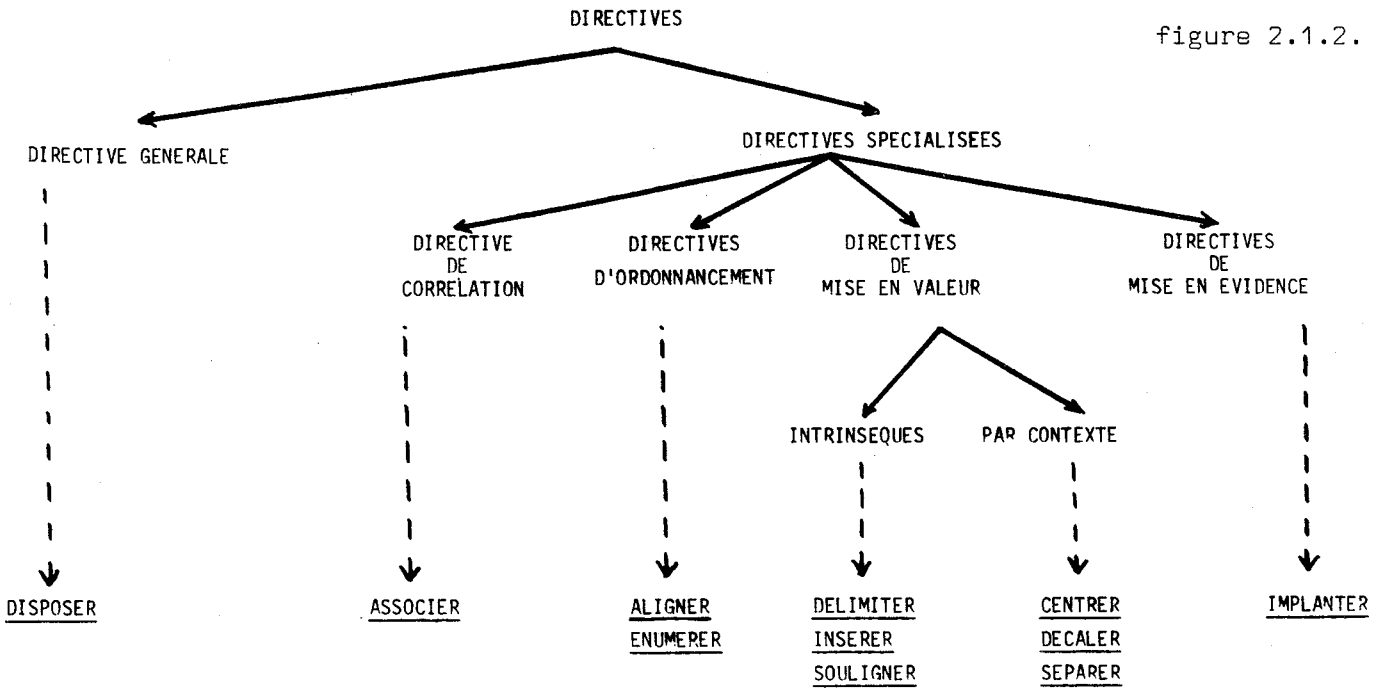
- Soit un mot-clé L.A.M.P.E. (indicateur de position essentiellement)
  EXEMPLE : "IMPLANTER (ENTETE)'RESULTATS OBTENUS APRES 3 ITERATIONS' FIN;"
  Signification : implanter, comme en-tête de page, la chaîne de caractères mentionnée.

- Soit un "littéral" (du type chaîne de caractères)
  EXEMPLE : "SOULIGNER ('.') TOTAL FIN;"
  Signification: souligner les informations constitutives de l'objet "TOTAL" par un trait composé du caractère "."

1.4.4.1. La classification des directives

Les directives se répartissent de la façon suivante :

figure 2.1.2.

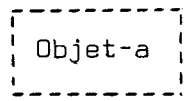


En vertu des concepts de base du langage qui assimilent tout objet à un rectangle, il nous est possible d'expliciter la signification et le rôle des diverses directives.

1.4.4.1.1. La directive générale

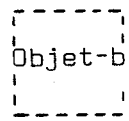
Nous désignons par directive générale une directive ayant la propriété, compte tenu de sa fonction, de pouvoir être employée à l'occasion de la description de n'importe quel type d'objet.

DISPOSER a pour rôle de faire appel et de regrouper, éventuellement, des éléments constitutifs d'un objet éditable.

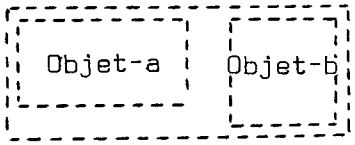


- Appel :

"DISPOSER objet-a FIN"



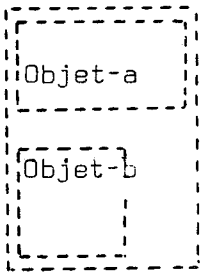
"DISPOSER objet-b FIN"



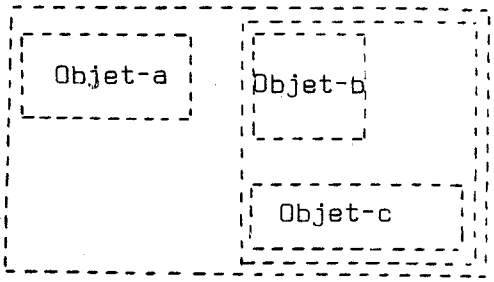
- Appel et regroupement :

"DISPOSER objet-a ET objet-b FIN"

"DISPOSER objet-a PUIS objet-b FIN"



"DISPOSER objet_a ET
DISPOSER objet_b PUIS
objet_c FIN
FIN"



REMARQUE

Le couple "DISPOSER ... FIN" joue dans le cadre des requêtes un rôle similaire à celui des parenthèses dans les expressions arithmétiques ou logiques. Ce qui lui permet du point de vue syntaxique :

a/ de procéder à tout regroupement d'opérandes de façon à les soumettre globalement à diverses actions matérialisées par des directives spécialisées.

En effet, en raison de l'imbrication possible des requêtes, (Cf. Remarque-3 §1.4.2.), nous pouvons écrire :

```
DIRECTIVE
    DIRECTIVE
        DIRECTIVE
            DISPOSER
                objet-a ET objet-b PUIS objet-c
            FIN
        FIN
    FIN
FIN ;
```

- Si le terme "DIRECTIVE" représente des directives spécialisées, la commande précitée illustre un exemple d'application de l'action traduite par chaque directive spécialisée à un regroupement d'opérandes (en l'occurrence : objet-a, objet-b, objet-c).

- Si le terme "DIRECTIVE" représente la directive "DISPOSER", la commande précitée a la même signification que la commande suivante :
"DISPOSER objet_a ET objet_b PUIS objet_c FIN;"

b/ De préciser la nature de la corrélation établie par les opérateurs de mise en relation) lorsqu'ils participent à la Représentation d'un objet de base composé (Cf.Première partie §2.3.5.1.2.).
Soient les deux commandes suivantes :

1/ "DISPOSER x PUIS DISPOSER y FIN ET z ;"

2/ "DISPOSER x PUIS y ET z ;"

Si "y" désigne un objet Représentation fixant l'aspect d'un objet de base composé,

- Dans le cas 1, une corrélation est établie globalement entre l'objet "y" et les objets "x" et "z".
  Ce qui implique que dans la mesure où le dispositif de Formation des pages physiques doit procéder à la décomposition de "y" en sous-ensembles, les informations constitutives des objets "x" et "z" figureront toujours sur la même page que chaque sous-ensemble de l'objet "y".

- Dans le cas 2, une corrélation est établie localement entre l'objet "y" et les objets "x" et "z".
  Ce qui implique que dans la mesure où le dispositif de Formation des pages physiques doit procéder à la décomposition de "y" en sous-ensembles, les informations constitutives de l'objet "x" figureront sur la même page que le premier sous-ensemble alors que les informations constitutives de l'objet "z" figureront sur la même page que le dernier sous-ensemble de y.

N.B. Ces deux variantes, imposées par l'éventail des possibilités de description que L.A.M.P.E. devait offrir, répondent aux principes de fonctionnement du dispositif de Formation des pages physiques.

1.4.4.1.2. Les Directives spécialisées

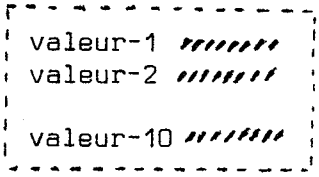
Nous désignons par "Directives Spécialisées" l'ensemble des directives qui ne peuvent être employées que pour décrire des "configurations particulières" tributaires de la nature de l'objet éditable.

a/ La Directive de corrélation
permet de faire appel, dans le cadre d'une Représentation, à l'objet "Modèle" approprié, de façon à ce que puisse être établie la connection entre les éléments constitutifs du Modèle et ceux de l'objet de base concerné.

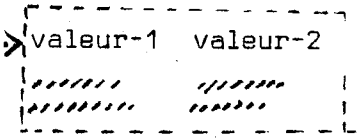
ASSOCIER s'adresse toujours à un objet Modèle représenté par sa désignation.

b/ Les directives d'ordonnancement ont pour rôle d'assurer la ventilation des informations constitutives d'un objet de base composé lors de la représentation à laquelle il doit nécessairement être soumis (Cf.§2.3.).

ENUMERER : répartition verticale des composants .....>



ALIGNER : répartition horizontale des composants ..>



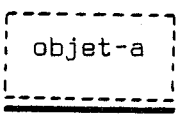
c/ Les Directives de mise-en-valeur intrinsèque permettent de distinguer certains composants d'un objet éditable en leur adjoignant des traits dont on peut fixer les caractères constitutifs à l'aide d'une "restriction".

En l'absence de restriction, c'est un caractère conventionnel qui est adopté :

* le "tiret" pour des traits horizontaux,
* le "I" pour des traits verticaux.

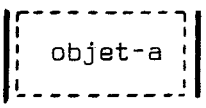
SOULIGNER a pour rôle d'implanter un trait en dessous des informations constituant l'objet désigné.

EXEMPLE : "SOULIGNER objet-a FIN"



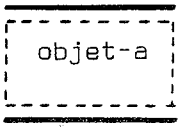
DELIMITER a pour rôle de border les informations constituant l'objet désigné par deux traits parallèles verticaux.

EXEMPLE : "DELIMITER objet-a FIN"



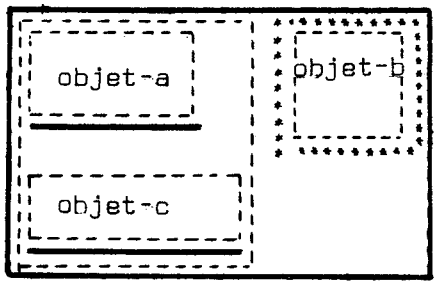
INSERER a pour rôle de border les informations constituant l'objet désigné par deux traits parallèles horizontaux.

EXEMPLE : "INSERER objet-a FIN"



Application générale

"DELIMITER INSERER DISPOSER SOULIGNER objet_a
PUIS objet_c
FIN
ET DELIMITER ('*') INSERER ('*') objet_b FIN FIN
FIN
FIN
FIN"

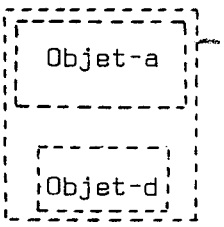


d/ Les Directives de mise-en-valeur par contexte permettent de distinguer certains des objets constitutifs d'un objet éditable en leur imposant une position précise par rapport à d'autres objets.

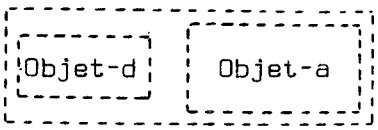
CENTRER a pour rôle de faire coïncider la partie médiane de l'objet concerné et celle des autres objets avec lesquels il est en relation.

EXEMPLES:

"DISPOSER objet_a PUIS
CENTRER objet_d FIN
FIN"



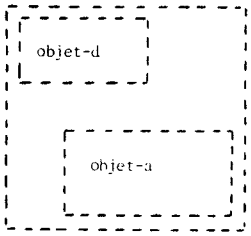
"DISPOSER objet_d
ET CENTRER objet_a FIN
FIN"



DECALER a pour rôle de décaler l'objet concerné par rapport aux objets avec lesquels il est en relation.

EXEMPLE :

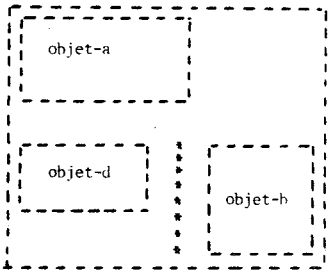
"DISPOSER objet_d
PUIS DECALER objet_a FIN
FIN"



SEPARER a pour rôle de séparer, à l'aide d'un trait dont on peut fixer la composition grâce à une restriction, deux objets entre lesquels est établie une relation.

EXEMPLE :

"DISPOSER objet_a PUIS objet_d
ET SEPARER ('*') objet_b FIN
FIN"



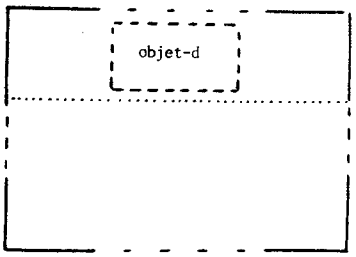
e/ La_Directive_de_mise-en-évidence_permet de donner un relief particulier à des composants d'un objet Implantation, en leur assignant une position privilégiée du rectangle Page-Virtuelle représentatif de l'objet considéré. Une telle directive ne peut apparaître que dans la description d'un objet Implantation.

Par exemple, un libellé implanté comme en-tête des pages d'un livre occupe une position qui, en règle générale, est réservé au titre du livre ou à celui du chapitre.

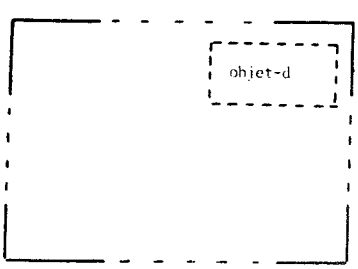
IMPLANTER a pour rôle de fixer l'emplacement de l'objet désigné . Cette directive nécessite l'adjonction d'une restriction désignant cet emplacement et impliquant l'utilisation d'indicateurs de position (HAUT, BAS, DROITE, GAUCHE, ENTETE, PRIORITE)

EXEMPLES :

"IMPLANTER (ENTETE) objet-d FIN"



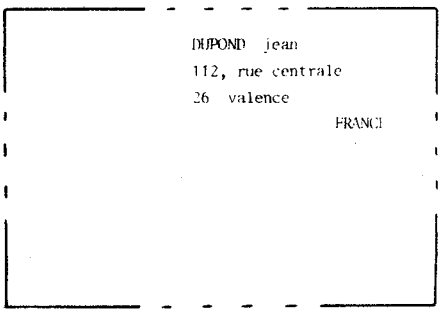
"IMPLANTER (HAUT, DROITE) objet-d FIN"



N.B. Un objet peut nécessiter d'être mis en évidence sans que l'emplacement soit désigné de façon absolue. Libre au système "M.P.E." d'assurer l'implantation optimum en fonction du contexte.

EXEMPLE :

"IMPLANTER (PRIORITE) identité FIN"



REMARQUES

Nous avons volontairement illustré l'action de chaque directive à l'aide d'exemples simples. Nous laissons le soin au manuel d'utilisation de montrer, puis, au lecteur d'évaluer l'étendue des possibilités et la souplesse de description offerte par l'utilisation :

* d'un nombre non limité de directives et d'opérateurs de mise en relation dans le cadre d'une requête,
* de plusieurs requêtes, dans le corps d'une même déclaration, en fonction de la nature de l'objet décrit.

1.4.4.2. La compatibilité des directives avec la nature de l'objet qu'elles décrivent.

Comme leur rôle et leur classification le laissent apparaître, l'utilisation d'une Directive doit être compatible avec la nature de l'objet. D'où le tableau suivant :

NATURE DE L'OBJET		"REPRESENTATION"	"COMPOSITION"	"IMPLANTATION"	"REPARTITION"
DESCRIPTION ENVISAGEE		Rectangle-unité	Rectangle-entité	Rectangle page-virtuelle	Rectangles pages-formelles
DIRECTIVES UTILISABLES	GENERALE	X	X	X	X
	DE MISE EN EVIDENCE			X	⊗
	DE MISE EN VALEUR INTRINSEQUE		X	⊗	⊗
	DE MISE EN VALEUR PAR CONTEXTE	X	X	⊗	⊗
	D'ORDONNANCEMENT	X			
	DE CORRELATION	X			

X : utilisation normale

⊗ : utilisation possible en raison des simplifications de description

1.4.5. L'étiquette

Une étiquette est un identificateur séparé de la directive à laquelle il s'applique par le caractère ":".

EXEMPLE : "debut : DIRECTIVE ..."

Cette étiquette peut figurer derrière un délimiteur de commande.

EXEMPLE : ".....FIN debut"

De même qu'en PL/1 où l'un des rôles de l'étiquette est de simplifier la rédaction de tous blocs imbriqués (qu'ils soient introduits par DO, BEGIN ou PROC), en L.A.M.P.E. l'étiquette intervient pour simplifier l'écriture des commandes imbriquées. Ainsi se trouve supprimée l'occurrence de plusieurs délimiteurs de commande successifs.

EXEMPLE : Soit l'ordre suivant :

```
DIRECTIVE-1
    DIRECTIVE-2
        DIRECTIVE-3
            DIRECTIVE-4
                DIRECTIVE-5
                    identificateur OPERATEUR identificateur
                FIN
            FIN  OPERATEUR identificateur
        FIN
    FIN
FIN
```

Il devient grâce à l'utilisation d'étiquette :

```
prem: DIRECTIVE-1
        DIRECTIVE-2
          DIRECTIVE-3
            Deux : DIRECTIVE-4
                     DIRECTIVE-5
                       identificateur OPERATEUR identificateur
                     FIN deux OPERATEUR-identificateur
      FIN prem
```

1.4.6. Le délimiteur de requête

En délimitant une requête, le point-virgule délimite à fortiori la ou les commandes imbriquées.

En conséquence, les délimiteurs de commande devant apparaître à la fin d'une requête peuvent être omis.

En appliquant cette mesure à la requête prise pour exemple ci-avant, nous obtenons :

```
DIRECTIVE_1
     DIRECTIVE_2
        étiquette_b : DIRECTIVE_3
                           DIRECTIVE_4
                              DIRECTIVE_5
                                 identificateur OPERATEUR identificateur
                        FIN étiquette_b OPERATEUR identificateur ;
```

1.5. LA FORME STRUCTUREE

Tout objet non-éditable propre à L.A.M.P.E., qu'il s'agisse d'un objet de base ou d'un objet complémentaire, est toujours un objet composé (Cf: classification; §2.2.2. première partie).

De même que tout objet composé PL/1, il regroupe selon une organisation rigide et structurée les informations destinées à participer à un traitement.

C'est pour traduire cette organisation que la forme structurée a été adoptée quand il s'agit de décrire la constitution de tels objets.

1.5.1. L'organisation de la "Forme structurée"

La "Forme structurée" L.A.M.P.E. est calquée sur la structure PL/1.

Une forme structurée décrit une arborescence dans laquelle nous conviendrons de désigner, comme à l'accoutumée:

- Noeud de la structure: l'élément dont dépend l'ensemble des éléments de la structure.

- Noeud d'une structure mineure: l'élément dont dépend un sous-ensemble des éléments.
- Elément terminal: tout élément dont ne dépend aucun autre élément de la structure.

EXEMPLE :

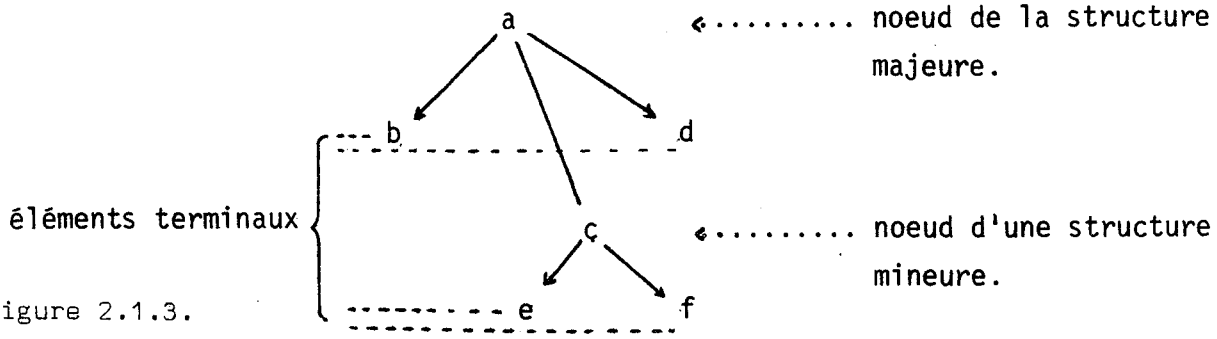


figure 2.1.3.

La traduction de cette arborescence en PL/1 et en L.A.M.P.E. fait apparaître l'identité des méthodes employées.

EN PL/1

```
1 identificateur_a,
   2 identificateur-b CHAR(8),
   2 identificateur_c,
      3 identificateur_e FIXED,
      3 identificateur_f FIXED,
   2 identificateur_d    FIXED;
```

EN L.A.M.P.E.

```
1 élément_constitutif_a,
   2 élément_constitutif_b,
   2 élément_constitutif_c,
      3 élément_constitutif_e,
      3 élément_constitutif_f,
   2 élément_constitutif_d;
```

REMARQUES

Dans une telle organisation :

- La hiérarchie des éléments constitutifs est représentée par un nombre (ou "indicateur de niveau").
- Le point virgule est utilisé pour délimiter la description ainsi programmée (la forme structurée constituant le corps d'une déclaration)

1.5.2. Les éléments constitutifs d'une "Forme structurée".

La nature des éléments constitutifs d'une "Forme structurée" dépend du type de l'objet dont elle constitue le corps de la déclaration.

a/ S'il s'agit d'un objet complémentaire, la forme structurée précise, en vue de leur édition, l'organisation, le format, et éventuellement la signification des informations constitutives d'un objet de base (Cf.2.1.).

Les éléments constitutifs de la forme structurée sont alors :

- des littéraux (chaînes de caractères) destinées à qualifier facultativement une ou un ensemble d'informations constitutives d'un objet de base.
- les modèles de données appropriés et sélectionnés parmi tous ceux disponibles en PL/1 [Cf IB-3-2].

Citons les principaux d'entre eux à titre d'exemple :

* Pour une information du type nombre entier :

  F(W[,d[, p]])

* Pour une information du type nombre flottant :

  E(w , d)

* Pour une information du type nombre complexe :

  { F(w[,d[,p]]) ; E(w,d) } , { F(w[,d[,p]]) ; E(w,d) }

* Pour une information chaîne de caractères ou de bits :

  A(w) et B(w)

Sachant que :

W désigne un scalaire représentant le nombre de caractères à imprimer

d désigne un scalaire représentant le nombre de caractères à droite du point décimal

p désigne un scalaire représentant le facteur d'échelle.

N.B. Les "Modèles de données" ne sont pas parenthésés. Ce ne sont pas des compléments d'information mais des éléments constitutifs de la "Forme structurée".

EXEMPLE :

Objet complémentaire L.A.M.P.E.

```
DCL complément MODELE;
    1 'struct'
      2'_s_struct_1'
         3 'feuille_a' F(7.2)
         3 'feuille_b' A(8),
         3 'feuille_c' E(8.5),
      2's_struct_2',
         3 'feuille_x' A(5),
         3 'feuille_y' A(10);
FIN;
```

Adapté à l'objet de base PL/1

```
DCL 1 struct,
     2 s_struct_1,
       3 feuille_a FIXED,
       3 feuille_b CHAR(8),
       3 feuille_c FLOAT,
     2 s_struct_2 ,
       3 feuille_x CHAR(5),
       3 feuille_y PIC((10)'9');
```

Schéma de la corrélation entre l'objet Modèle et l'objet de base correspondant.

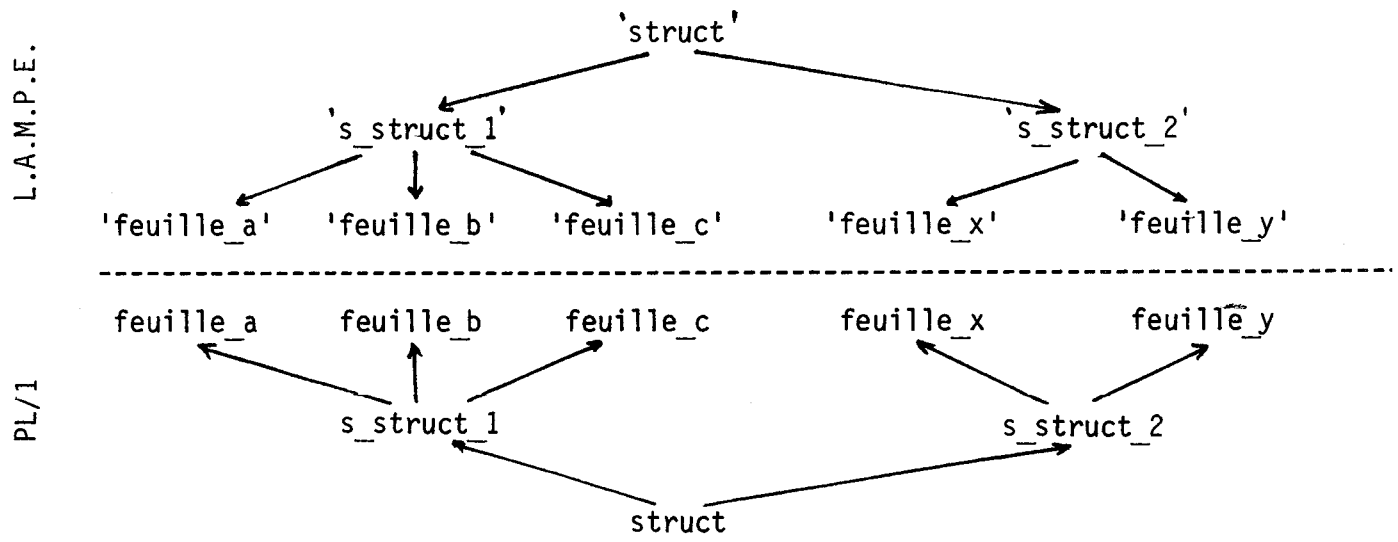


figure 2.1.4.

b/ S'il s'agit d'un objet de base, la "Forme structurée" décrit l'organisation d'un bloc d'information (Cf.§2.3.)
Les éléments constitutifs sont alors exclusivement des "Descriptions de données" puisées parmi celles proposées par PL/1. Il s'agit des attributs PL/1 fixant le mode des variables explicitement déclarées.
Citons les principaux, à titre indicatif.

FIXED, FLOAT, CHARACTER, BINARY, DECIMAL ...

EXEMPLE :

```
"DCL ensemble ((*)) FICHIER PERMANENT (Sélection (carte));
   :    1 (3),
   :       2(FIXED BIN (15), CHAR(2)),
   :    1 CHAR(25);
 FIN ensemble;"
```

REMARQUE :

"Modèles de données" et "Descriptions de données" sont toujours considérés comme les éléments terminaux de la forme structurée.

1.5.3. Les simplifications d'écriture d'une "Forme structurée"

Une forme structurée doit fréquemment traduire une arborescence très étendue.

Afin de simplifier son écriture nous avons recours, selon le cas, à l'une des deux méthodes: la factorisation et l'utilisation d'un facteur itératif.

1.5.3.1. La factorisation

La factorisation consiste à mettre en facteurs les composants d'un même niveau de l'arborescence.

La validité d'une factorisation est conditionnée par le fait que les composants considérés doivent être les noeuds de structures mineures identiques par le nombre et le type des éléments qu'elles comportent.

EXEMPLE , la forme structurée suivante :

```
1'STATISTIQUES',
   2'LIGNE_1',
        3'ELEMENT_A'  F(8.2),
        3'ELEMENT_B'  A(10),
   2'LIGNE_2',
        3'ELEMENT_A' F(8.2),
        3'ELEMENT_B' A(10);
```

peut s'écrire également :

```
1'STATISTIQUES',
   2('LIGNE_1','LIGNE_2'),
        3('ELEMENT_A' F(8.2),'ELEMENT_B'  A(10));
```

1.5.3.2. L'utilisation d'un facteur itératif

Le facteur itératif s'exprime sous la forme d'une liste de valeurs numériques (constantes entières ou scalaires) mise entre parenthèses. Ces valeurs numériques peuvent être associées par couple et séparées par ":" ; Ce sont des bornes.

Le facteur itératif est un complément d'information qui répond à la même syntaxe que les sélections (Cf.§1.2.3.2.).

Le facteur itératif a pour rôle de fixer la répétition des éléments constitutifs du niveau dans lequel il figure, ainsi que des structures mineures correspondantes.

EXEMPLE :

```
1'COUCHE',
  2 (3) 'COLONNE',
     3 (2) 'LIGNE';
```

décrit l'arborescence suivante :

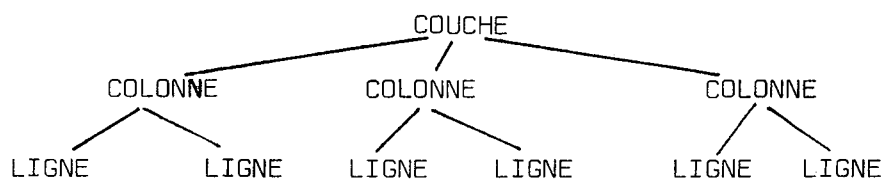


figure 2.1.5.

REMARQUES

a/ En vertu de son rôle, le facteur itératif implique l'existence d'un nombre de variables contrôlées sous-entendues égal à celui de ses membres.

b/ En tant que complément d'information, le facteur itératif respecte la règle de corrélation. Il peut ainsi se rapporter soit à l'indicateur de niveau, soit à un libellé.

- Dans le premier cas, il traduit la répétition du niveau et, par suite, de l'ensemble des indications que ce dernier comporte (voir exemple ci-dessus).

- Dans le deuxième cas, son intervention se solde par un résultat identique au précédent, mais, comme il se rapporte à un libellé, il adjoint à ce dernier la valeur de la ou des variables contrôlées.

EXEMPLE :

```
1  'couche',
  2 'colonne-' (3),
   3 'ligne-'  (2);
```

décrit l'arborescence suivante :

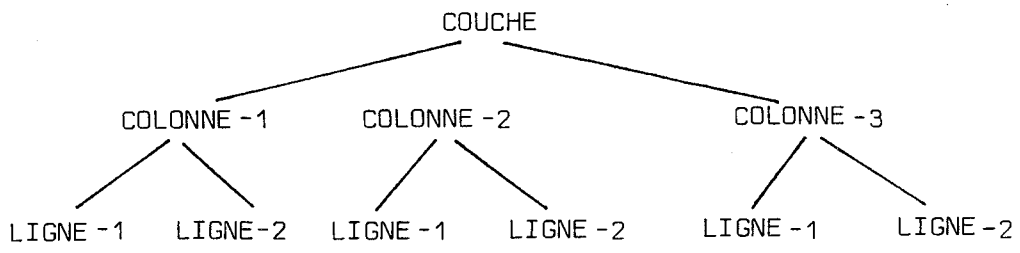


figure 2.1.6.

c/ L'utilisation d'un facteur itératif, outre la simplification qu'il apporte à l'écriture de toute Forme structurée étendue (Cf.§2.1.2.2.), permet à tout objet non éditable de s'accomoder de l'aspect dynamique que peut nécessiter sa déclaration (Cf.§1.2.2.2.1.).

1.6. LES INSTRUCTIONS

Les langages évolués tels que ALGOL, FORTRAN, PL/1 permettent de décrire et d'assurer la réalisation des traitements les plus divers.

Ces traitements ont cependant tous un point commun: ils font appel au moins une fois à l'un au moins des différents types de périphériques disponibles. Ainsi, selon le cas, ce sont l'imprimante, les unités de disques ou de bandes magnétiques, le lecteur-perforateur de cartes ...etc... qui apportent leur concours.

L.A.M.P.E. est, par contre, un langage spécialisé. Les traitements qu'il autorise ont tous la même finalité puisqu'ils se soldent toujours par des éditions. C'est donc l'imprimante qui est généralement sollicitée par les traitements; l'utilisation des unités de disque et bandes magnétiques n'étant requise que pour stocker le contenu des imprimés dont on veut différer l'édition.

L.A.M.P.E. dispose, de ce fait, d'une seule instruction dont le but est d'assurer la concrétisation de l'objet désigné (impression immédiate ou différée des informations que cet objet rassemble).

L'instruction L.A.M.P.E. est conforme au format suivant :

PUT[FILE(nom_du_fichier)] LAMPE(designation_de_l'objet)[(modalités de l'édition)];

1.6.1. Les Fichiers-résultats et les périphériques utilisés

C'est par l'option "FILE(nom_du_fichier)" que le programmeur indique le périphérique qu'il désire utiliser pour prendre connaissance ou conserver les résultats du traitement (concrétisation de l'objet désigné).

En effet, dans le contexte de l'I.B.M.360 un fichier est attaché (statistiquement ou dynamiquement) à une unité secondaire.

L'imprimante étant dans notre cas le périphérique le plus couramment sélectionné, en l'absence de l'option "FILE(nom_du_fichier)", de même que pour l'instruction "PUT" de PL/1, c'est l'option "FILE(SYSPRINT)" qui est adoptée.

1.6.2. La désignation de l'objet

Pour respecter l'homogénéité avec PL/1, la désignation de l'objet (Cf.§ 1.2.2.4.) est mentionnée sous la forme d'un complément d'information se rapportant aux mots-clés "PUT LAMPE".

1.6.3. Les modalités de l'Edition

Les modalités de l'Edition sont traduits par une liste d'options facultatives qui, en raison de leur rôle, constituent un complément d'information. Ce dernier se rapporte, en vertu de la règle de corrélation, aux mots-clés "PUT LAMPE".

Il s'agit d'une "restriction" précisant les modalités de l'Edition.

1.6.3.1. Les modalités relatives au support de l'Editeur

Les dimensions des pages à imprimer influent directement sur l'aspect général de l'Edition.

Il n'est besoin pour s'en rendre compte que de prendre un exemple simple tel que l'Edition d'un tableau PL/1. En fonction de la représentation envisagée, une faible diminution de la largeur des pages réceptrices peut entraîner une modification sensible de l'Edition: division en sous-tableaux imprimés chacun sur une page (Cf.Implémentation) au lieu de l'impression de la totalité du tableau sur une même page.

Nous rassemblons sous la rubrique "Modalités relatives au support de l'Edition" l'ensemble des contraintes auxquelles doivent satisfaire les dimensions du support.

D'où les options
- CARACTERE (arguments)
- LIGNE (arguments)

Syntaxe

CONTRAINTES_DIMENSIONS → CONTRAINTES_DIMENSION [,CONTRAINTES_DIMENSION]
CONTRAINTES_DIMENSION → {CONTRAINTES_HAUTEUR | CONTRAINTES_LARGEUR}
CONTRAINTES_HAUTEUR : LIGNE (ARGUMENTS)
CONTRAINTES_LARGEUR : CARACTERE (ARGUMENTS)

EXEMPLE :

```
PUT LAMPE (aediter) (LIGNE(50),CARACTERE(100));
```

a/ Indication des dimensions de la page

Un seul argument: soit une constante arithmétique,
soit l'identificateur d'une variable arithmétique.

Il représente :

* s'il s'adresse à l'option "CARACTERE", le nombre de caractères qui doivent être contenus dans la largeur de la page à imprimer.
* S'il s'adresse à l'option "LIGNE", le nombre de lignes qui doivent être contenues dans la hauteur de la page.

N.B. Les dimensions conventionnelles de la page d'un "listing" étant de 130/60,

* si l'option "CARACTERE" n'est pas explicitement mentionnée, c'est l'option "CARACTERE (130)" qui est assumée.
* si l'option "LIGNE" n'est pas explicitement mentionnée, c'est l'option "LIGNE(60)" qui est assumée.

b/ Réduction systématique du format de la page

Lorsqu'une instruction "PUT LAMPE" est mise en oeuvre, charge est confiée au système "M.P.E." de répartir, au mieux, sur le nombre de pages nécessaires, les informations regroupées au sein de l'objet désigné.
Il est possible de demander l'édition sur le même nombre de pages mais en réduisant au maximum leurs dimensions (soit la hauteur, soit la largeur, soit soit les deux simultanément).

Deux arguments sont nécessaires :

* La valeur maximale de la dimension considérée (Cf.a
* La valeur du "pas": PAS { - Soit une constante arithmétique / - Soit l'identificateur d'une variable arithmétique)

EXEMPLE :

PUT LAMPE (aediter) (LIGNE(50) , CARACTERE (130, PAS(10)));

c/ Impression d'un tableau de décisions

Au lieu de procéder à l'impression immédiate, il peut s'avérer intéressant de déterminer préalablement la solution la mieux adaptée à la concrétisation de l'objet désigné. Ce qui revient à considérer le nombre de pages nécessaires en fonction de leurs dimensions.

Dans ce but, le programmeur peut demander l'initialisation et l'impression d'un "tableau de décisions".

Trois arguments sont nécessaires :

* La valeur maximale de la dimension considérée ((Cf. a)
* La valeur du "pas" (Cf. b)
* La valeur minimale de la dimension considérée :

A { Soit une constante arithmétique / Soit l'identificateur d'une variable arithmétique

EXEMPLE :

PUT LAMPE (aediter) (LIGNE(50, PAS(5),A(30)),CARACTERE(130));

REMARQUES

Nous désignons par "Tableau de décisions" un tableau réservé par le compilateur L.A.M.P.E. et initialisé par le système M.P.E.

En fonction des principes d'implémentation, ce tableau est systématiquement alloué à chaque interprétation d'une instruction PUT LAMPE (Cf.3e partie §.2.3.3.5.2.).

- Il est constitué de quatre colonnes, chacune d'elles étant destinée à recevoir, rangées par ordre décroissant, d'une part les valeurs des dimensions et d'autre part le nombre de pages correspondant.

- Il regroupe des lignes, en nombre égal au nombre de cas significatifs (ceux, parmi les différents cas autorisés par les indications de l'instruction "PUT", qui traduisent une Edition réalisable).
- Il est constitué d'éléments de décision utilisables tant par le système M.P.E. que par le programmeur.
  En effet, sa désignation par un identificateur invariable V_OOGTABDECIS" rend possible son utilisation dans une séquence programmée.

Conséquences

- Si le programmeur n'est pas fixé sur les conditions que doivent satisfaire les dimensions du support, il décide, au vu du tableau imprimé, de la solution la plus favorable. La concrétisation de l'objet soumis à une telle évaluation ne peut alors être envisagée que lors d'un passage ultérieur du programme.
- Si, par contre, le programmeur est fixé sur les critères de son choix, il peut décider, au cours du même traitement, de la concrétisation de l'objet considéré en testant les indications fournies par le tableau de décisions.

1.6.3.2. Les modalités relatives à l'Edition proprement dite

a/ Sérialisation

Accompagner une Edition d'une sérialisation revient à numéroter les pages et à mentionner au bas de chacune de celles qui contiennent une partie des informations d'un objet composé l'indicatif de continuation" .../..." —
D'où l'option : "SERIALISATION"

EXEMPLE :

```
PUT LAMPE (aediter) (LIGNE(50), CARACTERE(130,PAS(10)),SERI);
```

b/ Itération

Afin de programmer la répétition d'une Edition, une première méthode qui n'appelle aucun commentaire, vient à l'esprit: intégrer une instruction "PUT" dans une boucle "DØ".

EXEMPLE :

```
DO   I = 1  TO  N  BY  1  ;
        PUT LAMPE (aediter) (SERI);
END;
```

Cependant l'instruction"PUT" se solde, comme nous le verrons dans la Troisième partie de la thèse, par deux traitements distincts :
- évaluation de l'implantation des informations,
- Impression effective des pages.

Afin de réaliser une économie appréciable du temps de traitement, il convient de faire porter l'itération essentiellement sur la phase consacrée à l'impression.
D'où l'option : "ITERATION" (argument).

L'argument étant : { Soit une constante arithmétique
                   { Soit l'identificateur d'une variable arithmétique

1.7. LES TRAITEMENTS CONDITIONNELS ET LES FONCTIONS INCORPOREES

La description et la concrétisation d'objets peuvent être conditionnées par la validité de certains tests . C'est ce qui permet aux programmeurs de définir des algorithmes de Mise-en-pages et d'Edition aussi fins qu'ils le désirent.

1.7.1. Les traitements conditionnels

Utilisant, par principe, les ressources du "langage hôte", L.A.M.P.E. fait appel à l'instruction "IF" de PL/1 pour assurer les mises en oeuvre conditionnelles qu'implique la description de tout algorithme de Mise-en-Pages et d'Edition.

Il s'ensuit que le langage PL/1-L.A.M.P.E. dispose d'une instruction conditionnelle se présentant sous la forme suivante :

IF Expression_scalaire PL/1_LAMPE THEN unité_PL/1_LAMPE;
ELSE unité_PL/1_LAMPE;

avec :

a/ Expression_scalaire_PL/1_LAMPE : une expression scalaire PL/1 par sa syntaxe mais qui s'en distingue par l'utilisation possible comme opérandes de valeurs fournies par des fonctions incorporées L.A.M.P.E.

b/ Unité_PL/1-L.A.M.P.E.
- instruction L.A.M.P.E.
- instruction simple PL/1(autre que: DØ, END, BEGIN, FORMAT, ENTRY, DECLARE, PROCEDURE)
- bloc : "DØ ..; inst(s) PL/1-L.A.M.P.E.;END;"
"BEGIN; inst(s) PL/1-L.A.M.P.E.;END;

c/ Instructions_PL/1-L.A.M.P.E.
- instructions PL/1 { de déclaration, de traitement }
- instructions L.A.M.P.E. { de déclaration, de traitement }

1.7.2. Les fonctions incorporées

Les fonctions incorporées L.A.M.P.E. retournent une valeur scalaire ou booléenne selon le rôle qui leur est imparti.

Leurs analogies avec les fonctions incorporées PL/1 ne sont pas que syntaxiques.

C'est ainsi que leur mise en oeuvre, de même que pour les fonctions PL/1, est matérialisée par l'indication de lèur identificateur (mot-clé LAMPE) auquel est adjoint, comme complément d'information, la liste d'arguments nécessaires.

Syntaxe

APPEL_FONCTION_L.A.M.P.E. →[NBPAGE|COMPATIBILITE] CORPS_FONCTION
CORPS_FONCTION →(REFERENCE_OBJET, HAUTEUR_PAGE, LARGEUR_PAGE)
HAUTEUR_PAGE → SCALAIRE
LARGEUR_PAGE → SCALAIRE

Les fonctions incorporées L.A.M.P.E. fournissent des renseignements propres à la concrétisation de tout objet éditable.

Elles sont au nombre de deux.

1.7.2.1. La fonction "COMPATIBILITE"

. Son rôle :

Déterminer si l'édition des informations rassemblées par l'objet désigné est possible ou non. Ce résultat est délivré sous la forme d'une valeur booléenne et est obtenu au terme d'une évaluation prenant en considération la configuration de l'objet, ses dimensions ainsi que celles du support de l'édition.

1.7.2.2. La fonction "NBPAGE"

. Son rôle :

Déterminer le nombre de pages, de dimensions données, nécessaires à l'édition des informations rassemblées par l'objet désigné.

Ce résultat est délivré sous la forme d'une valeur numérique et est obtenu au terme d'une évaluation prenant en considération la configuration de l'objet, ses dimensions et celles du support de l'édition.

DEUXIEME PARTIE

C H A P I T R E II

LE MANUEL D'UTILISATION

Lorsqu'un langage s'applique à des traitements particuliers, et qu'il impose un mode de raisonnement nouveau, il s'avère indispensable de préciser le rapport entre les ressources offertes et leur application à la résolution des différents problèmes concernés.

Le présent chapitre a précisément pour objet de traiter de la façon d'employer les "éléments du langage" pour programmer les différentes phases d'un traitement de Mise-en-pages et d'Edition, conformément au raisonnement proposé.

RAPPEL : Les phases essentielles d'un traitement programmé en L.A.M.P.E.

a/ La phase "Description"

a-1/ Description d'objets de base

Sélection et regroupement de données
- élaboration d'un objet FICHIER
- (déclaration de mode: FICHIER)

a-2/ Description d'objets prêts à l'édition

Représentation
- élaboration d'un objet MODELE
- (déclaration de mode: MODELE)
- élaboration d'un objet REPRESENTATION
- déclaration de mode: REPRESENTATION

Composition
- élaboration d'un objet COMPOSITION
- (déclaration de mode: PROPOSITION et d'attribut: FIXE)

Mise-en-pages
- élaboration d'un objet IMPLANTATION
- (déclaration de mode: PROPOSITION et d'attribut: LIBRE)
- élaboration d'un objet REPARTITION
- (déclaration de mode: PROPOSITION et d'attribut: RESTRICTIF)

b/ La phase "Traitement"

Concrétisation conditionnelle des objets éditables
- élaboration d'un algorithme
- (instructions PL/1-L.A.M.P.E.)

2.1. DECLARATION DE MODE "FICHIER"

L.A.M.P.E., extension de PL/1, doit autoriser le traitement de toutes les données dont l'édition peut normalement être envisagée.

En effet, quels que soient leur nature et leur nombre, les données traitées par un programme PL/1 ne peuvent être soumises au système M.P.E. que si elles se présentent sous la forme d'un objet de base unique.

D'où l'existence de l'objet Fichier qui sert à établir une transition entre, d'une part, les facilités accordées par PL/1 dans le traitement de grandes quantités d'informations, et d'autre part, les contraintes imposées par tout traitement L.A.M.P.E. (Cf première partie §2.2.2.)

Les données constitutives d'un objet peuvent aussi bien être fournies par un fichier rangé en mémoire externe que par des variables PL/1.

- Dans le premier cas, l'emploi d'un tel objet s'impose du fait que ni L.A.M.P.E., ni le système M.P.E. ne doivent tenir compte du mode de consultation caractérisant le fichier source.

- Dans le second cas, l'intérêt de l'objet Fichier transparaît notamment dans les deux applications suivantes, citées seulement à titre indicatif.

a/ Soit un tableau PL/1, de grande dimension, dont la valeur des éléments varie à chaque étape d'un même traitement. De façon à minimiser l'espace nécessaire en mémoire centrale, PL/1 autorise une allocation contrôlée de cette variable [IB-3-2] . A chaque allocation, la valeur antérieure de ses éléments est conservée.

L'objet fichier permet de regrouper, avant de les soumettre à une édition commune, les valeurs attribuées aux éléments du tableau lors des allocations successives.

b/ Soit une variable composée PL/1. Il peut s'avérer nécessaire, avant que de procéder à son édition, d'adjoindre ou de substituer à chacun de ses éléments, le résultat d'un traitement ayant l'élément pour argument (calculs divers, texte extrait d'un dictionnaire et correspondant à une valeur numérique etc...)

L'objet Fichier permet de dresser en mémoire externe la nouvelle version de la variable.

2.1.1. Les caractères spécifiques de l'objet fichier

2.1.1.1. Stockage facultatif des données sélectionnées

L'objet Fichier représente un groupement organisé de données extraites de supports divers.

Selon la nature de ces données, l'objet Fichier peut ou non avoir recours à l'utilisation d'un support de manoeuvre (en l'occurrence un disque magnétique). Par conséquent :

- Si les données sont extraites d'un support magnétique, il est souhaitable de ne pas les transcrire sur le support de manoeuvre; ce qui constituerait une opération redondante.

- Si les données proviennent, par contre, d'un fichier sur cartes ou de variables PL/1, il est nécessaire d'utiliser le support de manoeuvre en vu des traitements que peut imposer le respect des directives de mise-en-pages et d'Edition exprimées par le programmeur. Par exemple, les directives régissant la représentation d'un objet de base peuvent provoquer des balayages successifs des données ainsi regroupées.

La décision d'utiliser ou non un support de manoeuvre incombe au programmeur. D'où le rôle de l'attribut "PERMANENT" qui, par sa présence dans l'en tête d'une déclaration,provoque le stockage.

2.1.1.2. La communication dynamique des dimensions

L'objet Fichier, objet de base composé, réunissant des ensembles structurés de données, peut être doté, de même que les informations composées PL/1, de dimensions communiquées dynamiquement.

Les dimensions sont alors représentées par des variables arithmétiques. En raison de l'assimilation de la déclaration de l'objet Fichier à une déclaration PL/1 de type "AUTOMATIC" (Cf Manuel de Références §1.2.2.2.), ces valeurs effectives des dimensions sont alors celles affectées aux variables arithmétiques lors de l'entrée dans le bloc du programme contenant la dite déclaration.

2.1.2. La description de l'objet Fichier

2.1.2.1. L'origine de la méthode employée

En raison de l'utilisation possible d'un support de manoeuvre, la méthode employée pour décrire l'objet Fichier a été dictée par les principes conventionnels régissant le rangement en mémoire externe.

- Tout fichier résulte de la juxtaposition de zones en nombre proportionnel au volume des données à stocker.

Ces zones :

* sont dénommées "Enregistrements logiques",
* sont identiques les unes aux autres par l'ordonnancement, le nombre et la nature des données qui les composent
* sont distinctes en raison de la valeur de chaque donnée.

N.B. La transcription des enregistrements-logiques sur le support physique incombe au Système. Elle est consécutive à un regroupement éventuel de ces enregistrements en des blocs (ou Enregistrements-physiques) décrits par le J.C.L.

- Par analogie, tout objet Fichier est constitué "d'Enregistrements" et défini par deux critères :

* L'indication du nombre d'Enregistrements.
* La description de chaque Enregistrement (ordonnancement, nombre et nature de ses composants).

2.1.2.2. La description des "enregistrements"

Chaque Enregistrement d'un Fichier est un ensemble structuré de données. La description de ce regroupement est confiée, dans le corps de la déclaration à une Forme Structurée faisant appel, en raison de la nature des informations décrites, aux attributs de mode PL/1 (Cf Manuel de Référence §1.5.2.b.).

EXEMPLE :

```
DCL  union (N)  FICHIER PERMANENT (...);
     1  FIXED BIN (15),
     1  (3)
          2 (CHAR(23), FIXED BIN(31));
FIN union;
```

2.1.2.3. L'initialisation des "Enregistrements"

Chaque "Enregistrement" d'un objet Fichier, vue la diversité d'origin des données qui peuvent y être rangées, est initialisé par une fonction PL/1 écrite par le programmeur. Cette fonction a pour rôle d'initialiser, à chaque appel, un buffer dont l'organisation s'identifie à celle d'un enregistrement.

Il s'agit là d'une technique inspirée de celle nécessitée par l'utilisation, dans des langages de haut niveau, de packages spécialisés. [1B.3.8].

- La rédaction d'une telle fonction répond à une organisation précise qui varie selon la nature du support des données sélectionnées. Citons, à titre d'exemple, la nomenclature de deux de ces fonctions.

a/ Cas de données extraites de variables PL/1.

```
Sélect : proc (zone);
         DCL 1 zone ...; /* description du buffer*/
         <traitement initialisant "zone" à chaque appel>;
         RETURN (zone);
         END select;
```

b/ Cas de données extraites d'un support magnétique externe

```
Lecture : PROC (zone);
          DCL 1 zone ...;/*description du buffer*/
Début   : <ouverture du fichier>;
```

```
Traitement : ENTRY (zone) ;
             IF  <fin de fichier> THEN DO ;
                                    . <fermeture du fichier> ;
                                    : CALL  t_e_r_m_e ;
                                    : GOTO cefini ;
                                   END ;
             lecture d'un enregistrement du fichier
             + initialisation de "zone"  ;
            RETURN (zone) ;
cefini : END lecture ;
```

N.B. Une procédure dénommée "t_e_r_m_e" et fournie par le pré-compilateur assure les traitements impliqués par la détection de la fin d'un fichier-source.

- Les indications relatives à l'appel de la fonction et au point d'entrée de la séquence de traitement figurent, en tant que complément d'information, dans l'en-tête de la déclaration de l'objet Fichier.

EXEMPLE :

```
leccart: PROCEDURE (zone);
         DCL 1 zone ,
             2 code CHAR (5),
             2 ident CHAR (43),
             2 (prix,remise) FIXED BIN (15);
         IF <fin fichier> THEN DØ; CALL t_e_r_m_e; END;
         <lecture d'une carte et initialisation de "ZONE">;
         RETURN (zone);
         END leccart;

DCL 1 carte,
      2 code CHAR (5),
      2 ident CHAR (43),
      2 (prix,remise) FIXED BIN (15);
```

```
DCL fichcar((*)) FICHIER PERMANENT (leccart(carte));
        1 CHAR (5),
        1 CHAR (43),
        1 (2) FIXED BIN (15);
END;
```

2.1.2.4. Le volume de l'objet fichier

L'objet fichier peut être assimilé à un tableau de structures uni-dimensionel.

En effet, il s'agit :

* d'un tableau uni-dimensionel puisqu'il regroupe, les uns à la suite des autres, des composants ayant même configuration (les Enregistrements)
* d'un tableau de structures puisque chaque enregistrement regroupe des données de nature diverse.

Le nombre des Enregistrements constituant l'objet Fichier caractérise son "volume". Celui-ci, comme pour tout objet de base composé, est exprimé sous la forme d'une dimension mentionnée dans l'en-tête de la déclaration.(Cf Manuel de référence §1.2.3.).

L'objet fichier, en raison de son analogie avec un tableau uni-dimensionel, est toujours caractérisé par une seule dimension.

- Le nombre des Enregistrements peut être fixé arbitrairement par le programmeur; la dimension se présente sous la forme:
  * d'une constante entière
    EX: "DCL union (35) FICHIER ..."
  * d'une variable arithmétique simple (aspect dynamique explicite;
    EX : "DCL union (n) FICHIER ..."

- Le nombre d'Enregistrements, du fait que déclaration et initialisation sont des opérations conjointes, peut résulter d'une sélection parmi les buffers fournit par la fonction d'initialisation.

* EX-1 : "DCL union((3:22),24,(26:42))FICHIER...;
* EX-2 : "DCL union((3:1),(m:n))FICHIER...;"

- Le nombre d'Enregistrements peut être tributaire de la source des données. Il s'agit là de l'aspect dynamique implicite.

* EX-1: "DCL union (*) FICHIER ...;"
* EX-2: "DCL union ((3:22)(26:*)) FICHIER...;"

2.2. LA DECLARATION DE MODE MODELE

Tout objet de base,

pour être concrétisé par le système M.P.E.,

* peut s'il est élémentaire,
* doit s'il est composé,

être soumis préalablement à une "Représentation". D'où l'élaboration :

D'un objet "Représentation",

qui impose que l'objet de base et ses composants (s'il en a) aient été initialement caractérisés par :

Un objet "Modèle",

dont le rôle est de fournir toutes les précisions nécessaires sous la forme,

* de littéraux (textes explicatifs),
* de modèles de données.

2.2.1. Le caractère spécifique de l'objet "Modèle".

L'objet Modèle est un objet non-éditable composé.

Il regroupe, de ce fait, des "éléments constitutifs" dont le nombre doit être compatible avec les dimensions éventuellement variables de l'objet de base composé dont il conditionne la représentation.

Il est ainsi fait usage de paramètres qui ne peuvent être que des scalaires.

EXEMPLE : DCL complément (i,j) MODELE,(i,j) EXTERIEUR; ...

2.2.2. La description de l'objet Modèle

2.2.2.1. Les principes régissant la description de l'objet Modèle

En fonction de son rôle, un objet Modèle concerne toujours un objet de base (simple ou composé).

Sachant qu'un objet de base composé (tableau ou structure PL/1, Fichier L.A.M.P.E.) répond toujours à une organisation structurée, ses composants sont regroupés en sous-ensembles. Tel est le cas des composants appartenant :

* à une ligne, à une colonne, à une couche etc... d'un tableau
* à une arborescence d'une structure
* à un enregistrement d'un Fichier.

Les éléments constitutifs d'un objet Modèle s'adressent, de ce fait, soit à des sous-ensembles de composants,soit à des composants pris individuellement. A cet effet, le corps de la déclaration d'un objet Modèle se présente sous une forme hiérarchisée,dénommée précisément "Forme structurée . (Cf §1.5.). Ce qui, tout en étant conforme à la propriété de l'objet Modèle d'être non éditable, présente plusieurs avantages dont les plus significatifs consistent :

* en l'adaptabilité de l'objet Modèle à tous les types d'objet de base.
* en la souplesse de description résidant dans la possibilité de qualifier, de façon plus ou moins détaillée, les composants de l'objet de base concerné.

2.2.2.2. L'adaptabilité de l'objet Modèle aux divers types d'objets de base

a/ L'objet "Modèle" s'adresse à un objet de base simple

* Les éléments constitutifs apparaissent à un même niveau.
* L'emploi du facteur itératif ou de la factorisation est illégal.
* Une simplification d'écriture permet de supprimer le numéro du niveau unique.

EXEMPLE : Objet "Modèle" destiné à caractériser :

- un scalaire

  DCL complément MODELE ;'nombre de produits vendus' F(6);FIN;

- une chaîne de caractères :

  DCL complément MODELE ; A(45) ; FIN ;

b/ L'objet "Modèle" s'adresse à une structure PL/1

Un exemple permet de mettre en relief l'efficacité et l'opportunité de la forme structurée.

EXEMPLE : Objet complémentaire L.A.M.P.E.

```
DCL complément MODELE;
  . 1 'struct'
  .   2'struct_1'
  .     3'feuille_a' F(II.2),
  .     3'feuille_b' A(8),
  .     3'feuille_c' E(8.5),
  .   2's_struct_2',
  .     3'feuille_x' A(5),
  .     3'feuille_y' A(10);
  .
FIN ;
```

Adapté à l'objet de base PL/1

```
DCL 1 struct,
      2 s_struct_1,
        3 feuille_a FIXED,
        3 feuille_b CHAR(8),
        3 feuille_c FLOAT,
      2 s_struct_2,
        3 feuille_x CHAR(5),
        3 feuille_y PIC(10)'9');
```

c/ L'objet "Modèle" s'adresse à un tableau PL/1

Soit l'objet de base extérieur répondant à la déclaration suivante :

```
DCL  tab_pl_1 (3,2,4) FIXED BIN (15);
```

Si nous convenons de désigner respectivement par "ligne", "colonne" et "couche" les coordonnées de chaque élément de cet objet composé, le "Modèle" adéquat se présentera sous l'aspect suivant :

```
DCL  complément  MODELE;
        1 'couche-' (4),
         2 'colonne-'(2),
           3 'ligne-' (3) F (10.2) ; FIN;
```

Principe

Dans la déclaration d'un tableau PL/1, un ordre est imposé pour l'indication des dimensions: nombre de lignes, puis nombre de colonnes, puis nombre de couches etc...

Il en est de même dans la rédaction d'une Forme structurée applicable à un tableau.

Le niveau de plus haut rang (niveau 1) rassemble les indications destinées à caractériser le sous-ensemble des composants du tableau correspondant à la dimension de plus fort poids (celle mentionnée le plus à droite dans la déclaration).

Puis, suivant le même procédé, les niveaux de rang inférieur s'adresseront aux sous-ensembles correspondant aux dimensions de poids décroissants.

Ce qui se schématise de la façon suivante :

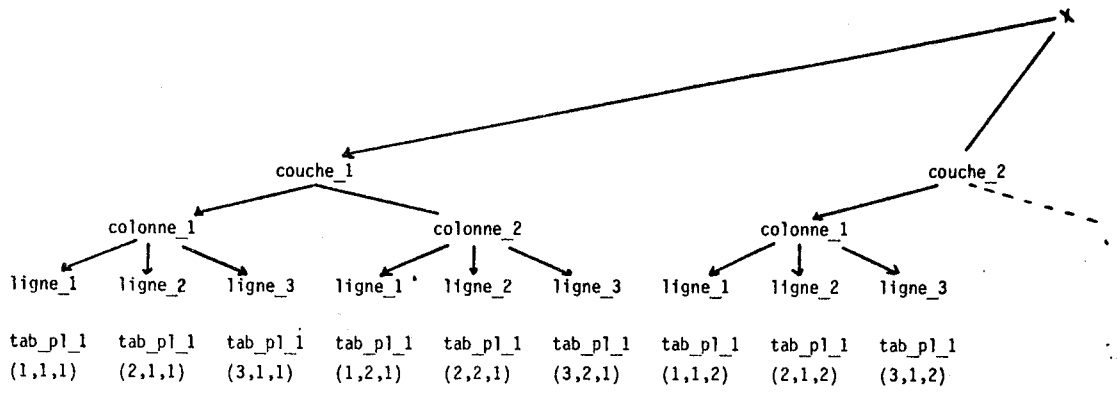


figure 2.2.1.

Corollaire : Application à un tableau de structures.

```
Objet de base : DCL  1  table(4)
                        2 branche_1,
                          3 feuille_a  CHAR (2), 3 feuille_b CHAR (5),
                        2 branche_2,
                          3 feuille_a FIXED, 3 feuille_b FLOAT;

Objet Modèle  : DCL indications MODELE;
                      1 'tarif applicable aux produits de la catégorie.'(4)
                        2'identification',
                            3('code-objet' A(2), 'code-usine' A(5)),
                        2'cout',
                             3('prix-unitaire'F(5.2),'remise' E(5.2);FIN;
```

d/ L'objet "MODELE" s'adresse à un "Fichier" L.A.M.P.E.

L'objet Fichier étant assimilé à un tableau de structures (Cf §2.1.2.4.) l'exemple précédent s'applique à ce cas-ci.

2.2.2.3. La souplesse de description

La possibilité de qualifier, de façon plus ou moins détaillée, les composants de l'objet de base auquel le Modèle s'adresse se traduit par une extension plus ou moins poussée de la forme structurée.

Soit, l'objet "tab_pl_1" cité ci-avant.

Au lieu de concevoir un "Modèle" matérialisant les différentes classes de composants, nous aurions pu indistinctement :

- Soit minimiser les indications apportées:

```
DCL complément_1  MODELE;
  : 1'élément'  (3,2,4) F(10.2);
.
FIN;
```

- Soit diversifier davantage les indications nécessaires,

```
DCL complément_2  MODELE;
    1('bilan financier de 1971-1972', 'bilan financier de 1972-1973',
        2('usine_a', 'usine_b'),
           3('premier semestre','deuxième semestre'),
              4 'dépenses' F (8.3),
              4 'recettes' F (10.2),
              4 'progression' F(6.2); FIN;
```

2.3. DECLARATION DE MODE REPRESENTATION

Tout objet de base simple, ne peut être soumis au "Système M.P.E." en vue de l'édition de sa valeur, sous un aspect autre que celui impliqué par sa déclaration, que si sa représentation a été préalablement définie.
D'où l'élaboration explicite

Tout objet de base composé, ne peut être soumis au "Système M.P.E en vue de l'édition des valeurs qu'il comporte, quel que soit l'aspect souhaité, que si sa représentation a été préalablement définie.
D'où l'élaboration (Cf première partie §2.1.2.2-a).

d'un objet "Représentation";

Lequel impose l'apport, pour l'objet de base concerné ou pour chacun de ses composants (s'il en a), des indications fournies par l'objet "Modèle" approprié

Procéder à une déclaration de mode Représentation revient à décrire un objet éditable destiné, comme son nom l'indique, à fixer l'aspect que l'on souhaite donner à un objet de base, tout en respectant l'idée maitresse du langage (Cf.I partie §1.3.2.)

Plus concrètement, c'est décrire la constitution d'un "Rectangle-Unité" (Cf.1ère partie §2.1.3.1.).

2.3.1. Les caractères spécifiques de l'objet "Représentation".

L'objet "Représentation" est un objet éditable simple.

A ce titre, qu'il fixe en vue de l'édition l'aspect d'un objet de base simple ou composé, il s'identifie, de même que les informations simples de tout langage évolué, à une donnée qui ne peut être traitée que dans sa totalité.

Ainsi, l'objet Représentation,

* en tant qu'objet simple ne peut être paramétré (Cf §1.2.5.2.
* en tant qu'objet simple construit à partir d'un objet de base, adopte implicitement pour dimensions celles de l'objet de base concerné (Cf.§1.2.5.1.).

2.3.2. La désignation de l'objet de base

L'objet Représentation est déduit soit d'un objet de base (simple ou composé) soit d'un sous-ensemble d'un objet de base composé. Dans ce dernier cas, les règles de sélection à appliquer (Cf.§1.2.3.2.).

- sont celles de PL/1. D'où par exemple :
  "DCL aediter_1 REPRESENTATION ATTRIBUT(tab_pl_1(*,*2));..."
  "DCL aediter_2 REPRESENTATION ATTRIBUT(struct_pl_1.élément_a);..."

- peuvent être plus poussées que celles proposées par PL/1, dans la mesure où l'objet de base est "dimensionné" (Tableau,Fichier). D'où par exemple :
  DCL aediter_3 REPRESENTATION ATTRIBUT(tab_pl_1(*,(1:5,8:12),2));...

2.3.3. L'application de l'idée maitresse à la description de l'objet Représentation

L'objet Représentation fixe la disposition, jugée la plus claire et la plus intelligible, d'un objet de base, sans nécessairement en calquer l'organisation intrinsèque.

Pour parvenir à l'élaboration d'un tel objet, L.A.M.P.E., conformément à l'idée maitresse, impose au programmeur de prendre diverses décisions indépendantes de la notion de dimensions (Cf.1ère partie §2.1.3.1.)

Ainsi,étant donné un objet de base, quelle qu'en soit la nature, décrire sa représentation revient à préciser :

* *l'aspect global qu'il doit revêtir lors de son édition,*
* *l'ordonnancement le plus favorable de ses composants, s'il en a,*
* *la composition et l'implantation des traits de séparation éventuellement nécessaires à la délimitation des composants ou des sous-ensembles de composants.*

Tels sont les trois stades du raisonnement que nous allons, ci-après, expliciter en distinguant d'une part la méthode à suivre pour respecter l'idée maitresse, et d'autre part la façon d'exprimer ces desiderata en L.A.M.P.E.

2.3.4. L'aspect global d'un objet de base dans le cadre de l'édition .

Définir l'aspect global que doit revêtir un objet de base lors de son édition, revient à préciser le type d'ordonnancement de ses composants (disposition sous forme d'un tableau, d'une structure, etc...).

La représentation d'un objet de base est tributaire des indications incluses dans l'objet Modèle approprié (Cf §2.2.).

Il s'ensuit que le choix de l'aspect global implique la disposition relative de l'ensemble des composants de l'objet de base et de l'ensemble des libellés fournis, si besoin est, par l'objet Modèle sélectionné.

2.3.4.1. Le mode de raisonnement dicté par l'idée maitresse .

L'aspect global de tout objet de base peut indifféremment être "conventionnel" ou "arbitraire".

2.3.4.1.1. Les représentations conventionnelles .

En adoptant une représentation conventionnelle, on convient de donner à un objet de base un aspect conforme à celui habituellement retenu pour l'édition de telles données (simples ou composées).

Ainsi, une représentation conventionnelle peut être :

* linéaire
  Les informations constitutives de l'objet de base sont alo éditées sous forme d'une chaîne de caractères (pouvant inclure des caractères blancs).

* structurée
  Les informations constitutives de l'objet de base sont alo éditées sous forme d'une structure.

* tabulée
  Les informations constitutives de l'objet de base sont alo éditées sous la forme d'un tableau.

a/ Les configurations-types

Les Représentations conventionnelles précitées offrent, chacune, des variante
A cet effet nous sommes convenu de préciser les différents aspects conforme
à une Représentation conventionnelle, à l'aide de schémas dénommés
"Configuration-type".

. Configuration-type linéaire

Une"configuration-type" correspondant à une Représentation linéaire est
conforme à l'implantation suivante :

Schématiquement :

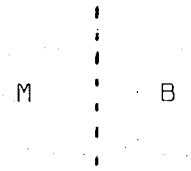
Concrètement :

NOMBRE D'ORDINATEURS EN SERVICE : 9000

. Configuration-type structurée

Une "configuration-type" correspondant à une Représentation structurée
est conforme à l'implantation suivante :

Schématiquement :



Concrètement :

ANNEE 71-72

RECETTES

DEPT.ELECTRICITE	3515250,00
DEPT.ELECTRONIQUE	2312530,00
DEPT.MECANIQUE	1223820,00

DEPENSES

DEPT.ELECTRICITE	2485810,00
DEPT.ELECTRONIQUE	1100500,00
DEPT.MECANIQUE	853200,00

ANNEE 72-73

RECETTES

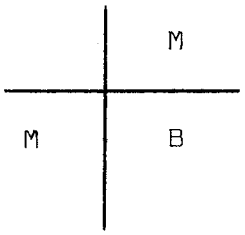
DEPT.ELECTRICITE	4253200,00
DEPT.ELECTRONIQUE	3872150,00
DEPT.MECANIQUE	1327100,00

DEPENSES

. Configuration-type tabulée

Une "configuration-type" correspondant à une Représentation tabulée peut êt
conforme à l'implantation suivante :

Schématiquement:



Concrètement:

		DEPT. ELECTRICITE	DEPT. ELECTRONIQUE	DEPT. MECANIQ
ANNEE 71-72	RECETTES	3515250,00	2312530,00	1223620,(
	DEPENSES	2465810,00	1100500,00	853200,(
ANNEE 72-73	RECETTES	4253200,00	3872150,00	1327100,(
	DEPENSES	2938500,00	1872000,00	982700,(

REMARQUE :

Les trois "configurations-types" citées ci-dessus, bien que d'aspect fondamentalement différent, ont un point commun : elles comportent toujours deux zones distinctes :

* La zone "B" destinée à recevoir les composants de l'objet de base, lesquels seront désormais désignés par le sigle C_B.
* Les zones "M" destinées à recevoir les libellés figurant dans les éléments constitutifs du "Modèle" sélectionné: libellés qui seront, désormais, désignés par le sigle L_M.

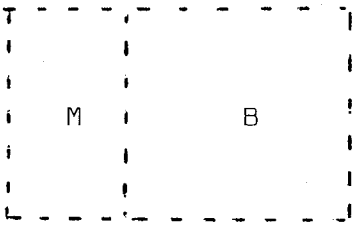
Définition : On appelle "configuration-type" le schéma représentatif d'une disposition, traditionnellement admise, des composants de l'objet de base et des libellés fournis par le "Modèle" approprié.

b/ Le schéma-directeur

Chaque forme de représentation conventionnelle respecte un nombre limité de "configurations-types" regroupées en un schéma récapitulatif que nous dénommerons "Schéma-directeur".

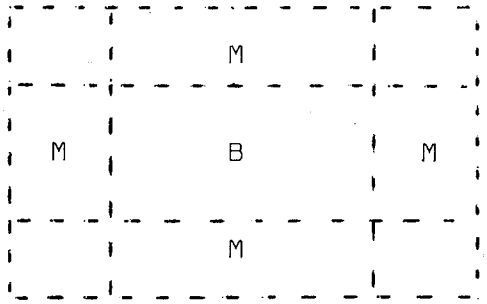
. Schéma-directeur d'une Représentation linéaire .

. Schéma-directeur d'une Représentation structurée .



Il s'identifie à l'unique configuration-type autorisée par une Représentation structurée.

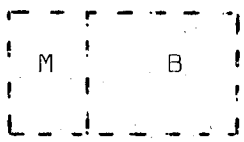
. Schéma-Directeur d'une Représentation tabulée .



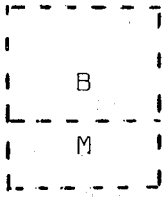
Il regroupe les 8 "configurations-types" permettant au programmeur

- Soit, d'ériger les libellés en intitulés de lignes ou de colonnes. (Représentation Tabulée mono-dimensionnelle).

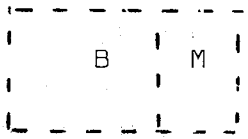
Conf.type I



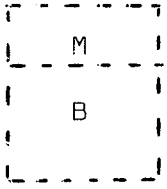
Conf.type 2



Conf.type 3

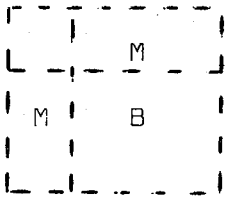


Conf.type 4

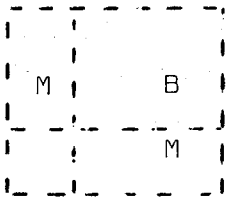


- soit,d'ériger les libellés en intitulés de lignes et de colonnes. (Représentation Tabulée bi-dimensionnelle).

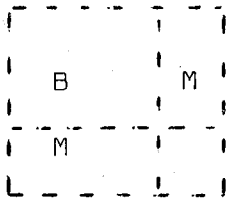
Conf.Type 5



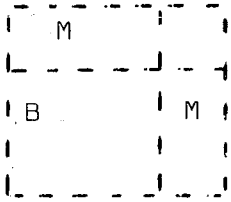
Conf.type 6



Conf.type 7



Conf.type 8



Définition : On appelle "schéma-directeur" la synthèse des différentes "configurations-types" que peut offrir une Représentation conventionnelle linéaire, structurée ou tabulée.

Conséquence : Adopter une Représentation conventionnelle revient à choisir un schéma-directeur et par suite à fixer, sans considérations de dimensions, l'aspect global d'un objet de base.

2.3.4.1.2. La Représentation arbitraire

Une "Représentation conventionnelle" fixe la disposition de C_B par rapport à L_M; c'est la raison pour laquelle le Programmeur est astreint à se conformer à la "Configuration-type" qu'il juge susceptible de favoriser l'interprétation des informations fournies par l'objet de base.

Par opposition, une "Représentation arbitraire" se caractérise par la libre disposition de C_B et L_M; le programmeur peut, de ce fait, concevoir la configuration qui lui semble la plus favorable à l'interprétation des informations fournies par l'objet de base. Cette configuration, librement élaborée, résulte de la disposition ordonnancée des zones B et M.

EXEMPLE :

ANNEE 71-72

DEPENSES		RECETTES	
DEPT.ELECTRICITE	3515250,00	DEPT.ELECTRICITE	2465810,00
DEPT.ELECTRONIQUE	2312530,00	DEPT.ELECTRONIQUE	1100500,00
DEPT.MECANIQUE	1223620,00	DEPT.MECANIQUE	853200,00

ANNEE 72-73

DEPENSES		RECETTES	
DEPT.ELECTRICITE	4253200,00	DEPT.ELECTRICITE	2938500,00
DEPT.ELECTRONIQUE	2872150,00	DEPT.ELECTRONIQUE	1872000,00
DEPT.MECANIQUE	1327100,00	DEPT.MECANIQUE	982700,00

2.3.4.2. La façon d'exprimer, en L.A.M.P.E. l'aspect global souhaité

Le choix de l'aspect global est une notion caractérisantl'ensemble de l'objet "Représentation". C'est donc dans l'en-tête de la déclaration que doit être exprimée la forme retenue.

D'où l'utilité des attributs :

- LINEAIRE
- TABULE
- STRUCTURE

} caractérisant une forme "conventionnelle"

- ARBITRAIRE : caractérisant une forme "arbitraire"

EXEMPLE :

DCL aediter_3 REPRESENTATION STRUCTURE (tab_PL_1 (*,*,2));

REMARQUE

Quelle que soit la nature de l'objet de base, il est syntaxiquement correct d'adopter n'importe laquelle des représentations dont la description peut être exprimée en L.A.M.P.E.

EXEMPLE-1 :

Un objet de base simple étant constitué d'une seule information est généralement soumis à une Représentation linéaire.
Une autre forme de représentation (conventionnelle ou non) peut également être adoptée. Ceci est syntaxiquement correct même si le résultat obtenu n'est pas esthétique.
Ainsi, une valeur simple associée au libellé qui précise sa signification, peut donner lieu a :

* Une représentation tabulée

D'où, après concrétisation, l'un des résultats suivants :

NOMBRE D'APPAREILS EXPEDIES A L'ETRANGER	12000

ou

12000	NOMBRE D'APPAREILS EXPEDIES A L'ETRANGER

ou

NOMBRE D'APPAREILS EXPEDIES A L'ETRANGER
12000

ou

12000
NOMBRE D'APPAREILS EXPEDIES A L'ETRANGER

* Une représentation structurée

D'où, après concrétisation, les résultats suivants :

NOMBRE D'APPAREILS EXPEDIES A L'ETRANGER 12000

EXEMPLE-2

Un objet de base composé, étant constitué de plusieurs informations simples, est généralement soumis à une représentation s'accordant avec son type.

Une autre forme de représentation (conventionnelle ou non) peut cependant fort bien convenir. Ainsi, tout objet "Fichier", en vertu de l'interprétation à laquelle il se prêtera, peut donner lieu :

- à une représentation linéaire : (édition de textes, par exemple)
- à une représentation tabulée : (édition de statistiques, par exemple)
- à une représentation structurée : (édition de bilans financiers, par exemple)
- à une représentation arbitraire.

2.3.5. L'ordonnancement des composants de l'objet de base

a/ L'ordonnacement des composants de l'objet de base constitue une étape, dans la description d'une Représentation, qui fait logiquement suite à celle ayant permis de définir l'aspect global de l'objet de base.

- Dans le cas d'une Représentation conventionnelle, il en est ainsi du fait que le choix de l'aspect global s'est traduit par l'adoption d'un schéma-directeur. Par suite l'ordonnancement de C_B se traduit par :
  * l'indication de la configuration-type retenue parmi celles proposées par le schéma-directeur adopté
  * la disposition des informations à éditer dans les zones B et M de la configuration-type retenue.

- Dans le cas d'une Représentation arbitraire, il en est ainsi

du fait que le choix de l'aspect global est librement défini par le programmeur. Par suite l'ordonnancement de C_B se traduit par :

* la conception d'une configuration regroupant les zones B et M.
* la disposition des informations à éditer dans les zones B et M respectives.

b/ L'ordonnancement de C_B se ramène à l'ordonnacement de sous-ensembles de C_B.

- Les tableaux PL/1, les structures PL/1 et les fichiers L.A.M.P.E. en tant qu'objets de base composés, regroupent les données en sous-ensembles. C'est ce qui explique l'existence :

* de lignes, de colonnes, de couches, etc... dans un tableau
* de niveaux dans les structures PL/1 et les fichiers L.A.M.P.E

- Les informations constitutives d'un même sous-ensemble sont indépendantes les unes des autres.

En conséquence, fixer l'ordonnancement de C_B revient à disposer, conformément à l'aspect global préalablement adopté, les sous-ensembles de C_B et non les composants eux-mêmes; ce qui permet, d'une part, de respecter les liens établis lors de la déclaration entre chaque composant de l'objet de base et, d'autre part, de n'autoriser que les ordonnancements significatifs.

c/ L'ordonnancement des sous-ensembles de C_B et l'ordonnancement des libellés correspondant fournis par l'objet Modèle associé, sont deux opérations équivalentes.

Dans le cadre d'une Représentation, une association est nécessairement établie entre les composants d'un objet Modèle et ceux de l'objet de base traité (Cf §2.2.2.2.); Dès lors, chaque niveau, figurant dans le corps de la déclaration de l'objet Modèle sélectionné, concerne indistinctement des informations propres à ce dernier et un sous-ensemble de l'objet de base.

Parmi les informations concernant un même niveau, seules sont éditables, et de ce fait destinées à être ordonnancées, les "libellés" (chaînes de caractères) et le sous-ensemble correspondant de C_B.

- Dans le cas d'une Représentation conventionnelle :
  * l'assignation des libellés de même niveau à des zones M précises du Schéma-directeur, correspond implicitement au choix d'une configuration-type.
  * L'ordonnancement des libellés dans les zones M entraine implicitement l'ordonnancement de C_B.

- Dans le cas d'une Représentation arbitraire :
  * la disposition des libellés de même niveau correspond implicitement à la définition d'une configuration.
  * l'ordonnancement des libellés entraîne implicitement l'ordonnancement de C_B.

L'ordonnancement des composants d'un objet de base et, selon le cas, le choix d'une configuration-type ou l'élaboration d'une configuration, sont deux opérations dépendantes et assumées conjointement.

Règle d'association : soumettre des sous-ensembles de composants d'un objet de base à des opérations impliquées par une Représentation, revient à soumettre à ces mêmes opérations les libellés appartenant au niveau correspondant de l'objet-Modèle associé.

Corollaire : Soumettre à des directives des sous-ensembles de composants d'un objet de base ou les libellés correspondants fournis par le Modèle associé, se ramène à adopter comme opérande l'indication des niveaux appropriés de la forme structurée décrivant l'objet-Modèle considéré.

2.3.5.1. Le mode de raisonnement dicté par l'idée-maîtresse

Tout ordonnancement de C_B est tributaire :

* du type de la Représentation envisagée
* de la nature de l'objet de base considéré

L'influence de ces critères sur le nombre d'ordonnancements possibles de C_B suffit à le prouver.

2.3.5.1.1. Le nombre d'ordonnancement de C_B est fonction du type de la Représentation.

Hypothéses: - Prenons pour objet de base un tableau PL/1, dénommé "tab_pl_1", et regroupant des valeurs arithmétiques.

* tab_pl_1 répond à la déclaration suivante :
  "DCL tab_pl_1 (4,3,2)FIXED;"
* tab_pl_1 regroupe les données selon la disposition en mémoire exprimée par la figure 2.2. 2.

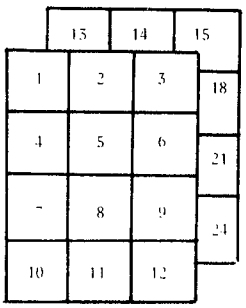


figure 2.2.2.

- Désignons :
  * par "n" le nombre des dimensions
  * par N le nombre des ordonnancements possibles des sous-ensembles de C_B conformément au schéma-directeur adopté
  * par "COU" , "COL", "LIG", les diverses catégories de sous-ensembles de C_B figurant dans l'objet de base.

Démonstration

L 'édition, est par définition, une représentation dans le plan.

A fin que les informations qu'il regroupe puissent être éditées, tout tableau de plus de deux dimensions doit subir une transformation le ramenant à un tableau bi ou uni-dimensionnel.

Ce qui en L.A.M.P.E. résulte :

* du choix de l'aspect global de la Représentation,
* de l'ordonnancement des classes de C_B.

Ainsi, en prenant toujours n > 0 nous aurons :

a/ Pour une Représentation tabulée bi-dimensionnelle

$$N = E[3-\frac{3}{n}] \times \sum_{i=E[\frac{n+1}{2}]+1}^{n+E[\frac{1}{n}]} A_n^{i-1}$$

avec $A_n^i = \frac{n!}{(n-i)!}$

et E[.]= partie entière de [

Application

Les composants de tab_pl_1 pourront être ordonnés de 12 façons différentes (voir figure 2.2.3)

b/ Pour une Représentation tabulée uni-dimensionnelle

$$N = 2 \times n!$$

Application

Les composants de "tab_pl"1" pourront être ordonnés de 12 Façons différentes (voir figure 2.2.4) et figure 2.2.5).

c/ Pour une Représentation structurée

$$N = n!$$

Application

Les composants de "tab_pl_1" pourront être ordonnés de 6 façons différentes.(Voir figure 2.2. 5)

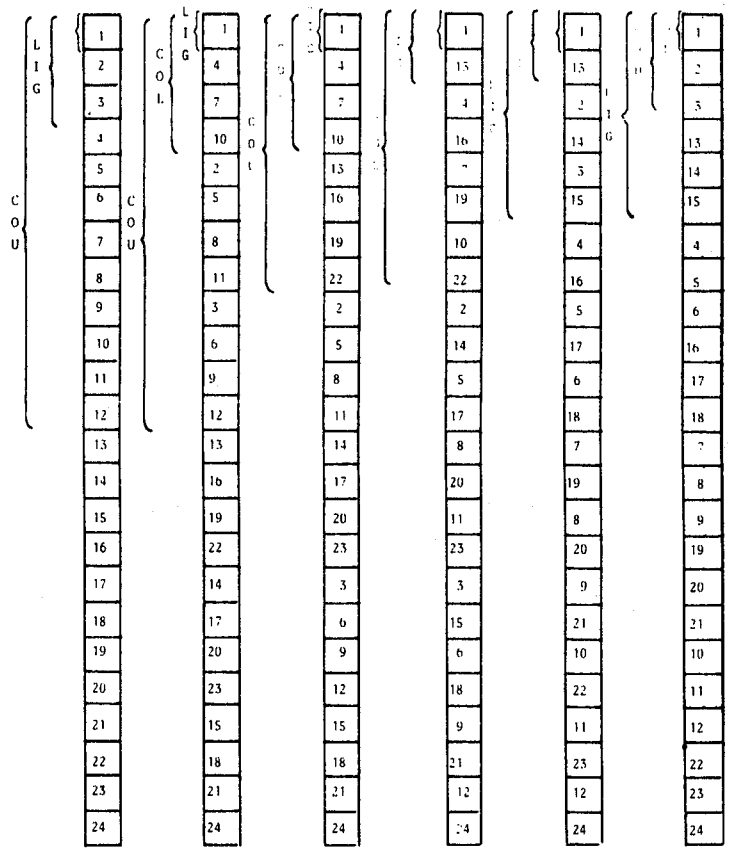


figure 2.2.4.

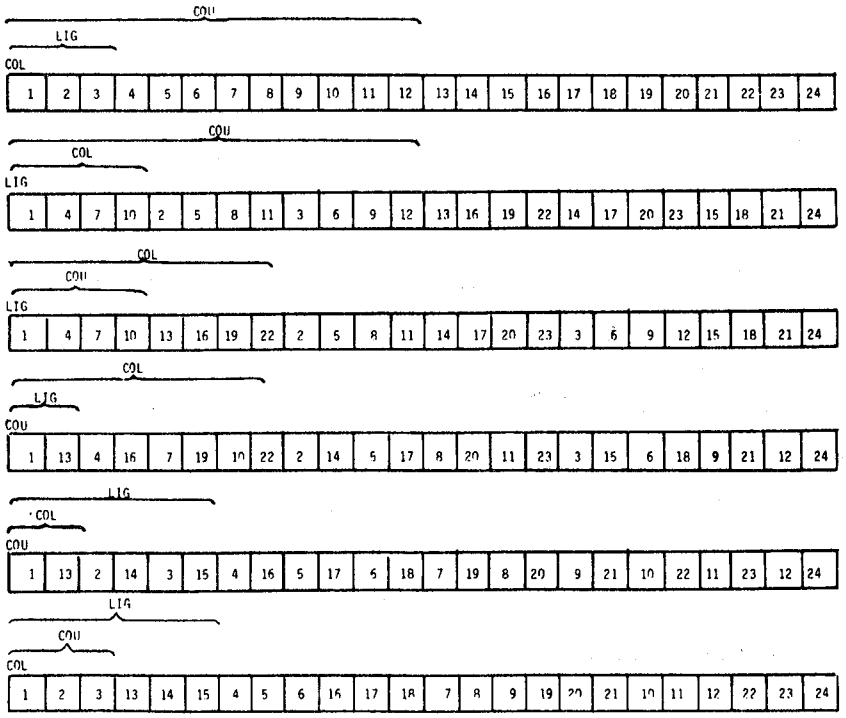


figure 2.2.5.

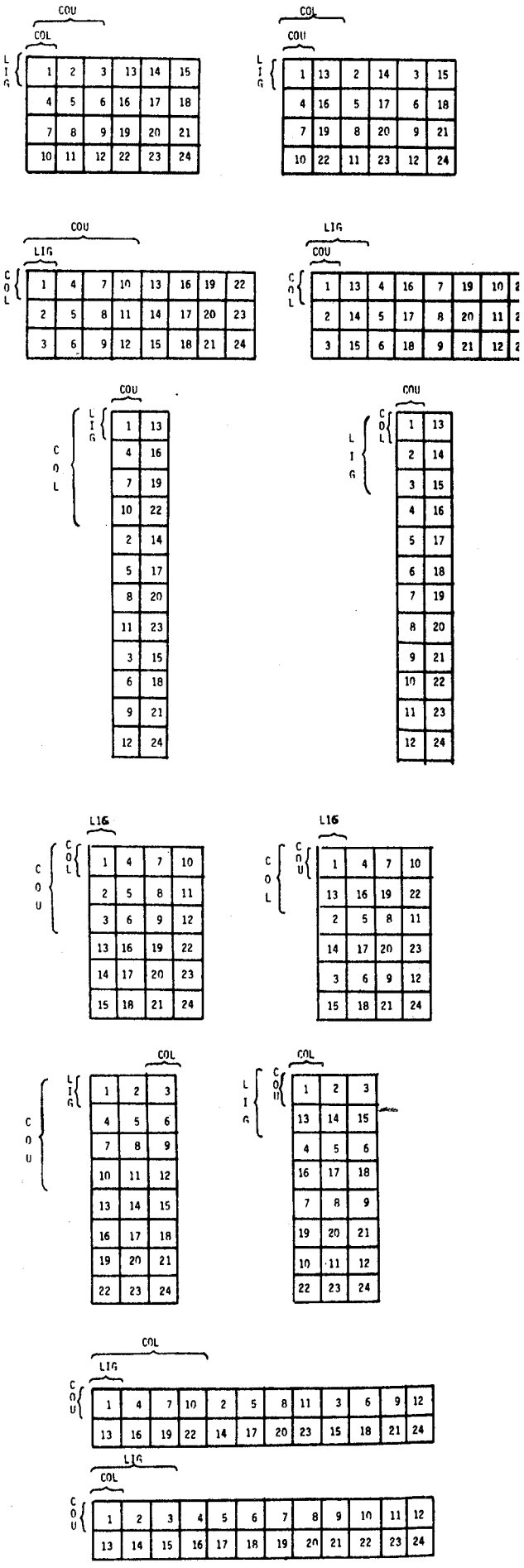


figure 2.2.3.

d/ Pour une Représentation linéaire

$$N = n!$$

Les Représentations linéaires et structurées ne se distinguent, en effet, que par l'implantation des zones B et M (Cf.§2.3.4.1.1.) et non par les ordonnancements possibles de C_B.

e/ Pour une Représentation arbitraire

Une Représentation arbitraire, ne se différenciant des Représentations conventionnelles que par le non-respect d'une configuration-type, autorise, le même nombre d'ordonnancements de C_B que l'ensemble des diverses Représentations conventionnelles

$$N = 2\,n! + E[3-\frac{3}{n}] \times \sum_{i=E[\frac{n+1}{2}]+1}^{n+E[\frac{1}{n}]} A_n^{i-1}$$

Conclusion

Le nombre d'ordonnancements possibles de C_B est fonction du type de la Représentation adoptée.

2.3.5.1.2. Le nombre d'ordonnancements de C_B est fonction de la nature de l'objet de base

Pour montrer l'influence de la nature de l'objet de base sur le nombre d'ordonnancements de C_B, il nous suffit de prouver que les formules précédemment établies fournissent une valeur numérique variant, pour un même type de Représentation, en fonction de l'objet de base considéré.

Démonstration

Pour un même type de Représentation, seul le paramètre "n" peut faire varier la valeur de N.

Ce paramètre désigne le nombre de sous-ensembles de C_B permutables.

A ce titre, nous convenons de le dénommer "indice de permutabilité" de l'objet de base. Ainsi :

a/ Pour un objet de base simple

n = 1

Tout objet de base simple possède, en effet, un seul sous-ensemble d'éléments: celui contenant l'unique information constitutive.

b/ Pour un objet Structure PL/1

n = nombre de niveaux constitutifs des structures mineures Identiques, Exclusives et Maximales.

- Structures Identiques, parce qu'elles doivent
  - * être composées d'informations de même type (regroupées essentiellement en une arborescence).
  - * répondre à la même organisation.

- Structures Exclusives, parce que leurs noeuds doivent être les composants exclusifs d'un même niveau de la structure majeure.

- Structures Maximales, parce que parmi les structures mineures identiques qui peuvent exister dans toute structure PL/1, leur noeuds doivent occuper le plus haut niveau.

EXEMPLE :

L'indice de permutabilité de l'objet suivant a pour valeur 2. En effet, les structures mineures répondant aux critères précédemment énoncés ont pour noeuds respectivement A,B,C,D,E et comportent deux niveaux.

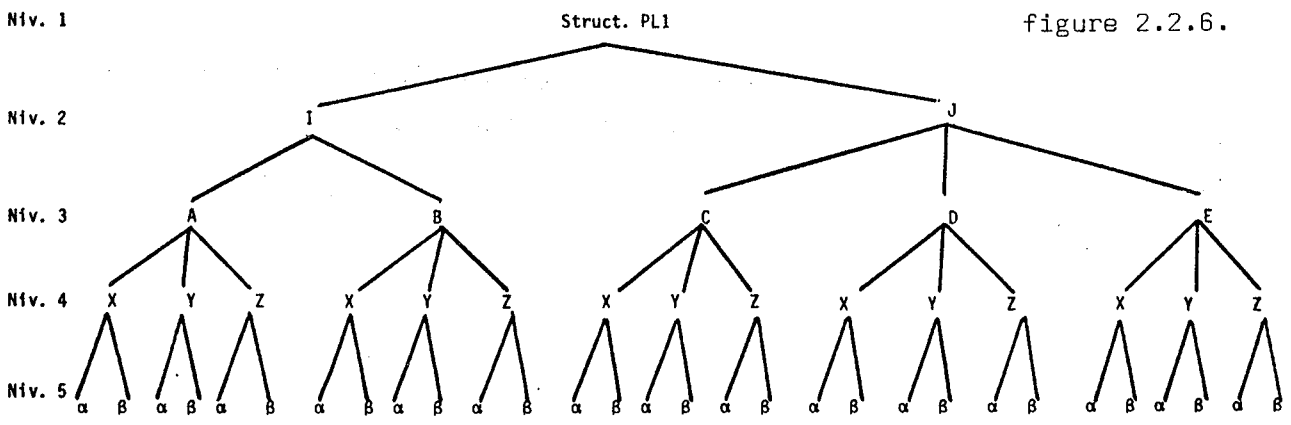


figure 2.2.6.

c/ Pour un objet Tableau

n = nombre de dimensions du tableau

d/ Pour un objet Fichier

L'objet "Fichier" s'identifie à un tableau virtuel uni-dimensionnel dont les éléments peuvent être des structures (Cf.§2.1.2.4.).

n = nombre de dimensions du tableau virtuel (l'unité) + nombre de niveaux des structures mineures Identiques, Exclusives et Maximales constituant éventuellement chaque élément du tableau virtuel.

Conclusion

Le nombre d'ordonnancements possibles de C_B est fonction de la natur de l'objet de base.

La détermination grâce aux formules précitées du nombre d'ordonnancements possibles de C_B permet de détecter les incompatibilités entre la nature de l'objet de base et le type de la Représentation envisagée.

Ainsi, il se révèle impossible d'adopter une "Représentation tabulée bi-dimensionnelle pour un objet de base simple, un objet Tableau uni-dimensionnel ou tout objet Structure dont l'indice de permutabilité est unitaire (Cf.2.3.5.1.1-a).

2.3.5.2. La façon d'exprimer en L.A.M.P.E. l'ordonnancement envisagé

L'ordonnancement de C_B est décrit dans le corps de la déclaration de l'objet Représentation et se traduit par la rédaction de requêtes (Cf.§1.4.).

Deux étapes fondamentales, dictées par l'application de la règle d'association (Cf §2.3.5.) sont à distinguer.

2.3.5.2.1. La corrélation entre l'objet de base et l'objet "Modèle" approprié

Rédaction d'une requête établissant la connection indispensable entre chacun des niveaux de l'objet "Modèle" et chaque sous-ensemble de composants de l'objet de base.

D'où :

- l'emploi de la directive : ASSOCIER,
- la désignation précise de l'objet "Modèle" par la mention de son identificateur éventuellement suivi des paramètres effectifs (Cf.§ 2.2.1.)

2.3.5.2.2. La manipulation des composants de chaque niveau d'un Modèle

Le choix d'une configuration-type et l'ordonnancement des composants de l'objet de base sont deux opérations dépendantes et assumées conjointement (Cf §2.3.5.)

- L'adoption d'une configuration-type parmi celles offertes par le schéma-directeur se ramène à désigner les zones M à utiliser (Cf §2.3.4.1.1.)

- La disposition des composants d'un objet de base, conformément à la règle association, se ramène à la manipulation des informations désormais associées à chacun des niveaux de l'objet Modèle sélectionné.

Les traitements, en vertu de la définition du langage, sont confiés à des requêtes, qui adoptent la désignation des niveaux du Modèle pour opérandes et décrivent simultanément ces deux opérations (Cf §2.3.5.-a).

En conséquence, une seule requête suffit à décrire l'implantation et l'organisation des libellés dans une zone M.

* Le choix de l'implantation est communiqué par la directive introduisant la requête.
* L'ordonnancement des libellés est décrit par les opérateurs figurant dans le corps de la requête.

a/ Dans le cas d'une Représentation conventionnelle, le nombre de requêtes est égal au nombre de zones M disponibles dans la configuration type retenue.

- Le choix de l'implantation des libellés

  * La configuration-type comprend plus d'une zone M. Un choix s'impose. D'où l'utilisation de la directive "IMPLANTER" et l'apport de précisions par la présence de l'un des indicateurs. de positions "HAUT", "BAS", "DROITE", "GAUCHE" (Cf§1.4.4.1.2.e)

  * La configuration-type comprend une seule zone M. Aucun choix ne s'impose. D'où l'utilisation de la directive "DISPOSER".

- L'ordonnancement des libellés

  Application des opérateurs de position (cf. §1.4.3.1) à la désignation des niveaux du Modèle concerné. Chaque niveau, désigné par son indicateur numérique est :

  * qualifié par l'identificateur du "Modèle"
    EX: "complément.1"

  * complété éventuellement par un modèle de données précisant la longueur, en caractères, impartie aux libellés figurant dans le niveau sélectionné.
    EX: "complément.1 A(15) puis complément.3 A(10)

b/ Dans le cas d'une Représentation arbitraire, une seule requête assure la description.

- Le choix de l'implantation des libellés

  En raison du non respect d'une configuration-type, l'organisation des informations dans le Rectangle-Unité résulte de la disposition horizontale ou verticale des libellés et des valeurs correspondantes de l'objet de base. D'où l'emploi des directives "ENUMERER" et "ALIGNER".

- L'ordonnancement des libellés

  Application des opérateurs de position à la désignation des niveaux du Modèle concerné selon les mêmes principes que dans le cas d'une Représentation conventieonnelle.

REMARQUE

La représentation d'un objet de base n'implique pas nécessairement la qualification, par des libellés, des divers sous-ensembles de données. L'objet Modèle requis par la représentation ne comportera alors pas de libellés. Néanmoins, le raisonnement précité n'en subira aucune altération. L'utilisateur doit simplement considérer l'existence de libellés virtuels.

N.B. Pour matérialiser, sur des exemples concrets, la méthode précédemment exposée, nous renvoyons le lecteur à l'annexe 2.

2.3.6. Fixer la composition et l'implantation des traits de séparation

Les séparations, constituées par un alignement de caractères vides ou non, font partie des artifices assurant la mise en valeur, en vue de l'édition (Cf 1ère partie §2.1.2.2.-a), des données constitutives de tout objet de base composé.

2.3.6.1. Le mode de raisonnement dicté par l'idée-maitresse

Parmi les diverses séparations qu'il est possible d'établir dans le cadre de la Représentation d'un objet de base composé, il est nécessaire de distinguer celles tributaires du type de Représentation adopté, de celles librement programmées par l'utilisateur.

a/ Séparations tributaires du type de Représentation

Conformément aux aspects habituellement requis, seules les Représentations tabulées nécessitent un démarquage explic ite des zones B et M; opération qui pour cette raison est systématiquement effectuée sans que le programmeur ait besoin d'intervenir (cf. § 2.3.4.1.1).

b/ Séparations tributaires des décisions de l'utilisateur

Conformément aux principes régissant l'ordonnancement de C_B, séparer C_B revient à séparer quelle que soit la Représentation adoptée, les

sous-ensembles de C_B. En conséquence, la règle d'association ramène la séparation des sous-ensembles de C_B à celle des intitulés fournis par l'objet Modèle sélectionné.

2.3.6.2. La façon de décrire, en L.A.M.P.E., les traits de séparation

Pour définir à l'aide de L.A.M.P.E., la nature et l'implantation des traits de séparation, on utilise, dans le cadre des requêtes manipulant les informations associées à chaque niveau du modèle (Cf.§2.3.5.2.2.), la directive SEPARER.

a/ L'emploi d'une restriction (Cf §1.4.4.) permet de définir les caractères constitutifs du trait de séparation.
EXEMPLE : SEPARER('*') ...

b/ L'interprétation de la directive est conditionnée par l'opérande auquel elle s'adresse. En effet:
- L'opérande peut être constitué par la référence à un niveau de l'objet Mod Les traits de séparation délimitent alors les intitulés et les composants correspondants du sous-ensemble de C_b
  EXEMPLE : " ... SEPARER complément.1 ... ; "
- L'opérande peut être constitué par une commande introduite par la directiv générale et ayant pour opérande la référence à un niveau de l'objet Modéle La directive générale, errige en un bloc d'informations, les intitulés appartenant au niveau du Modéle désigné et les sous-ensembles correspondants de C_b .Le trait sépare alors ce blocd'informations de celui précédemment mentionné dans la requéte (voir annexe 2).
  EXEMPLE : " ... complément_1 PUIS SEPARER DISPOSER complément_2 ... "

2.4. LA DECLARATION DE MODE "PROPOSITION"

Les diverses dispositions et mises en évidence impliquées par la mise-en-pages des informations constitutives d'objets éditables simples sont exprimées sous la forme d'objets éditables composés.

D'où l'existence des objets "Composition", "Implantation" et "Répartition".

Déclarer de tels objets, revient à décrire conformément aux concepts de base du langage et en fonction de la complexité de l'Edition envisagée

* soit un Rectangle-Entité
* soit un Rectangle Page-Virtuelle
* soit des Rectangles Pages-Formelles

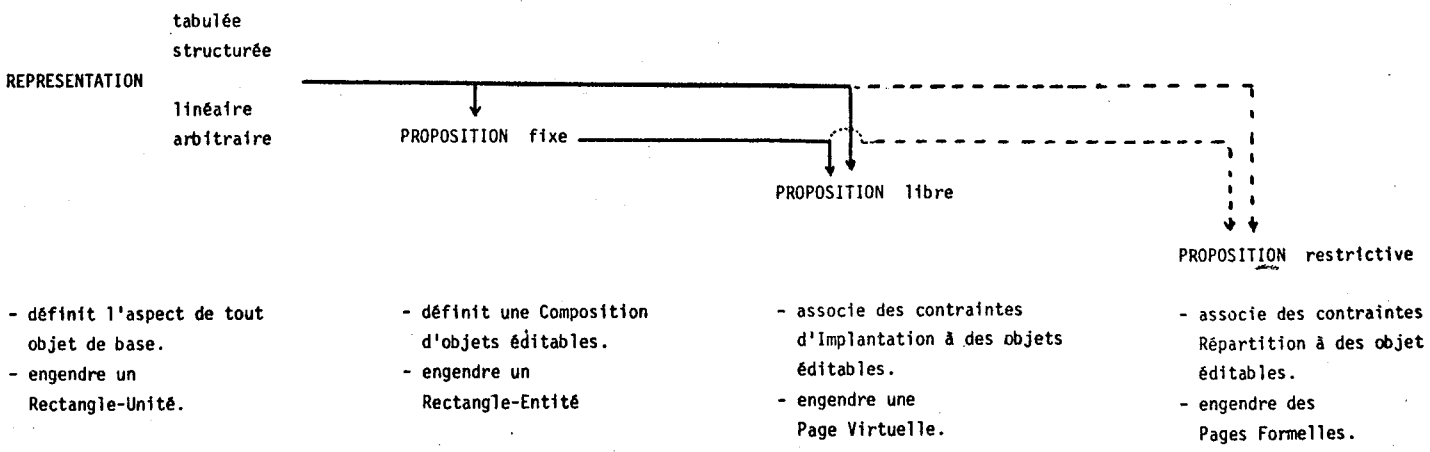
2.4.1. Les caractères spécifiques des déclarations de mode proposition

Les objets décrits par une déclaration de mode proposition sont à la fois composés et éditables.

A ce titre, ils sont paramétrables (Cf.§1.2.5.2.2.); ce qui autorise la communication dynamique de leurs éléments constitutifs, en l'occurence des objets éditables.

2.4.2. La méthode employée

La méthode employée a été exposée lors de la description du langage (Ch.II,première partie) et se trouve résumée dans le schéma bien connu suivant qui met en évidence les différentes phases du raisonnement.



Les objets éditables composés sont construits à partir d'objets éditables de complexité inférieure ou égale à la leur.

Ceci nous amène à préciser la relation existant entre la complexité d'un tel objet et sa constitution.

La complexité d'un objet éditable composé caractérise la phase du raisonnement décrite en L.A.M.P.E.(Composition,Implantation,Répartition). Ce qui se traduit par l'occurrence, dans l'en-tête de la déclaration, de l'un des attributs suivants :

- FIXE (étape Composition)
- LIBRE (étape Implantation)
- RESTRICTIF (étape Répartition)

La constitution d'un objet éditable composé définit les critères de coexistence des objets éditables utilisés ou, plus précisément, la disposition et la mise en évidence des informations que ces derniers rassemblent.

Une telle organisation se traduit, en L.AM.P.E., par des requêtes constituant le corps de la déclaration; ce qui est conforme à la propriété des objets élaborés d'être éditables.

En outre, elle doit être en accord avec l'attribut figurant dans l'en-tête de la déclaration puisque, selon le cas, elle fixe la configuration d'un "Rectangle-entité", d'un "rectangle-Page-Virtuelle" ou de "rectangles-Pages-Formelles".

2.4.2.1. L'étape "Composition"

L'étape Composition précise la mise en valeur et la disposition (fixe) des objets éditables utilisés.

Ceux-ci devant être de complexité inférieure ou égale à celle de l'objet Composition, ne peuvent être que des objets Représentation et Composition.

Le corps de la déclaration se compose ainsi d'une **seule** requête faisant appel :

- à la directive générale
- aux directives de mise-en-valeur (intrinsèques et par contexte)
- aux opérateurs de position.

EXEMPLE GENERAL :

```
DCL entite PROPOSITION FIXE ;
    DELIMITER INSERER DISPOSER
                          SOULIGNER A PUIS C FIN
                      FIN ET B ;
FIN ;
```

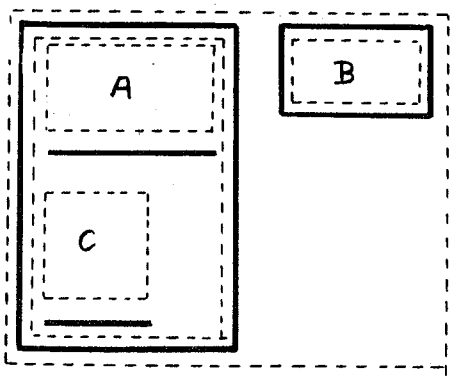


figure 2.2.7.

REMARQUE :

On aurait pu obtenir le même résultat que précédemment en rédigeant les déclarations suivantes :

DCL X PROPOSITION FIXE ; SOULIGNER A PUIS C ; FIN ;
DCL entite PROPOSITION FIXE ; INSERER DELIMITER X ET B ; FIN ;

Particularité

En vertu des concepts du langage et des principes régissant le fonctionnement du système "M.P.E.", le nombre d'objets éditables constituant un objet Composition n'est pas limité.

2.4.2.2. L'étape Implantation

L'étape Implantation définit la mise en évidence ou précise l'ordre d'implantation des informations constitutives des objets éditables utilisés, en fonction de l'importance que le programmeur accorde, de par leur signification, à ces derniers.

Toute liberté est ainsi laissée au système M.P.E., d'ordonnancer au mieux, au sein d'un "Rectangle-Page-Virtuelle" les informations constitutives des objets éditables.

Les objets éditables utilisés sont des objets "Représentation" et Composition; ceux-là mêmes auxquels il est logique de faire subir une Implantation.

Le corps de la déclaration se compose de requêtes rangées en fonction de l'importance attribuée aux objets auxquels chacune d'elles s'adresse.

Ces requêtes font appel indifféremment :
- à la directive de mise en évidence (IMPLANTER)
- à la directive générale (DISPOSER).

Particularités

a/ En vertu du principe régissant, dans le cadre d'une Implantation, la disposition des informations à mettre en évidence (Cf première partie 2.1.2.2.

- Les requêtes faisant appel à la directive de "Mise-en-évidence" (IMPLANTE sont au plus au nombre de deux et figurent en tête du corps de la déclaration puisque s'adressant implicitement à des objets dotés d'une importance particulière dans l'Edition envisagée.

- La "restriction", qui accompagne nécessairement la directive "IMPLANTER", peut fixer la position retenue (HAUT, BAS, DROITE, GAUCHE) ou laisser au système M.P.E. le soin de choisir, au sein du "Rectangle-Page-Virtuelle", celle convenant le mieux.("PRIORITE)

b/ Le corps de la déclaration se compose de plusieurs requêtes
Leur nombre n'est pas limité.

REMARQUE

En vertu des simplifications de description autorisées par L.A.M.P.E toute déclaration décrivant un objet "Composition" peut être évitée dans la mesure où cet objet entre dans la constitution d'un autre objet élaboré. (Cf.§1.2.6.3.).

En conséquence, indépendamment des directives générales et de mise en évidence, les requêtes peuvent faire appel :
- aux directives utilisées dans la déclaration d'un objet composition,
- aux opérateurs de mise en relation.

EXEMPLE :

Les deux séquences suivantes sont identiques.

```
DCL aediter PROPOSITION LIBRE;
    IMPLANTER (PRIORITE) DISPOSER SOULIGNER objet_a FIN PUIS
                                          objet_b ET objet_c;
    DISPOSER DELIMITER INSERER tableau ; FIN aediter;

DCL entité_1 PROPOSITION FIXE ;
    DELIMITER INSERER tableau ; FIN ;
DCL entité_2 PROPOSITION FIXE;
    DISPOSER SOULIGNER objet_a FIN PUIS objet_b ET objet_c ; FIN;
DCL aediter PROPOSITION LIBRE ;
    IMPLANTER (PRIORITE) entité_2 ; DISPOSER entité_1 ; FIN ;
```

2.4.2.3. L'étape "Répartition"

L'étape Répartition permet de désigner, parmi les objets soumis à une "Implantation", ceux qui doivent respecter une consigne restrictive; en l'occurrence, leur coexistence sur une même page lors de l'Edition.

Elle traduit le raisonnement suivant :

Etant donné un ensemble d'informations ayant fait l'objet de la Représentation de Composition et participant à une "Implantation", vouloir que certaines d'entre elles, que l'on désigne explicitement, puissent figurer sur une même page de "listing" quelles que soient les dimensions adoptées pour cette dernière.

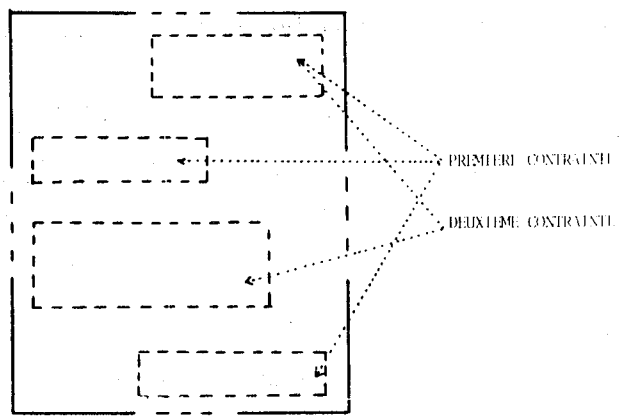


figure 2.2.8.

Le corps de la déclaration se compose d'un nombre de requêtes égal au nombre de restrictions envisagées.

Ces requêtes font appel :

- à la directive générale,
- à l'opérateur de liaison ("AVEC"), puisque chacune d'elles s'adresse nécessairement à plus d'un objet dont les positions respectives ont été précisées au cours de l'étape précédente.

Particularités

- En ce qui concerne le corps de la déclaration
  * le nombre de requêtes n'est pas limité (autant que de contraintes)
  * les objets traités ne peuvent en aucun cas être un nombre supérieur à celui des objets participant à l'étape précédente (Implantation).

- En ce qui concerne l'en-tête de la déclaration
  * Une restriction mentionne l'identificateur de l'objet Implantation.
  * Si l'objet Implantation concerné est paramètré, l'objet Répartition l'est nécessairement.

  Par contre, l'indication des paramètres dans la restriction est facultative puisqu'elle indique un objet de référence et ne traduit pas une utilisation au "sens L.A.M.P.E." du terme.

EXEMPLE :

Les deux séquences suivantes sont identiques.

```
DCL précédent (alpha, beta, gamma) PROP LIBR ; ...;
DCL courant (alpha, beta, gamma) PROP REST (precedent) ;...;
```

```
DCL precedent (alpha, beta, gamma) PROP LIBR ; ... ;
DCL courant (alpha, beta, gamma) PROP REST (precedent (alpha,
                                            beta, gamma)); ...;
```

REMARQUE

Conformément aux simplifications de description autorisées par L.A.M.P.E., (Cf §1.2.6.3.), toute déclaration décrivant un objet Implantation peut être évitée dans la mesure où tous les composants sont astreints à figurer sur une même page et par suite à participer à l'élaboration d'un objet Répartition.

Conséquences

a/ Le mode figurant dans l'en-tête de la déclaration n'implique plus l'adjonc d'une restriction fixant l'objet Implantation concerné.

b/ Indépendamment de la directive générale et de l'opérateur de liaison, les requêtes peuvent faire appel :
- à toutes les directives utilisées dans la déclaration de mode PROPOSITION-FIXE.
- Aux opérateurs de position.

EXEMPLE :

Les deux séquences suivantes sont identiques.

```
DCL cefini PROPOSITION RESTRICTIF ;
    DISPOSER IMPLANTER (PRIORITE) objet_a FIN AVEC objet_b AVEC
                                                  objet_c ; FIN ;
```

```
DCL etape_1 PROPOSITION LIBRE;
    IMPLANTER (PRIORITE) objet_a;
    DISPOSER objet_b ; DISPOSER objet_c ; FIN ;
DCL cefini PROPOSITION RESTRICTIF (etape_1)
    DISPOSER objet_a AVEC objet_b AVEC objet_c ; FIN ;
```

2.5. LES INSTRUCTIONS ET FONCTIONS INCORPOREES

La concrétisation conditionnelle des objets implique l'utilisation des instructions et fonctions incorporées L.A.M.P.E.

Leur emploi dans un programme est régi par les mêmes règles que celui des instructions et fonctions incorporées PL/1.

Aucune précision n'est, de ce fait, à apporter sur cette ultime phas de tout algorithme de Mise-en-Pages et d'Edition.

Nous renvoyons le lecteur, aux deux exemples mentionnés en Annexe-2 et annexe-3 de façon qu'il puisse avoir un aperçu de l'aspect global d'un traitement L.AM.P.E.

T R O I S I E M E P A R T I E

<><><><><><><>

L'IMPLEMENTATION DU LANGAGE

TROISIEME PARTIE

C H A P I T R E I

CONTROLE DES DIVERSES PHASES DU TRAITEMENT D'UN PROGRAMME PL/1-L.A.M.P.E.

La séquence d'opérations à laquelle est soumis un programme PL/1-L.A.M.P.E. peut se résumer de la façon suivante :

Le programme est en un premier temps soumis à un Pré-processeur chargé de substituer aux déclarations et instructions L.A.M.P.E. des séquences de déclarations et d'instructions PL/1; le pré-processeur détecte, en outre, les erreurs du texte L.A.M.P.E. et procède, si possible, à leur correction.

Le module "pré-traité" est alors confié au compilateur PL/1 qui fournit un module-objet.

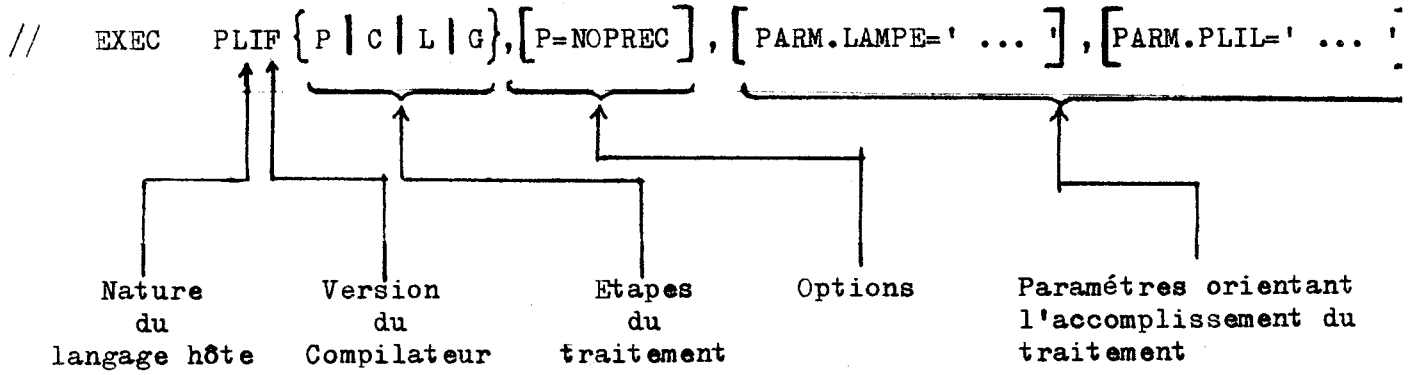
L'édition de liens établit la corrélation entre le module objet et les composants des bibliothèques PL/1 et L.A.M.P.E. (procédures-outils du système M.P.E.).

Finalement, le module chargeable est exécuté.

Toutes ces phases du traitement sont appelées par des commandes au "Système" d'exploitation [IB-3-6] dont nous allons détailler les fonctions et les propriétés.

I.1. LA MISE EN OEUVRE DES PHASES DU TRAITEMENT(DU POINT DE VUE DE L'UTILISATEUR)

Par compatibilité avec la méthode d'appel des compilateurs des divers langages évolués disponibles, tout programme PL/1-L.A.M.P.E. est introduit par une carte "EXEC" dont le format est ci-après explicité.



I.1.1. Les divers indicateurs représentatifs des étapes du traitement

L'indicateur "P" commande le PRE-TRAITEMENT
L'indicateur "C" commande la COMPILATION du texte PL/1 (original et général)
L'indicateur "L" commande la MISE-EN-OEUVRE de l'EDITEUR DE LIENS
L'indicateur "G" commande l'EXECUTION.

Ainsi, étant donné un programme PL/1 - L.A.M.P.E., il peut être souhaitable de tester la validité des instructions L.A.M.P.E. avant de procéder à la compilation PL/1.

D'où les deux séquences :

```
//        EXEC    PL1FP
//SYSIN   DD      *
          .
          .
          .
/*

//        EXEC    PL1FPC
//SYSIN   DD      *
          .
          .
          .
/*
```

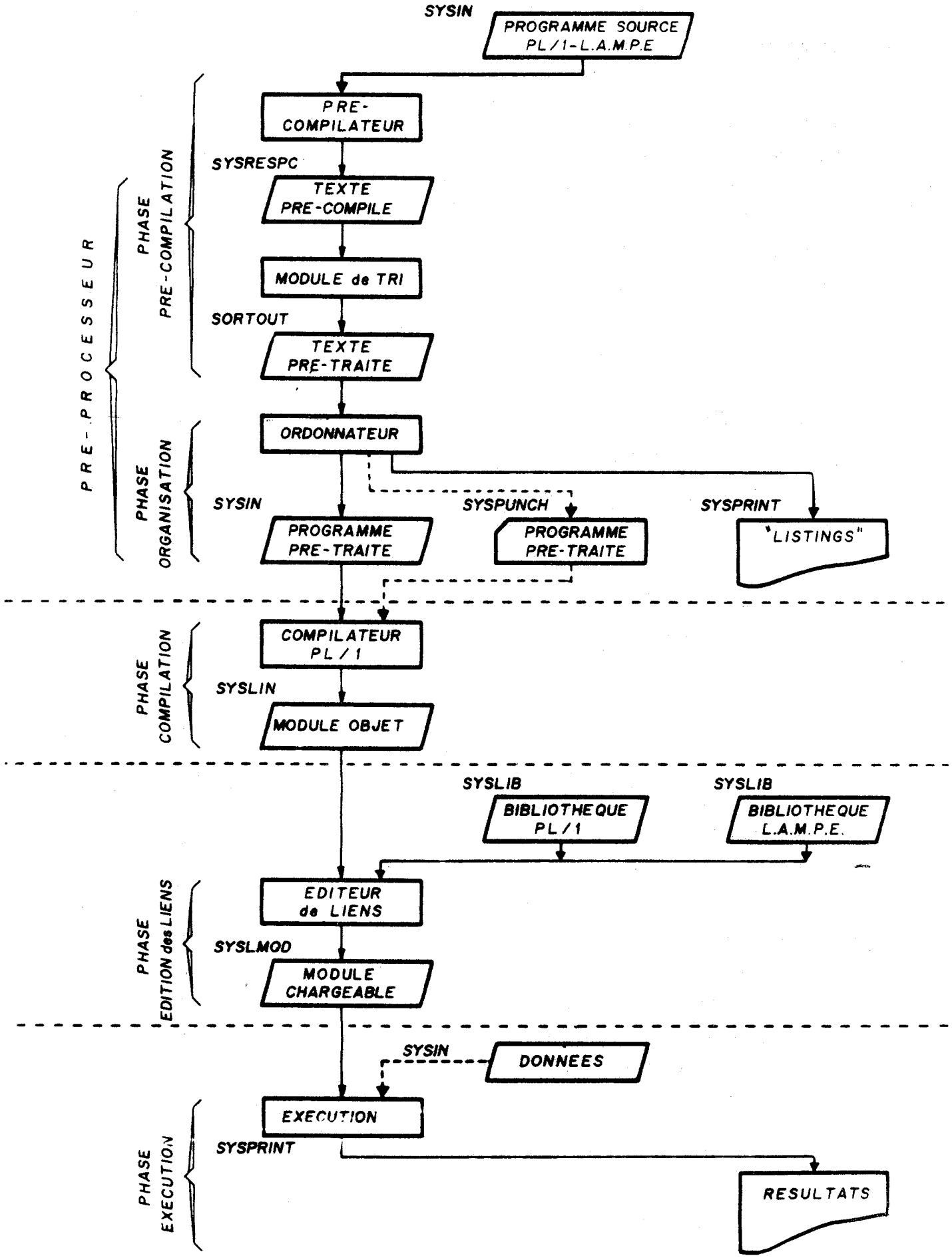

SYSIN
PROGRAMME SOURCE PL/1-L.A.M.P.E
PRE-COMPILATEUR
SYSRESPC
TEXTE PRE-COMPILE
MODULE de TRI
SORTOUT
TEXTE PRE-TRAITE
ORDONNATEUR
SYSIN
SYSPUNCH
SYSPRINT
PROGRAMME PRE-TRAITE
PROGRAMME PRE-TRAITE
"LISTINGS"
COMPILATEUR PL/1
SYSLIN
MODULE OBJET
SYSLIB
BIBLIOTHEQUE PL/1
SYSLIB
BIBLIOTHEQUE L.A.M.P.E.
EDITEUR de LIENS
SYSLMOD
MODULE CHARGEABLE
SYSIN
DONNEES
EXECUTION
SYSPRINT
RESULTATS
PRE-PROCESSEUR
PHASE PRE-COMPILATION
PHASE ORGANISATION
PHASE COMPILATION
PHASE EDITION des LIENS
PHASE EXECUTION

figure 3.1.1.

1.1.2. L'option

La compilation du texte L.A.M.P.E. et le tri du texte généré, d'une part, la diversification des éléments constitutifs du texte généré, en fonction du rôle qui leur est imparti, d'autre part, représentent les deux phases du pré-traitement (Cf organigramme p.3).

Afin de permettre la compilation du seul texte PL/1, en cours de mise-au-point, la compilation du texte L.A.M.P.E. peut être supprimée par l'option "P=NOPREC".

1.1.3. Les paramètres

Introduits par le mot-clé "PARM", les paramètres s'adressent soit au compilateur PL/1, soit au pré-processeur L.A.M.P.E. D'où leur qualification.

1.1.3.1. Les paramètres s'adressant au compilateur PL/1

Il n'est pas nécessaire de préciser leurs objectifs puisqu'ils sont déjà familiers à tout utilisateur du langage PL/1 [IB-3-2]

1.1.3.2. Les paramètres s'adressant au pré-processeur L.A.M.P.E.

a/ Les paramètres permettant de modifier la nature des listes délivrées par le pré-processeur :

- La mention du paramètre "LIST-SOURCE" permet d'obtenir une liste du programme source.
- La mention du paramètre "LIST-OBJET" permet d'obtenir une liste du programme produit à l'issue du pré-traitement. Le texte PL/1 est transcrit tel quel, mais le texte L.A.M.P.E. est signalé à l'attention du programmeur et suivi éventuellement des diagnostics d'erreurs, puis du texte PL/1 généré.

```
PROG : PROCEDURE OPTIONS(MAIN);
       DCL (ALPHA, BETA, GAMMA) FIXED(15);
         .
         .
         .
       DO I=1 TO N: TAB(I)= I*I/2; END;

-->L.A.M.P.E.: PUT LAMPE (AEDITER) (CARAC =130, LIGNE=40)
-->DIAGNOSTIC: ABSENCE ';' FINAL.
               CORRECTION EFFECTUEE.
-->GENERATION: <Génération de l'expansion de l'instruction.>

       IF <>20 GOTO TRAIT_SUIV ;
         .
         .
         .
END  PROG :
```

- La mention du paramètre "LIST-ATTEST" permet d'obtenir une liste réduite du texte pré-traité. Elle se résume à l'édition des instructions L.A.M.P.E, de leur position dans le programme source, éventuellement des diagnostics d'erreurs et du texte généré.

```
-->L.A.M.P.E.: PUT LAMPE (CARACT=130,LIGNE=40);
-->POSITION  : 93
-->DIAGNOSTIC: ABSENCE IDENTIFICATEUR DE L'OBJET.
               ERREUR FATALE : GENERATION ANNULEE.

    .
    .
/*
```

b/ Les paramètres s'appliquant au texte soumis au compilateur PL/1.

- La mention du paramètre "TRACE" met en commentaire les instructions L.A.M.P.E. au sein du programme pré-traité.
- La mention du paramètre "DECK" permet d'obtenir un paquet de cartes correspondant au programme pré-traité.

REMARQUES

1- La communication des paramètres au pré-compilateur est régie par les règles habituelles [IB-3-2]

2- En l'absence de paramètres explicitement indiqués, l'adoption de certains d'entre eux est réalisée, par défaut, en fonction des étapes déclenchées. Ainsi,il est convenu que ;

- PLIFP{C|L|G}	implique LIST_SOURCE , LIST_ATTEST
- PLIFP	implique LIST_SOURCE , LIST_OBJET
- PLIFP{C|L|G},P=NOPREC	implique LIST_SOURCE , TRACE
- PLIFP,P=NOPREC	implique LIST_SOURCE , LIST-OBJET , TRACE

1.1.4. Les définitions de données

L'accomplissement des diverses étapes est tributaire de la définition des données communiquées et de celles de leurs supports. Les données peuvent,

- Soit conditionner le fonctionnement du compilateur et du pré-compilateur
  Elles figurent généralement dans des "fichiers de manoeuvre",et les cartes "DD" correspondantes sont incluses dans les procédures cataloguées (Cf §1.2.)

- Soit dépendre du traitement traduit par le programme
  Ce sont les divers fichiers utilisés,dont le programmeur doit assurer la description explicite à la suite de l'instruction "EXEC PLIFP...".

- Soit être impliquées par le déroulement de la plupart des programmes
  Leur utilisation est, alors, suffisamment fréquente pour que la rédaction des cartes "DD" en soit simplifiée.
  EXEMPLE : Sortie sur imprimante : //GO.SYSPRINT DD SYSOUT=A
  Lecture de cartes perforées : //SYSIN DD *
  :
  /*

1.2. LES PROCEDURES CATALOGUEES

Il existe quatre procédures cataloguées

"PCLG" "PCL" "PC" "P"

A chacune de ces combinaisons correspond, dans la "PROCLIB", une expansion comportant deux parties (voir Annexe 5)..

- La mise en oeuvre du pré-compilateur, du tri et de l'Ordonnateur,
- La mise en oeuvre du Compilateur PL/1, de l'Editeur de liens et du Chargeur, puis de la phase Exécution.

TROISIEME PARTIE

C H A P I T R E II

LE PRE-TRAITEMENT

Nous entendons par pré-traitement, l'ensemble des opérations assumées par le pré-processeur et conduisant à la transformation de tout programme-source PL/1-L.A.M.P.E., en un programme "pré-traité" qui satisfasse aux critères imposés par le compilateur PL/1.

Le pré-processeur comporte un ensemble de "modules".

- Les uns sont regroupés en une unité de traitement, dénommée pré-compilateur, qui assure trois fonctions :

  * Analyse lexicographique du programme-source,
  * Analyse syntaxique du texte L.A.M.P.E.
  * Génération des expansions PL/1; en l'occurrence, élaboration de déclarations et de séquences d'instructions régissant le fonctionnement du système M.P.E.

- Les autres sont indépendants. Dénommés "Tri" et "Ordonnateur", ils assurent, respectivement, la classification et la répartition des composants du texte préalablement traité.

Nous nous bornerons, dans le présent chapitre, à décrire au travers des fonctions essentielles du pré-traitement, les aspects marquants de ces modules.

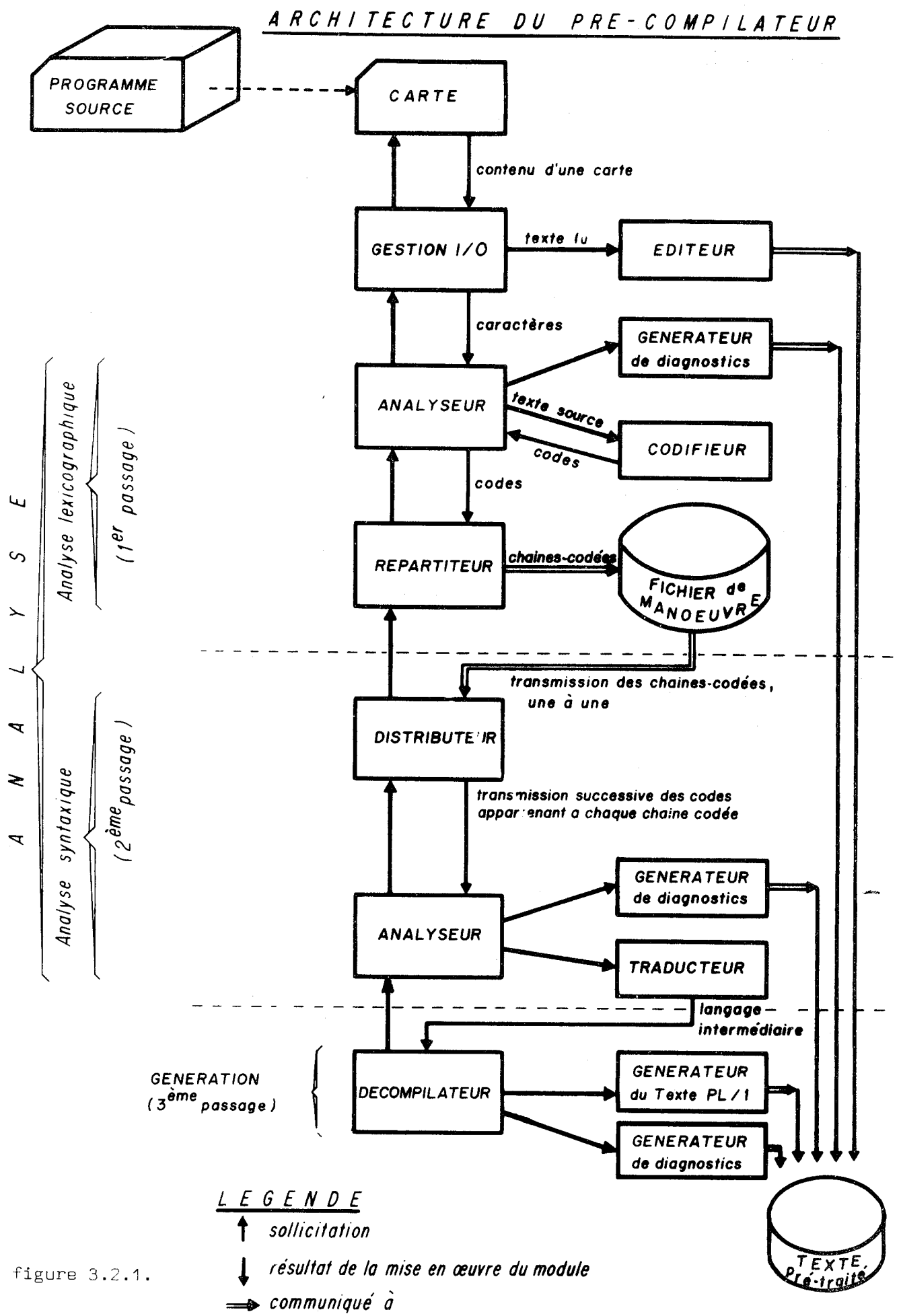

ARCHITECTURE DU PRE-COMPILATEUR
PROGRAMME SOURCE
CARTE
contenu d'une carte
GESTION I/O
texte lu
EDITEUR
caractères
GENERATEUR de diagnostics
ANALYSEUR
texte source
codes
CODIFIEUR
codes
REPARTITEUR
chaines-codées
FICHIER de MANOEUVRE
transmission des chaines-codées, une à une
DISTRIBUTEUR
transmission successive des codes appartenant a chaque chaine codée
GENERATEUR de diagnostics
ANALYSEUR
TRADUCTEUR
langage intermédiaire
GENERATION (3ème passage)
DECOMPILATEUR
GENERATEUR du Texte PL/1
GENERATEUR de diagnostics
TEXTE pré-traité
ANALYSE
Analyse lexicographique (1er passage)
Analyse syntaxique (2ème passage)
LEGENDE
sollicitation
résultat de la mise en œuvre du module
communiqué à

figure 3.2.1.

2.1. L'ANALYSE LEXICOGRAPHIQUE

L'Analyse Lexicographique, admettant pour donnée le programme-source qu'elle considère comme une suite bornée de caractères, accomplit dans notre cas des tâches plus délicates que celles qui lui sont habituellement dévolues.

Celà pour plusieurs raisons :

a/ L'Analyse Lexicographique est réalisée dans le cadre d'un pré-compilateur; A ce titre, bien qu'elle ne doive traiter que le texte L.A.M.P.E., c'est l'intégralité du texte-source qui lui est soumis.

b/ L'Analyse Lexicographique s'adresse à un programme-source homogène (Cf.première partie §2.2.1.3.).

* Instructions et déclarations L.A.M.P.E. sont introduites par les mêmes mots-clés que leurs homologues PL/1.
* Les mots-clés L.A.M.P.E. apparaissant dans toute instruction ou déclaration ne sont pas différenciés par l'occurrence d'un préfixe.

c/ L'Analyse Lexicographique est confrontée à des facilités d'écriture offertes par PL/1 et reprises par L.A.M.P.E.:

* Les commentaires sont considérés comme des délimiteurs.
* La perforation d'un programme est totalement libre.

Il incombe donc à l'Analyse Lexicographique :

* de déterminer les Unités syntaxiques (tout caractère ou groupement de caractères susceptibles d'avoir une valeur syntaxique).
* de codifier chaque unité syntaxique.
* de distinguer leur appartenance à PL/1 ou à L.A.M.P.E. et de les soumettre, dans ce dernier cas, à l'Analyse Syntaxique.

Ces tâches sont confiées à des modules spécialisés; en l'occurrence, respectivement: l'ANALYSEUR, le CODIFIEUR et le REPARTITEUR (voir architecture du pré-compilateur).

2.1.1. Le module Analyseur

Sollicité, dans le cadre de l'Analyse Lexicographique par le module Répartiteur, le module Analyseur a pour rôle :

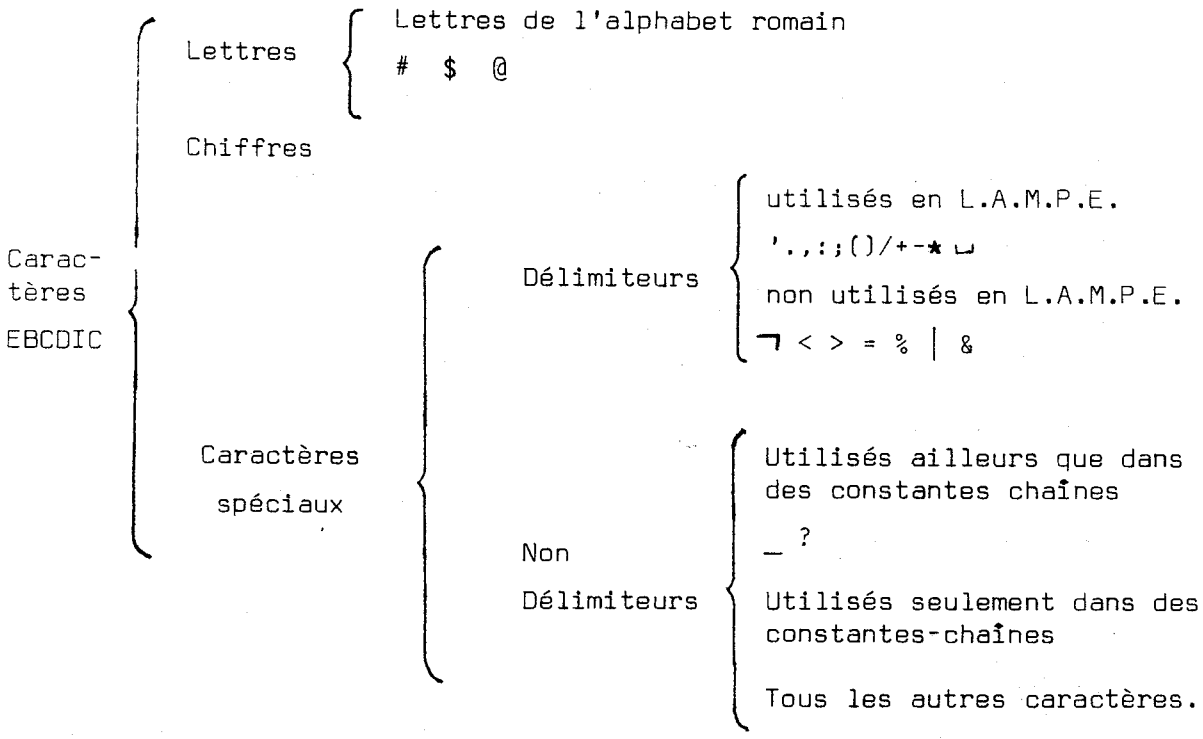
* de déceler les unités-syntaxiques,
* de reconnaître les unités syntaxiques,
* de retourner pour chaque unité syntaxique, après activation du module codifieur, le code la caractérisant (ou pseudo-code).

2.1.1.1. L'ensemble des caractères pris en considération

Les composants de tout texte PL/1-L.A.M.P.E. se répartissent :

- en Mots-clés,
- en Identificateurs,
- en Littéraux { Constante entière, ; Constante Chaîne,
- en Commentaires,
- en Caractères spéciaux jouant le rôle de délimiteurs.

Ces divers éléments sont constitués d'un ou de plusieurs caractères, EBCDIC, dont nous donnons, ci-après, une structuration appropriée au contexte PL/1-L.A.M.P.E.



Dans notre cas, l'Analyseur, ne sachant de prime abord si l'unité syntaxique courante appartient à L.A.M.P.E. ou à PL/1, reconnaît la gamme de caractères communs aux deux langages.

2.1.1.2. La Sélection des Unités Syntaxiques

La sélection d'une Unité Syntaxique est basée sur la détection des délimiteurs.

Sont considérées comme unités syntaxiques :

- Toutes chaînes de caractères autres que des commentaires,
  * Constantes chaînes de caractères (concaténation de caractères délimitée par des quotes.
  * Chaînes de caractères représentatives d'Identificateurs, de Mots-clés et de constantes entières.

- Tous caractères spéciaux délimiteurs autres que les blancs ou que les caractères bornant commentaires et constantes-chaînes.

2.1.1.3. La reconnaissance des Unités Syntaxiques

La composition de toute Unité Syntaxique présente dans un programme PL/1-L.A.M.P.E. est régie par l'une des règles suivantes :

TEXTE_PL1_LAMPE → {DELIMITEUR | NON_DELIMITEUR}*

NON_DELIMITEUR → IDENTIFICATEUR | LITTERAL

DELIMITEUR → ␣ | CARACTERE_SPECIAL_DELIMITEUR | COMMENTAIRE

COMMENTAIRE → /* { caractères à l'exception de : * suivi de/ }* */

IDENTIFICATEUR, LITTERAL → (voir grammaire en annexe)

CARACTERE_SPECIAL_DELIMITEUR → (voir: §2.1.1.1.)

REMARQUES

- Les lettres, les chiffres, les symboles ('et /) sont les pivots des processus de reconnaissance des Unités Syntaxiques.
- La reconnaissance des Unités Syntaxiques permet au module Répartiteur de déceler l'appartenance de l'instruction ou de la déclaration courante à L.A.M.P.E. ou à PL/1 (Cf §2.1.3.1.). A cet effet, nous convenons de qualifier de 'significatives' (pour le pré-compilateur) les Unités Syntaxiques appartenant au texte L.A.M.P.E.
- Les Unités Syntaxiques significatives sont seules soumises, après stockage éventuel en mémoire (Cf §2.1.1.4.a. et b) puis codification (Cf.§2.1.2.), à l'Analyse Syntaxique.

Conséquences

A toutes Unités Syntaxiques sont associées, sous la dénomination de "Caractéristiques", trois précisions : NATURE, CLASSE et INDICATEURS SEMANTIQUES

- La "Nature" précise l'appartenance à l'une des catégories :
  * Caractère
  * Identificateur
  * Littéral

- La "classe" et les "Indicateurs sémantiques" sont dépendantes de la "Nature". Elles peuvent être soit connues dès l'Analyse Lexicographique si elles s'appliquent à des ensembles d'Unités Syntaxiques exhaustifs (Mots-clés), soit déterminées au cours de l'Analyse Syntaxique, dans le cas contraire (Identificateurs littéraux).

2.1.1.4. L'organisation des données en mémoire

L'utilisation de tables est nécessairement requise pour stocker et fournir les caractéristiques associées à chaque Unité-Syntaxique significative .

a/ Le Dictionnaire

Il contient :

* l'ensemble des mots-clés L.A.M.P.E., certains mots-clés PL/1 considérés comme des pivots de l'Analyse Syntaxique (Cf.Annexe 6) et les abréviations de tous ces mots-clés.
* les identificateurs et littéraux (de plus d'un caractère) rencontrés dans le texte L.A.M.P.E. et stockés au cours de l'Analyse Lexicographique.

b/ La table de référence

Elle regroupe les caractéristiques associées aux éléments rangés dans le dictionnaire.

c/ La table des poids

Elle constitue une table d'équivalence comportant les valeurs numériques (ou poids) attribuées à chacun des 256 caractères EBCDIC de façon à préciser leurs propriétés.

Classification			Poids
Chiffres (10)			1
Lettres (29)			2
Caractères spéciaux	Délimiteurs	Utilisés en L.A.M.P.E.(12)	3
		Non utilisés en L.A.M.P.E.(7)	4
	Non Délimiteurs	Utilisés ailleurs que dans des constantes-chaînes (2)	5
		Utilisés seulement dans des constantes-chaînes (194)	6

Chaque élément de la table est désigné par la valeur numérique du caractère correspondant.

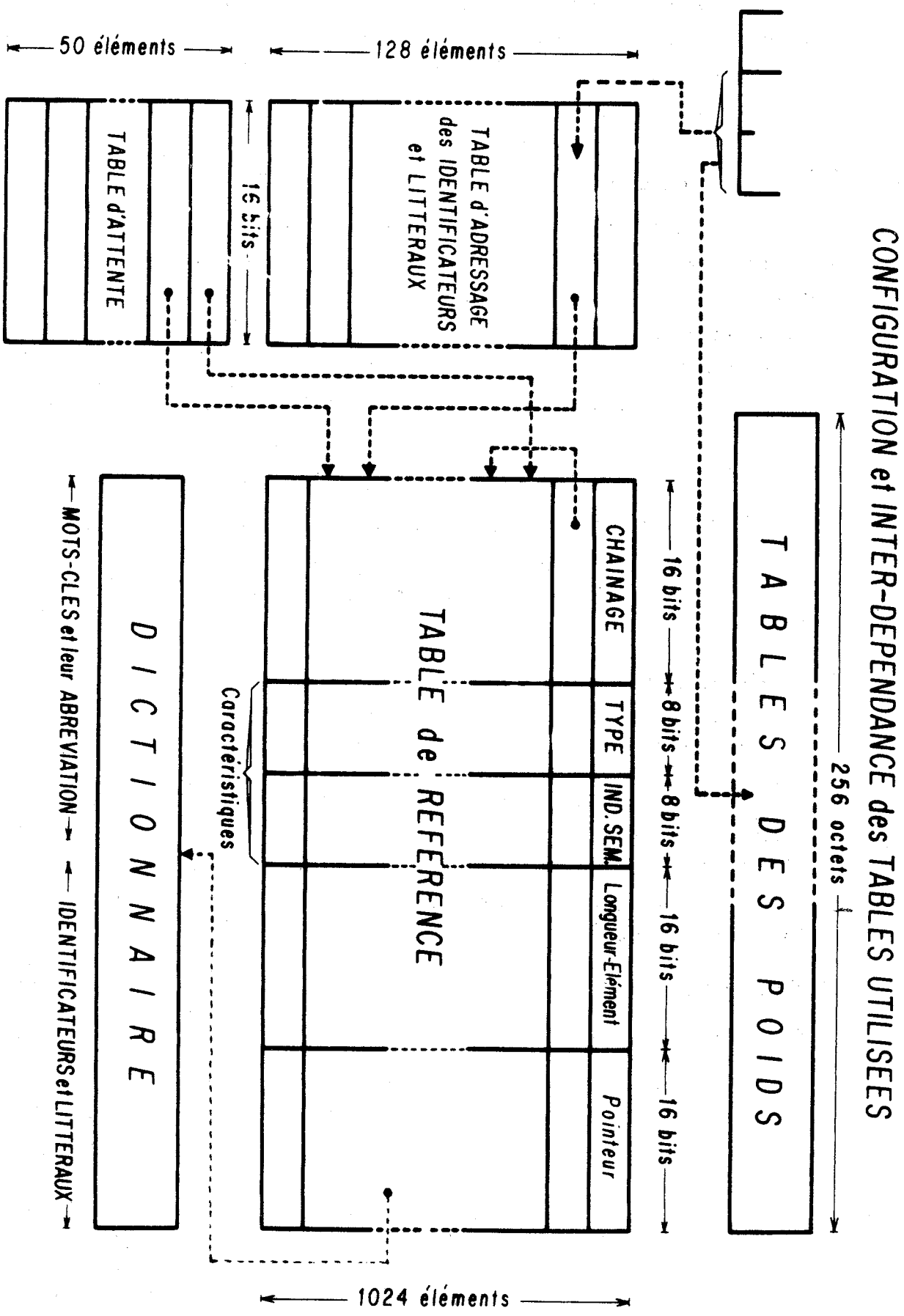

CONFIGURATION et INTER-DEPENDANCE des TABLES UTILISEES
TABLES DES POIDS
256 octets
16 bits
8 bits
8 bits
16 bits
16 bits
TABLE de REFERENCE
CHAINAGE
TYPE
IND. SEM.
Longueur-Elément
Pointeur
Caractéristiques
1024 éléments
DICTIONNAIRE
MOTS-CLES et leur ABREVIATION
IDENTIFICATEURS et LITTERAUX
50 éléments
128 éléments
TABLE d'ATTENTE
16 bits
TABLE d'ADRESSAGE des IDENTIFICATEURS et LITTERAUX

figure 3.2.2.

d/ La table d'adressage

Elle regroupe des pointeurs vers l'un des 1024 éléments de la table de référence. Chaque élément de la table d'adressage est désigné par la valeur du Hash-code de l'Unité Syntaxique sélectionnée, à condition qu'elle comporte plus d'un caractère (voir § 2.1.1.4.a).

e/ La table d'attente

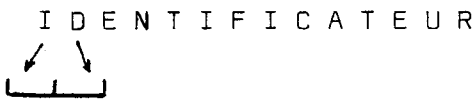
Elle regroupe, de même que la table d'adressage, des pointeurs vers la table de référence. Elle désigne les premières unités syntaxiques de l'instruction ou de la déclaration courante qui, par prudence, ont été rangées dans le "dictionnaire" alors qu'on ne pourrait savoir, à ce stade du traitement, si elles étaient "significatives".

2.1.1.5. La méthode de recherche et de stockage des Unités Syntaxiques significatives.

a/ Le Hash-code (adressage dispersé)

Le Hash-code adopté est celui utilisé dans le compilateur PL/1(F) [IB-3-4]. Chaque Unité syntaxique de plus d'un caractère est décomposée en bloc de deux caractères. A chaque bloc correspond une valeur numérique (code des caractères).

EXEMPLE :



Les valeurs numériques de chaque couple de caractères sont ajoutées les unes aux autres modulo 127.

b/ Collision et litige

Il y a collision, lorsque deux Unités syntaxiques ont même Hash-code (quel que soit le nombre de caractères qu'elle comporte).

Il y a litige, lorsque deux unités syntaxiques ont même Hash-code et même nombre de caractères.

EXEMPLE : L I M I T E et M I L I T E

c/ Principe de stockage

La consultation du dictionnaire n'est nécessaire qu'en cas de litige (voir Figure 3.2.3.)

N.B. Aucun mot-clé n'étant réservé, chacun d'eux peut être utilisé comme littéral ou comme identificateur de variable.

- Dans le premier cas, il n'y a aucune ambiguité. Sa fonction est déterminée dès l'Analyse lexicographique, compte-tenu de la codification particulière d'un littéral chaîne de caractères (placé entre deux quotes). Par suite, un élément distinct, pointant vers le texte du mot-clé figurant dans le dictionnaire, est généré dans la table de Référence.

- Dans le second cas, le module Analyseur ne peut reconnaître l'utilisation du mot-clé comme identificateur de variable.
  Cette nuance importante dans l'emploi du texte d'un mot-clé ne peut être décelée :
  * de façon systématique, que lors de l'Analyse Syntaxique,
  * à titre exceptionnel, dès l'analyse lexicographique.

  Cette dernière possibilité n'est requise que par la recherche des critères permettant de décider de l'appartenance de l'instruction ou de la déclaration courante, à PL/1 ou à L.A.M.P.E. Cette tâche est assumée par le module Répartiteur (Cf.§2.1.3.).

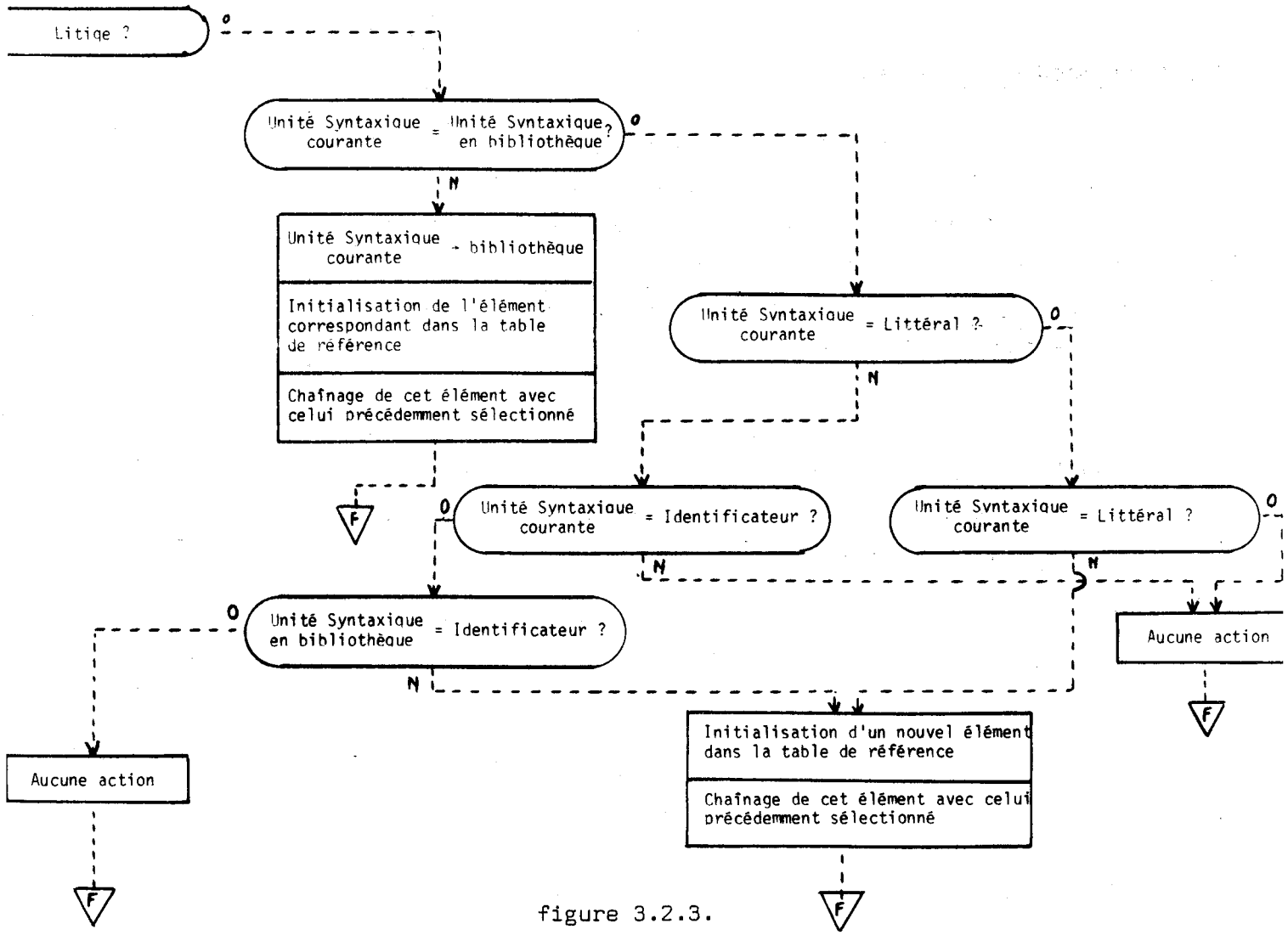


figure 3.2.3.

2.1.2. Le module codifieur

Sollicité par le module Analyseur, le module codifieur a la charge de renvoyer pour chaque Unité-Syntaxique communiquée, le pseudo-code correspondant.

Le pseudo-code occupe trois octets :

* le premier précise la classe de l'Unité Syntaxique [Annexe 6]
* les deux autres contiennent un pointeur vers l'élément correspondant de la table de Référence.

Ainsi, lors de l'Analyse Syntaxique,

* l'indication de la classe permet de s'assurer immédiatement, en fonction du contexte, de la validité de l'Unité-Syntaxique.
* le pointeur permet d'acquérir tous renseignements complémentaires nécessaires à la poursuite de la pré-compilation.

2.1.3. Le module Répartiteur

Le module Répartiteur assume le traitement-clé de l'Analyse Lexicographique. Il a, en effet, pour rôle de sélectionner, de reconnaître (ce qui lui vaut d'être assimilé à un mini-analyseur syntaxique) et de codifier chacune des instructions et déclarations L.A.M.P.E. présentes dans le programme-source. A ces fins, il a recours aux modules spécialisés Analyseur et Codifieur dont il déclenche, directement ou indirectement, le fonctionnement.

Le module Répartiteur, dont l'existence est motivée du fait que l'Analyse lexicographique à laquelle il participe, fait partie intégrante d'un pré-traitement, a une influence bénéfique sur la compilation de L.A.M.P.E.

Il permet, en effet :

* de réduire la durée des trois passages (seul le texte L.A.M.P.E. est codifié).
* d'alléger l'Analyse Syntaxique (seules les instructions et déclarations L.A.M.P.E. sont traitées).
* de prendre en compte, sans difficulté, les options que l'utilisateur peut mentionner dans la carte "EXEC" (cf.§1.1.).

N.B. A l'image de son langage hôte, L.A.M.P.E. offre des facilités d'écriture qui, pour certaines compliquent la compilation. Il en est ainsi, par exemple, de la possibilité de traiter des objets dans une séquence du programme qui précède la déclaration de l'objet considéré.

Les vérifications sémantiques, la corrélation à établir entre déclarations et instructions concernant un même objet (Cf.§2.3.2.) impliquent que le texte L.A.M.P.E. , présent dans le programme-source, soit codé dans son intégralité dès le premier passage.

A cet effet,

* le Répartiteur est mis en oeuvre une seule fois et termine son traitement à l'issue du premier passage.
* les chaînes codées, images du texte L.A.M.P.E. sont rangées en mémoire annexe (Cf.§2.1.3.4.) avant de servir de données à l'Analyse Syntaxique.

2.1.3.1. La sélection d'une instruction ou d'une déclaration

Sélectionner une instruction ou une déclaration revient à détecter son origine (première Unité Syntaxique faisant suite au délimiteur de l'instruction ou de la déclaration précédente) et son extrêmité (le délimiteur, en l'occurrence le ";").

Il s'ensuit que le point virgule représente le pivot du processus de sélection. Si la détection de l'origine ne pose aucune difficulté, il n'en est pas de même pour l'extrêmité. En effet, à la différence de PL/1, L.A.M.P.E. n'utilise pas le ";" aux seules fins de délimiter instructions et déclarations. En conséquence, la sélection, dans le programme-source, d'une instruction ou d'une déclaration est tributaire de leur reconnaissance; ce qui implique que la reconnaissance précède la recherche de l'extrêmité.

2.1.3.2. La reconnaissance d'une instruction ou d'une déclaration

Reconnaître une instruction ou une déclaration, revient à déterminer si elle est susceptible d'appartenir à PL/1 ou à L.A.M.P.E. Ce qui est d'autant plus délicat que :

- Les instructions et les déclarations L.A.M.P.E. sont au même niveau que leurs homologues PL/1.
- Les fonctions L.A.M.P.E. sont appelées comme les fonctions PL/1.

Sachant que les instructions et déclarations à soumettre à la précompilation sont introduites par des mots-clés empruntés à PL/1 (respectivement : PUT, IF, DECLARE (ou DCL)) :

a/ Décider de l'appartenance d'une instruction PUT, à PL/1 ou à L.A.M.P.E. revient à chercher l'occurrence du mot-clé "L.A.M.P.E.".

b/ Etudier l'instruction IF, revient à rechercher à la suite du mot-clé THEN la présence d'instructions L.A.M.P.E.

Ainsi se trouvent levées nombre de difficultés [GT-3-1], et l'analyse se traduit schématiquement de la façon suivante :

IF... { ; → instruction à soumettre au compilateur PL/1
mot-clé "THEN" { PUT ... → est-ce une instruction L.A.M.P.E.? (Cf.a)
Autre U.S. → recherche d'une instruction ou d'une déclaration L.A.M.P.E.

c/ Décider de l'appartenance d'une déclaration à PL/1 ou à L.A.M.P.E. revient à se conformer au schéma suivant :

DCL { identificateur { mode L.A.M.P.E. → Déclaration L.A.M.P.E.
"(...")" { mode L.A.M.P.E. → déclaration L.A.M.P.E.
; → à soumettre au compilateur PL/1
autre U.S. → à soumettre au compilateur PL/1
autre unité syntaxique → à soumettre au compilateur PL/1.

N.B. Cette technique tient compte de deux difficultés :

* l'occurrence, toujours possible, d'erreurs syntaxiques dans la rédaction d'un programme.
* l'utilisation tolérée des mots-clés comme identificateurs de variables.

2.1.3.3. La recherche de l'extrêmité d'une instruction

La condition nécessaire, mais non suffisante (Cf §.2.1.3.1.) pour que soit reconnue l'extrêmité d'une instruction ou d'une déclaration est de rechercher le point virgule final.

L'algorithme peut se résumer ainsi :

Si " ; " { ∈ déclaration L.A.M.P.E. { est suivi { Cst entière, Directive, "FIN" } Ce n'est pas l'extrêmité de la déclaration
sinon, c'est l'extrêmité
∉ déclaration L.A.M.P.E. C'est l'extrêmité d'une instruction.

REMARQUES

Une instruction ou une déclaration ne pouvant être reconnue, de prime abord, comme appartenant à PL/1 ou à L.A.M.P.E.:

- la codification et le stockage éventuel des Unités Syntaxiques significatives sont systématiquement entrepris. Ils sont poursuivis dès que l'appartenance à L.A.M.P.E. est reconnue. Dans le cas contraire, ils sont abandonnés et simultanément, dictionnaire, table de Référence et d'Adressage sont restaurés à leur situation antérieure grâce aux renseignements fournis par la table d'Attente.

- L'appartenance à PL/1 ou à L.A.M.P.E. ne peut être mentionnée sur le support du texte pré-traité (Cf.§2.4) en raison de contraintes physiques (le stockage sur disque a souvent lieu avant l'identification de l'instruction).

EXEMPLE :

```
        MENSUELLEMENT AU CLIENT DE L'ETABLISSEMENT */ PROPOSITION RESTRICTIVE.

DCL  FACTURE/* CET OBJET EST REPRESENTATIF DU MODELE DE FACTURE A ADRESSER.
```

Une table dite de "Sélection" renferme, au terme du pré-traitement, le numéro d'ordre des instructions L.A.M.P.E. incluses dans le programme-source.

2.1.3.4. Le stockage des chaînes codées

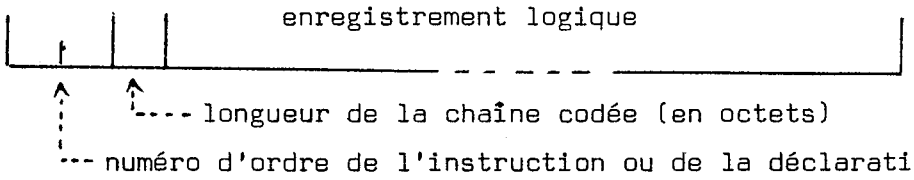
Après codification de l'instruction ou de la déclaration L.A.M.P.E. courante, le Répartiteur range la chaîne codée dans un fichier temporaire. Toute chaîne codée donne lieu, selon sa dimension, à un ou plusieurs enregistrements logiques.

a/ Chaque enregistrement logique est identifié par deux paramètres :

- Le numéro de l'instruction ou de la déclaration traitée; ce qui permet de préciser, à l'issue de l'Analyse Syntaxique, à quelle instruction ou déclaration L.A.M.P.E. se rapportent les diagnostics et le texte PL/1 générés. Il est également possible, grâce à ce paramètre, de savoir si l'enregistrement courant comporte ou non la suite de la chaîne codée figurant dans l'enregistrement précédent.

- Le nombre d'octets effectivement occupés par la chaîne codée.

b/ Chaque enregistrement physique contient un seul enregistrement logique et est identifié par une "clé" permettant d'y avoir accès directement.



2.1.3.5. La préparation à l'extraction des chaînes codées

Afin que chaque chaîne codée puisse être extraite isolément dans la suite du traitement, cinq indications sont conservées en mémoire centrale dans la "Table d'Extraction".

* La clé du premier des enregistrements physiques supportant la chaîne codée.
* Le nombre d'enregistrements physiques successifs requis en fonction de la longueur de la chaîne codée.
* L'indication précisant si la chaîne codée transcrit une instruction ou une déclaration L.A.M.P.E.
* Le pseudo-code de l'identificateur de l'objet, soit déclaré s'il s'agit d'une déclaration, soit traité s'il s'agit d'une instruction.
* Le chaînage vers l'élément de la table concernant l'occurrence d'un identificateur identique.

2.2. L'ANALYSE SYNTAXIQUE

Au terme de l'Analyse Lexicographique, le texte L.A.M.P.E. est dissocié du texte PL/1, regroupé et codifié.

Il appartient au pré-compilateur, au cours de l'Analyse Syntaxique, de veiller à la conformité syntaxique du seul texte L.A.M.P.E. Le compilateur PL/1 s'assurant, quant à lui, du respect des règles générales régissant la

rédaction de tout programme (portée des déclarations, opportunité d'un bloc, etc...).

L'Analyse Syntaxique s'acquitte de trois tâches essentielles :

- L'extraction à partir du texte L.A.M.P.E., tel qu'il se présente à l'issue de l'Analyse Lexicographique, des chaînes codées représentatives de chaque instruction et de chaque déclaration.

- L'Analyse de chaque chaîne codée sélectionnée.

- La traduction de chaque chaîne codée en un "langage intermédiaire" plus concis que le langage L.A.M.P.E. et plus propice tant aux multiples vérifications de conformité qu'à la génération du texte PL/1 équivalent.

Chacune de ces opérations est confiée à un module spécialisé; respectivement: le DISTRIBUTEUR, l'ANALYSEUR et le TRADUCTEUR (voir architecture du pré-compilateur).

2.2.1. Le module Distributeur

Mis en oeuvre par le module Analyseur, le Distributeur sélectionne, dans le fichier temporaire initialisé lors de l'Analyse Lexicographique, les chaînes codées successives et en distribue, un à un, les pseudo-codes constitutifs.

Indépendamment de ce fonctionnement très classique, le principe de sélection est dicté par la méthode d'implémentation imposée en raison de certaines facilités d'écriture offertes par L.A.M.P.E.

En effet, les objets "propres",

* peuvent être construits à partir d'autres objets L.A.M.P.E.
* peuvent être, selon leur type, paramétrés ou non.

De façon à ce que toutes les vérifications successives puissent être effectuées, une instruction et les déclarations de l'ensemble des objets dont elle déclenche, directement ou non, la concrétisation (Cf.première partie §.2.2.2.1.) doivent être traitées simultanément.

A cet effet, le Distributeur soumet au module Analyseur,

* initialement, la chaîne codée représentative d'une instruction.
* successivement, en fonction de la progression de l'analyse, les chaînes codées représentatives de l'objet à concrétiser et de ceux participant à la concrétisation, dans la mesure où ils n'ont pas déjà été traités.

2.2.2. Le module Analyseur

L'analyse est confiée à un algorithme directement déduit des règles syntaxiques du langage L.A.M.P.E.

Ces règles, en effet, une fois mises sous forme "LL(I)" sont traduites par un transformateur de grammaire en un programme PL/1.

Ce dernier assure l'analyse descendante des chaînes codées obtenues à l'issue de l'analyse lexicographique, et gérées par le DISTRIBUTEUR.

Il ne nous appartient pas,ici, d'expliciter davantage la méthode ainsi adoptée et nous renvoyons le lecteur, pour plus amples renseignements, aux ouvrages spécialisés : [GR-3-1][KN-3-1].

Néanmoins, il nous semble intéressant de fournir quelques indications directement liées à notre travail. Elles concernent, d'une part, la transformation de la grammaire L.A.M.P.E. (Cf Annexe 4) en une grammaire LL(I) et, d'autre part, les raisons qui nous ont incité à adopter PL/1 pour exprimer l'algorithme d'analyse.

a/ La mise sous forme LL(1) des règles de la grammaire L.A.M.P.E.

Elle est aisée, exception faite de celle relative aux "commandes".
Pour celle-ci la raison de la difficulté est claire. La possibilité de ne pas mentionner autant de "délimiteurs" que la commande comporte de "directives" (Cf.§ 1.4.5. 2e partie) n'est pas conforme aux quatre lois que doit respecter toute règle susceptible d'être mise sous forme "LL(I) :[KN-3-1]
Ce problème classique, puisque déjà rencontré en PL/1 (boucles DØ et blocs imbriqués) a été facilement résolu en confiant l'interprétation de cette dérogation à une routine sémantique.

b/ L'utilisation de PL/1 pour assurer l'implémentation

L.A.M.P.E. est un langage expérimental promis à diverses évolutions de façon à s'adapter aux possibilités nouvelles qui seront offertes par le système M.P.E.

Seul un langage de haut niveau peut autoriser aisément l'évolution de l'implémentation et assurer, présentement, une application rapide autant qu'efficace des théories proposées.

Plusieurs arguments nous ont fait choisir PL/1 comme langage d'implémentation :

* ses aptitudes au traitement de listes [BE-3-1]
* la variété des procédures d'entrée-sortie
* la diversité des fonctions standards
* la souplesse d'utilisation dûe aux allocations dynamiques
* la possibilité d'extraire, en cours d'exécution du programme, tout renseignement sur les variables traitées ("Dopes Vectors").

2.2.3. Le module Traducteur et les routines sémantiques

Le module Traducteur, en fonction des indications qui lui sont progressivement fournies par l'Analyseur, exprime dans le "langage intermédiaire" le traitement décrit en L.A.M.P.E.

Chaque étape de la traduction est confiée à des procédures dénommées "Routines Sémantiques".

Compte tenu des fonctions très différentes que peuvent assumer les routines sémantiques, il nous a semblé opportun de distinguer :

* celles qui complètent l'Analyse Syntaxique et en ponctuent les étapes.
* celles qui concrétisent l'Analyse Syntaxique et réalisent la traduction.

Ainsi, bien qu'elles soient toutes mises en oeuvre sur l'initiative de l'algorithme d'analyse,

* les premières sont considérées comme faisant partie intégrante de l'Analyseur.
* les secondes sont symboliquement regroupées en un module dénommé Traducteur.

a/ Le rôle des routines sémantiques complétant l'Analyse Syntaxique est le suivant :

- Effectuer si possible, en cas d'erreur syntaxique, la correction qui s'impose et délivrer le message approprié en agissant sur le générateur de diagnostics.
- Décider, en fonction du contexte, chaque fois que son utilisation est ambigüe, si un identificateur doit être considéré comme un mot-clé ou comme la désignation d'un objet.
- Déterminer, par comptage, à quelle directive se rapporte un délimiteur dans le cas de commandes imbriquées.

b/ Les particularités des routines sémantiques constitutives du Traducteur sont liées à la nature et à l'utilisation du "Langage intermédiaire" que nous décrivons dans le paragraphe suivant.

2.2.4. Le "Langage intermédiaire"

Le langage intermédiaire est utilisé par les routines sémantiques du Traducteur. Il permet de transformer le texte L.A.M.P.E. en deux listes de "cellules":

* une liste de cellules représentatives de l'instruction courante.
* une liste de cellules représentatives des objets impliqués dans la concrétisation demandée.

2.2.4.1. Les cellules constitutives du langage intermédiaire

Nous désignons par cellules, des structures d'informations destinées à exprimer sous forme condensée, les divers traitements décrits dans un texte L.A.M.P.E. Le langage intermédiaire dispose d'une gamme limitée de cellules qui se répartissent de la façon suivante :

a/ Les cellules concernant les objets

* cellule-donnée
* cellule-description de données

b/ Les cellules concernant le traitement

* cellule-instruction
* cellule-déclaration
* cellule-opérande
* cellule-directive

c/ Les cellules de corrélation

* cellule-lien
* cellule-noeud

2.2.4.2. Le rôle et la constitution des diverses cellules

a/ La cellule donnée contient l'ensemble des propriétés d'un objet.

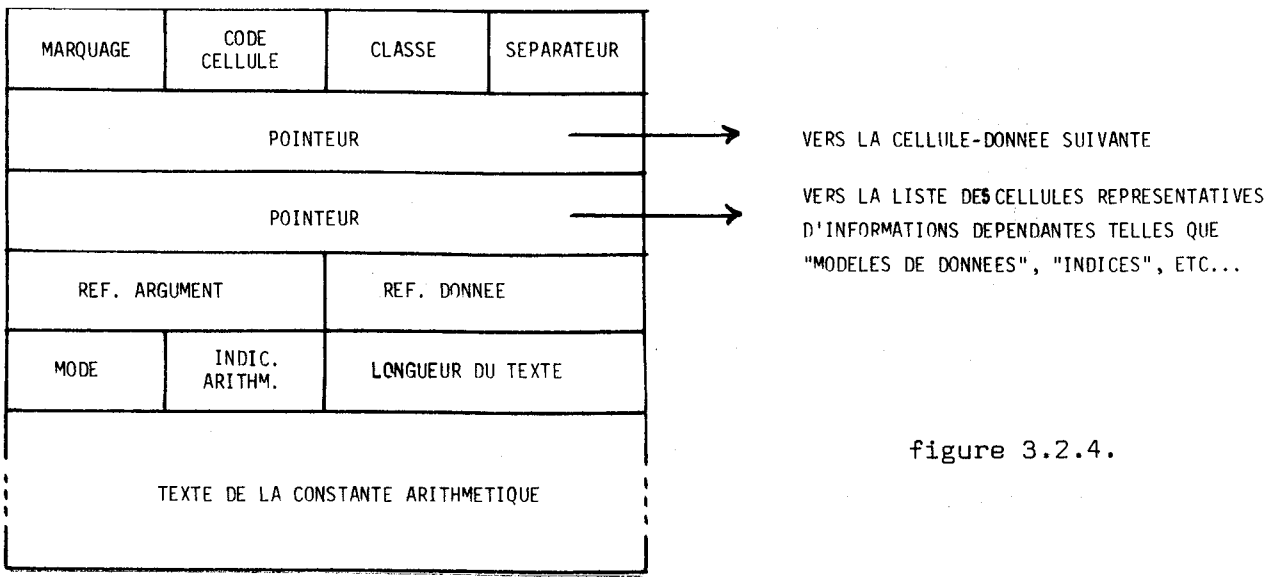


figure 3.2.4.

- La classe (voir annexe 6) précise le type de la donnée :
  * soit un objet (propre ou extérieur) désigné par son identificateur
  * soit une constante chaîne de caractères (littéral L.A.M.P.E.)
  * soit une constante arithmétique (littéral L.A.M.P.E.)

- Le séparateur précise la nature du lien existant entre deux données
  Seuls sont pris en considération : "."","":""(".
  * "," ":" et "(" sépare des données indépendantes
  * "." ":" et "(" séparent des données dépendantes

- "Ref-Argument" désigne dans la Table de Référence les indications relatives à l'objet passé en argument, si la cellule courante représente un paramètre formel mentionné dans la déclaration d'un objet L.A.M.P.E. (Cf Manuel de Référence §1.2.5.2.2.a)

-"Ref-Donnée" désigne, dans la Table de Référence, les indications relatives à l'objet traduit par la cellule courante.

- Le mode (voir annexe 6) exprime le type de l'objet (PROPOSITION, REPRESENTATION, MODELE, FICHIER, EXTERIEUR).

- L'indicateur arithmétique précise les propriétés de toute constante arithmétique de façon à déterminer, lors de l'expansion, le modèle de donnée PL/1 à générer.
  La codification de cet indicateur est déduite de la formule suivante :

  [R|I] V [D|B] V [FI|FL]

  Sachant que :

R (pour Réel)	admet pour code	"00100000"
D (pour Décimal)	" "	"00010000"
FI(pour Fixe)	" "	"00001000"
I (pour Imaginaire)	"	"00000100"
B (pour Binaire)	" "	"00000010"
FL(pour Flottant)	" "	"00000001"

- Le texte de la constante arithmétique si telle est la donnée représentée par cette cellule.

Application : Traduction dans le langage intermédiaire d'une variable indicée.
"structure.tableau (-2:+15,4:N)"

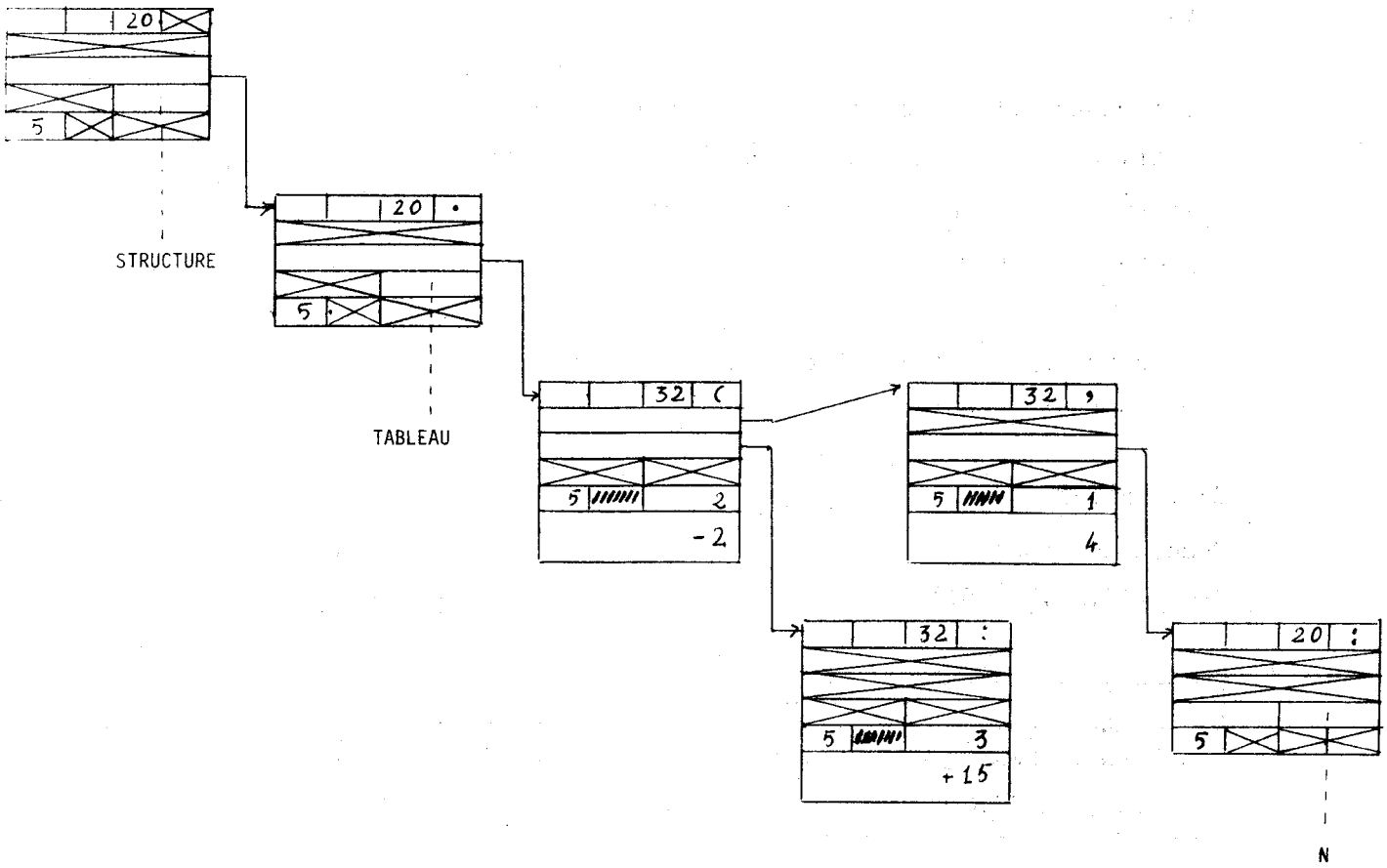


figure 3.2.5.

b/ La cellule "Description de Données" renferme toutes indications sur les conditions de traitement d'une valeu r simple (Modèle de données, dans le cas de la description d'une Edition; attribut PL/1, dans le cas de la description d'un Fichier).

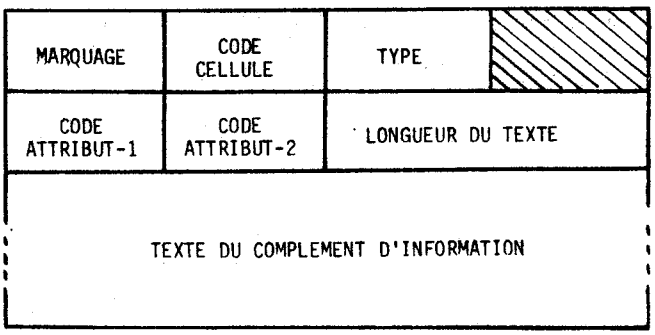


figure 3.2.6.

- Le type est représenté par l'indicateur sémantique s'il s'agit d'un attribut PL/1 ou par le code de la lettre identifiant le format (A,B,C, E,F,P,X).

- Les codes attributs désignent, par leurs indicateurs sémantiques, les attributs PL/1 accolés éventuellement à l'attribut PL/1 initial et dont le rôle est d'apporter plus de précision.

  EXEMPLE : "FIXED BINARY"

- La zone texte est destinée à renfermer tout complément d'information se rapportant à des attributs PL/1 ou à la lettre d'un modèle de données.

Application

Modèle de données
"(C(F(15,2),F(8,5)))

Attribut
FIXED BINARY (15,6)

figure 3.2.7.

REMARQUE

L'analyse de conformité du modèle de données ou des attributs PL/1 se fait au niveau du pré-compilateur, bien que PL/1 assume leur interprétation. Ce choix a été motivé par la possibilité, donnée au programmeur, de demander, pour faciliter la mise au point du programme, essentiellement la pré-compilation du texte L.A.M.P.E.(Cf §1.1.1.).

c/ La cellule "Instruction", rassemble ou désigne toutes informations nécessaires à la concrétisation d'un objet.

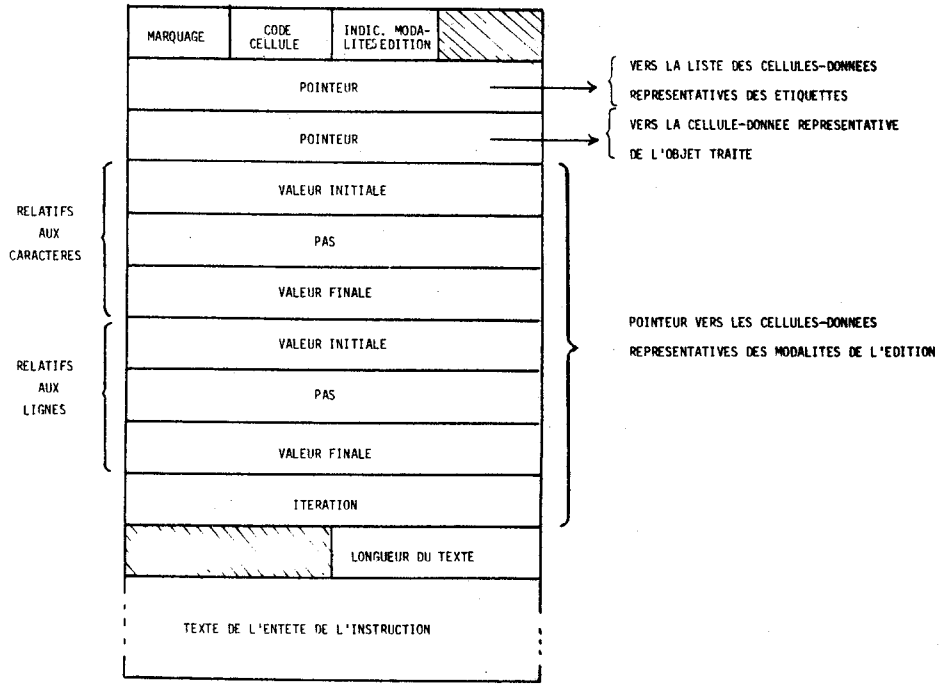
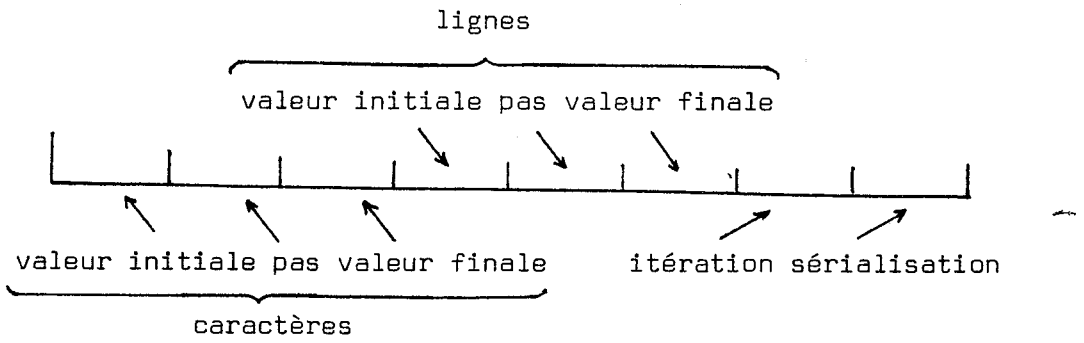


figure 3.2.8.

N.B. Nous désignons par "indicateur" des modalités de l'Edition une zone de 8 bits indiquant, chacun, l'occurrence ou non, dans instruction considérée, des diverses modalités possibles.



d/ La cellule "Déclaration", dépendante directement d'une cellule-lien, rassemble ou désigne toutes les informations définissant un objet L.A.M.P.E.

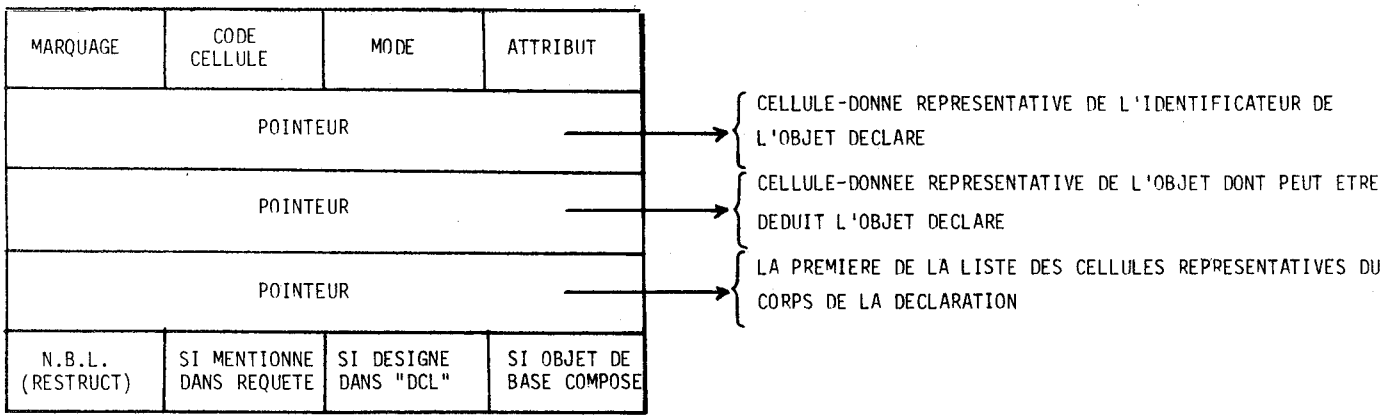


figure 3.2.9.

REMARQUE

La cellule Déclaration n'inclut pas, sous forme codifiée, les spécifications figurant dans l'en-tête de la déclaration.

En effet, les spécifications (Cf.Manuel de Référence, §1.2.5.2.2.6) fixant le mode et l'interdépendance des paramètres dont peut dépendre la description d'un objet propre à L.A.M.P.E., concernent des objets virtuels en nombre variable. Par conséquent, les paramètres et leurs spécifications sont exprimés sous la forme d'une liste de "cellules données"; ce qui, en outre, favorise le travail du Décompilateur (Cf.§2.3.2.).

e/ La cellule "Opérande" a pour but de décrire la composition des diverses requêtes dans le corps de la déclaration de tout objet éditable.

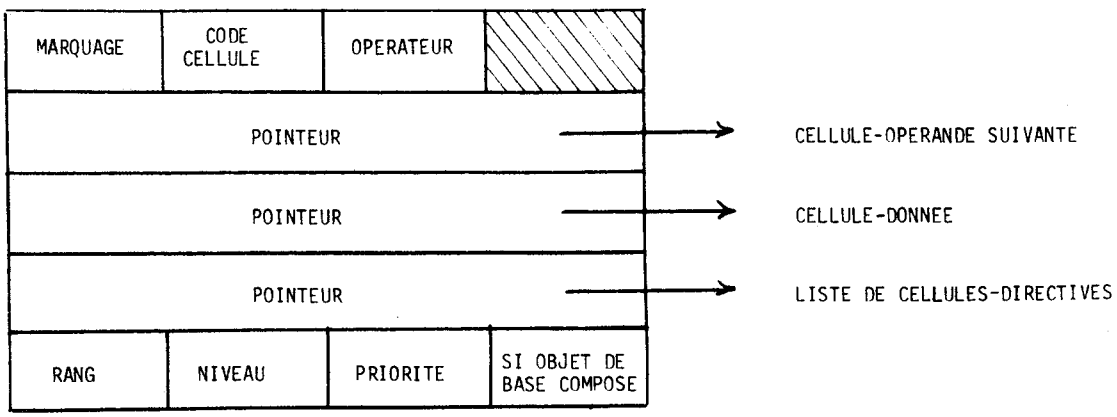


figure 3.2.10.

REMARQUES

Sur la constitution des requêtes

Rappel : Les commandes constitutives de toute requête peuvent être introduites par une directive générale ou par une directive spécialisée.
Dans le premier cas, elles décrivent une entité et structurent la requête en groupements d'opérandes (ou "groupes").
Dans le deuxième cas, elles se bornent à définir les contraintes de représentation à appliquer à des objets ou à des groupes.

Conséquences : La requête, étant constituée à partir d'une ou de plusieurs commandes, peut résulter de l'association de plusieurs "groupes".

Sur la paramétrisation des opérandes

Paramètrer un opérande figurant dans le corps de la déclaration d'un objet éditable revient à mentionner :

* la requête à laquelle il appartient
* le "groupe" qui le contient au sein de la requête
* son numéro d'ordre dans le groupe.

A la notion de requête est associé le paramètre "RANG"
A la notion de groupe est associé le paramètre "NIVEAU"
A la notion de numéro d'ordre est associé le paramètre "PRIORITE"

Sur la traduction des requêtes dans le Langage Intermédiaire

Le corps de la déclaration de tout objet éditable, quel que soit le nombre des requêtes qu'il comporte, engendre une seule liste de Cellules-Opérandes dont la constitution respecte les principes suivants :

- Une directive générale indique le début d'un "groupe". Son occurrence provoque la génération d'une Cellule-Opérande.
- Une directive spécialisée s'adresse à un groupe ou à un objet indifféremment. Sachant que plusieurs directives spécialisées peuvent concerner un même opérande (Cf.Manuel de Référence §1.4.2.) leur occurrence provoque la génération de Cellules-Directives repérées par la Cellule-Opérande relative à l'objet ou au groupe concerné.
- Un identificateur d'objet ou un littéral représente un objet. Son occurrence provoque la génération d'une Cellule-Donnée repérée par une Cellule-Opérande simultanément générée.

- Le mot-clé FIN délimite la portée d'une directive. S'il concerne une directive générale, son occurrence provoque la diminution d'une unité de la valeur attribuée au paramètre "Niveau" à affecter à la prochaine cellule-opérande.
  La portée des directives (générales ou spécialisées) est vérifiée par la gestion d'une pile.

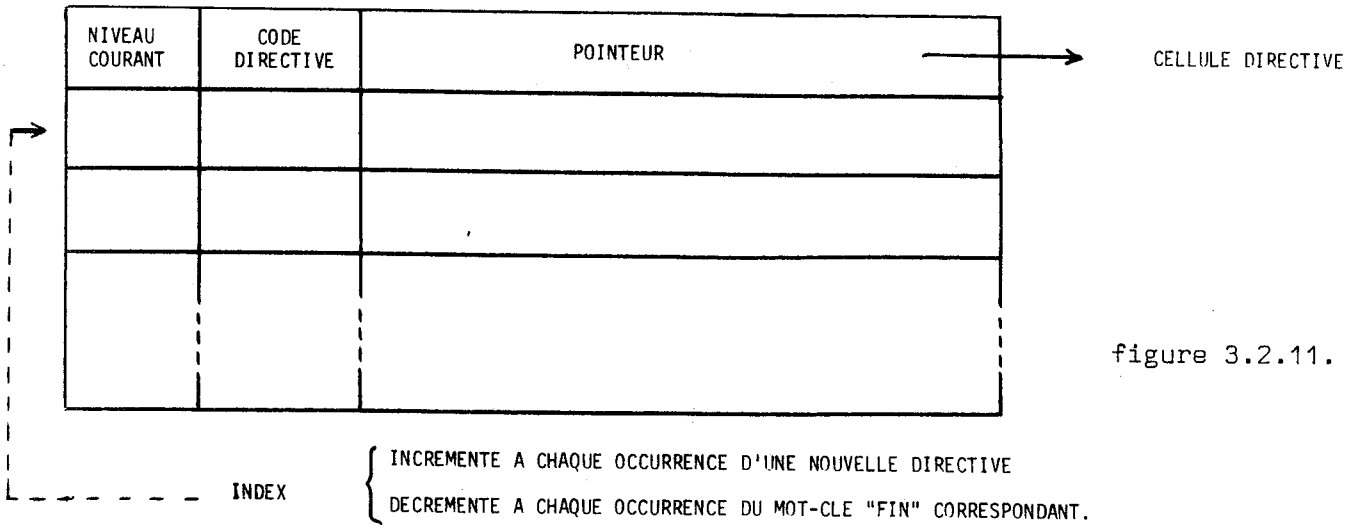


figure 3.2.11.

f/ La cellule "Directive" comporte les diverses contraintes de représentation imposées à un opérande par chaque directive spécialisée.

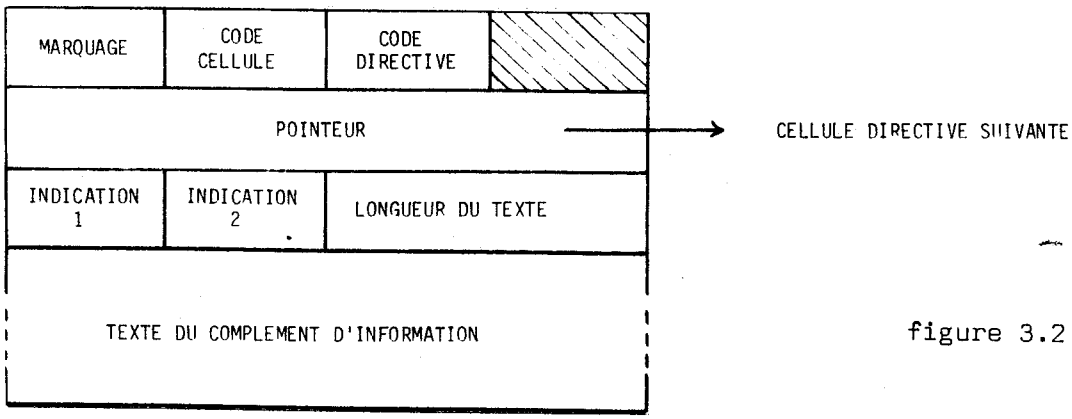


figure 3.2.12.

Selon que le complément d'information, éventuellement associé à la directive spécialisée, est constitué de mots-clés ou d'une chaîne de caractères, c'est le code des mots-clés ou le texte de la chaîne de caractères qui est inclus dans la cellule.

EXEMPLE : IMPLANTER (HAUT,DROITE) ...
SOULIGNER ('...') .....

g/ La_cellule-lien permet d'unir, sous forme d'une liste, les ensembles de cellules représentatives des déclarations concernant les divers objets impliqués dans la concrétisation déclenchée par l'instruction en cours d'analyse (Cf: §2.2.1.)

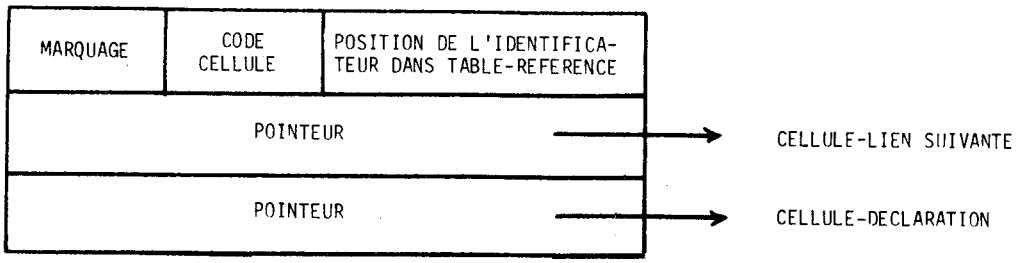


figure 3.2.13.

h/ La_cellule-Noeud coordonne les informations hiérarchisées constitutives du corps de la déclaration de tout objet non-éditable.

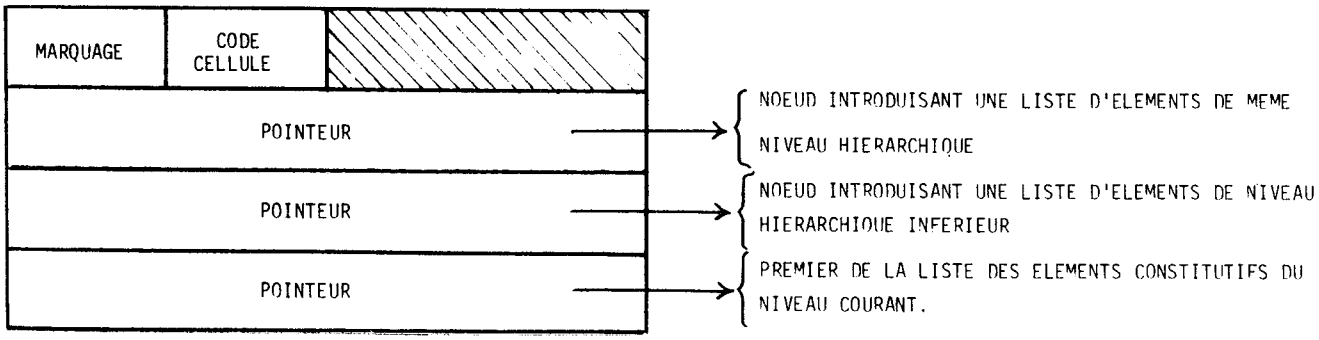


figure 3.2.14.

2.2.4.3. Les aptitudes de PL/1 à générer des listes de cellules

Nous nous bornerons à donner dans les quelques lignes suivantes les indications essentielles sur les possibilités de PL/1 à traiter des listes de variables.

Nous renvoyons le lecteur, pour plus amples informations, aux ouvrages spécialisés [BE-3-1][IB-3-1][IB-3-2].

a/ Description_des_listes_de_cellules

Le langage intermédiaire étant constitué de huit cellules distinctes et spécialisées, la traduction d'un texte L.A.M.P.E. se ramène à générer et à chaîner un nombre, à priori non déterminé, de versions différentes de chaque type de cellule.

Chaque type de cellule n'est autre qu'une structure PL/1. Toute déclaration de variable pouvant être, en PL/1, statique (attribut: STATIC) ou dynamique (attributs: AUTOMATIC, CONTROLED ou BASED), c'est cette dernière forme qui a été retenue.

Une variable basée offre l'avantage de pouvoir être allouée plusieurs fois par le programmeur. Chaque allocation demeure accessible, à tout instant, grâce à l'indication fournie par une variable "POINTER". Une variable "POINTER" peut faire partie intégrante de variables chaînées(cellules). PL/1 est ainsi particulièrement adapté au traitement de listes.

b/ Description des cellules

Toute structure basée ne peut admettre pour élément une variable de longueur non précisée (variable de type VARYING par exemple).
Pour remédier à cette restriction de PL/1, nous utilisons l'option REFER.

EXEMPLE :

```
DCL I  struct   BASED ( P ),
       2 a      FIXED BIN (15),
       2 b      CHAR  (x REFER ( a ) ) ;
```

```
Avec :    DCL    x       FIXED BIN ( 15 ) ;
D'où :           x = 18 ;
                 ALLOCATE struct SET (P);
```

Toute variable"basée" ne peut comprendre un tableau dont les bornes sont définies dynamiquement. C'est l'une des raisons pour laquelle a été conçue la cellule-Opérande et non une cellule représentant globalement le corps de la déclaration d'un objet éditable.

c/ Recherche de l'efficacité maximale.

Générer une version d'une cellule donnée revient à allouer une structure "basée" et à initialiser le pointeur correspondant.
Il est possible de simplifier le processus de localisation des diverses versions d'une cellule dans la mémoire et de minimiser le temps de traitement. A cet effet, nous allouons les cellules dans une zone réservée (variable de type AREA) dont on a préalablement fixé les dimensions.
Ces zones réservées étant elles-mêmes "basées", il nous est possible, dans la mesure où la zone courante n'offre pas un volume suffisant,(32K octets maximum), d'en allouer dynamiquement une nouvelle.

2.3. LA DECOMPILATION ET LA GENERATION DE L'EXPANSION PL/1.

L'interprétation de la liste de cellules dressée à l'issue de l'analyse syntaxique, puis la génération du texte PL/1 équivalent à chaque instruction et déclaration L.A.M.P.E., sont les deux ultimes étapes de la pré-compilation.

Elles sont prises en charge respectivement par des modules spécialisés dénommés: DECOMPILATEUR et GENERATEUR.

2.3.1. Les considérations générales

2.3.1.1. L'analogie dans les implémentations de PL/1 et de L.A.M.P.E.

En PL/1, constantes ou variables arithmétiques, pointeurs, étiquettes, données offset, sont des valeurs dotées d'une configuration invariable. Elles utilisent un nombre pré-défini d'octets.

Par contre, chaînes de caractères, tableaux, structures, ont des dimensions et une configuration variant au gré du programmeur.

Dans le cas de traitements conventionnels, les procédures de la "PL/1-LIBRARY" traitent les premières, directement, et les secondes par l'intermédiaire de DOPES VECTORS [IB-3-4].

Ainsi se trouve désignée une structure d'informations résidant en mémoire centrale, lors de l'exécution du programme, et fournissant, à chaque sollicitation, tous renseignements sur la valeur désignée.

En L.A.M.P.E., aucun objet ne répond à une configuration rigide. Par analogie avec PL/1, à chaque objet déclaré correspondent, en mémoire, des "Vecteurs-Informations". Il s'agit de structures PL/1 apportant, lors de l'exécution du programme pré-traité, aux procédures-outils de la "SYSMPE-LIBRARY", toutes indications sur les caractéristiques du dit objet.

Cette technique d'implémentation est séduisante à plus d'un titre. Elle impose au Système M.P.E. une architecture modulaire; ce qui permet d'envisager une évolution aisée tant des techniques d'Edition proposées par L.A.M.P.E. que des performances des procédures-outils actuelles.

Elle fournit,en outre,les mêmes avantages que les "Dopes-Vectors".

En effet, Nous avons pu, en concevant L.A.M.P.E., étendre les possibilités de PL/1 et, en utilisant les Dopes Vectors, en réaliser l'implémentation sans perturber le compilateur PL/1. Par analogie, un utilisateur de PL/1-L.A.M.P.E. envisageant d'améliorer,par extension, ce langage, pourra, grâce aux Vecteurs Informations, matérialiser son projet sans altérer l'architecture du pré-processeur.

2.3.1.2. L'origine et la nature du texte généré

Les séquences d'instructions et de déclarations PL/1 insérées dans le texte-source sont déduites de la liste des cellules dressées en mémoire centrale.

2.3.1.2.1. Le rôle des expansions

a/ Description des Vecteurs-Informations sous forme de déclarations de structures PL/1.
L'initialisation est assurée, soit dans le cadre de la déclaration (attribut INITIAL), soit par une séquence d'instructions complétant la déclaration. Cette dernière solution est retenue seulement lorsque la structure PL/1 est "basée" ou que certaines de ses valeurs sont déduites de traitements effectués au cours de l'exécution du programme pré-traité (transformation de variables PL/1 en chaînes de caractères, analyse des DOPES-VECTORS PL/1, etc...)
Cette solution offre, en outre, l'avantage d'autoriser une initialisation différée. La séquence d'instructions se présente alors sous forme d'une procédure.

b/ Allocation dynamique contrôlée des Vecteurs-Informations dont les déclarations ont une portée globale.

c/ Mise-en-oeuvre à l'aide d'une instruction CALL ou d'une instruction d'affectation :

* de procédures-outils assurant un traitement conventionnel en vue de la concrétisation d'un objet.

* des séquences d'instructions générées en vue de l'initialisation différée des Vecteurs-Informations.
* des Dispositifs de Formation et d'Impression des pages physiques (Cf.première partie §2.3.2.)

Notons que toute mise en oeuvre peut être immédiate ou différée (instruction de mise en oeuvre insérée dans une procédure).

d/ Traitement itératif à l'aide de l'instruction DØ.
Il est requis par la mise en oeuvre, éventuellement répétitive, du dispositif de Formation et du Dispositif d'Impression des Pages-Physiques. (par exemple: rangement sur support externe des informations constitutives de l'objet Fichier).

2.3.1.2.2. La nature des expansions

En fonction de leur rôle, les expansions se répartissent :
* en déclaration PL/1 de tout type.
* en instructions PL/1 puisées dans une gamme volontairement réduite (affectation, CALL, DØ, END, PROCEDURE, PUT STRING, ALLOCATE, FREE).

2.3.1.2.3. L'insertion des expansions

Les instructions PL/1 générées apparaissent toujours en lieu et place de l'instruction ou de la déclaration L.A.M.P.E. dont elles sont issues.

Il n'en est pas de même pour les déclarations PL/1 qui peuvent définir des valeurs de portée globale. A cet effet, l'organisation du programme se trouve systématiquement modifiée par le pré-traitement.
D'où :

Avant la pré-compilation	Après la pré-compilation
TRAITER : PROC OPTIONS(MAIN) ;	TRAITER : PROC OPTIONS(MAIN) ;
	E_OOGDEBUT :
	END E_OOGDEBUT ;
END TRAITER ;	END TRAITER ;

2.3.1.3. La normalisation du texte généré

Nous retrouvons dans l'étude des expansions les mêmes distinctions que celles signalées dans la description de L.A.M.P.E.

Expansions
- de déclarations
  - d'objets éditables (corps de la déclaration constitué de requêtes)
  - d'objets non-éditables (corps de la déclaration sous forme structurée)
- d'instructions

2.3.1.3.1. L'expansion des déclarations L.A.M.P.E.

Toute déclaration L.A.M.P.E. comporte deux parties: l'en-tête et le corps (Cf. Manuel de Référence §1.2.1.)

- l'en-tête engendre, sous forme de déclaration PL/1,
  * Le Vecteur-Information renseignant sur le type(mode) et sur les particularités (attribut) de l'objet déclaré.
  * Les Vecteurs-Informations représentatifs des objets (paramètres) dont dépend l'objet déclaré.

- Le corps engendre :
  * sous forme de déclarations PL/1, les Vecteurs Informations décrivant l'objet.
  * sous forme d'instructions PL/1, si nécessaire, la séquence d'initialisation des Vecteurs Informations de base (Cf. §2.3.2.3.3.) et la mise en oeuvre des procédures-outils éventuellement requises.

2.3.1.3.2. L'expansion des instructions L.A.M.P.E.

L'expansion des instructions L.A.M.P.E. comporte toujours :

- Une déclaration PL/1 décrivant un vecteur-Information destiné à regrouper les résultats fournis par le Dispositif de Formation des pages physiques
- Des instructions PL1/ assurant :
  * la mise en oeuvre éventuelle des séquences d'initialisation des Vecteurs-Informations de l'objet concerné.
  * la mise en oeuvre systématique des Dispositifs de Formation et d'Impression.

Texte source

```
BEGIN;

     DCL aediter propo FIXE;

     FIN  aediter ;

     PUT LAMPE (aediter);

END ;
```

Texte pré-traité

```
BEGIN;
   DCL aediter...;/*Vecteur-Information*/
   DCL.......; /*Vecteur-Information*/
    :
   DCL.......;/*Vecteur-Information*/
   P_OOGINIT:PROC;la séquence d'initialisa-
              :    tion*/
              : ===
          END P_OOGINIT;
   CALL P_OOGINIT;/*Activation de l'ini-
                    tialisation*/
   CALL T_FORM(aediter,...)/*Formation*/
   CALL T_IMPR(aediter,...)/*Impression*/

END;
```

2.3.1.3.3. Les Vecteurs-Informations

En fonction de leurs attributions, les Vecteurs-Informations se répartissent de la façon suivante :

* Vecteurs-Informations-Directeurs, destinés à régir le fonctionnement des procédures-outils du Système M.P.E.
* Vecteurs-Informations de Base, destinés à fournir, aux processus et procédures-outils, toutes indications sur leur domaine d'application.

Il nous est ainsi possible de distinguer:

a/ Les Vecteurs-Informations-Directeurs relatifs à l'en-tête des déclarations

MARQUAGE	CODE VECTEUR	MODE	ATTRIBUT
X		Y	
H		L	

↔ Coordonnées
- fournies, en vue de l'édition, par le dispositif de Formation
- exploitées par le dispositif d'Impression.

figure 3.2.15.

b/ Les Vecteurs-Informations-Directeu rs relatifs aux directives

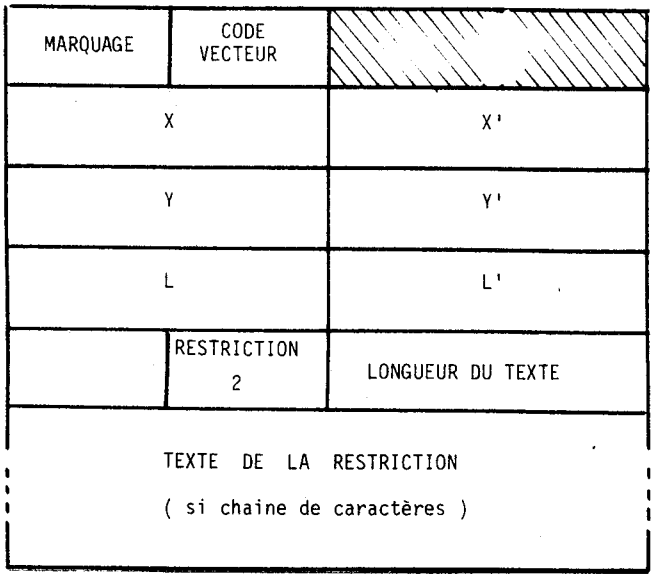


figure 3.2.16.

c/ Les Vecteurs-Informations directeurs relatifs aux corps des déclarations

Leur configuration varie en fonction de la nature de l'objet déclaré (éditable ou non-éditable). Leur dimension dépend du volume des informations constitutives du corps de la déclaration (voir: §2.2.4.2.-e)

d/ Les Vecteurs-Informations de base relatifs à des variables PL/1 dont la valeur est modifiée au cours de l'exécution du programme L.A.M.P.E.

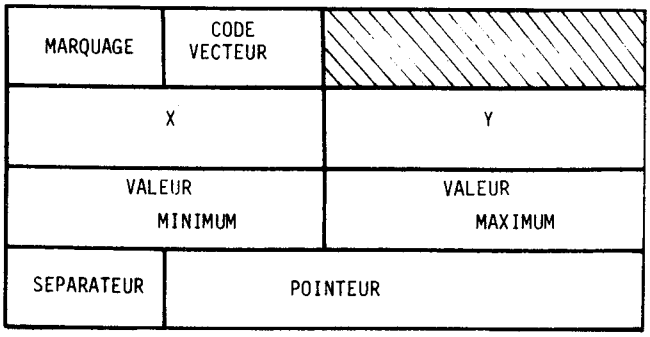


figure 3.2.17.

e/ Les Vecteurs-Informations relatifs à des valeurs essentiellement soumises à l'édition.

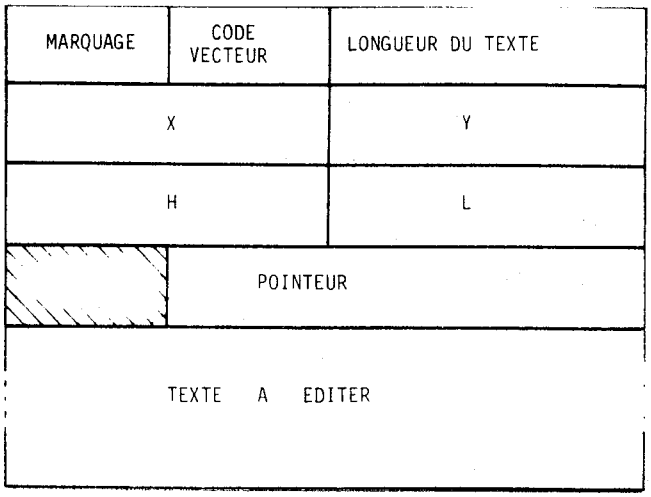


figure 3.2.18.

La chaîne de caractères représentative de l'information à éditer peut être obtenue de diverses façons selon que l'information est:

- une variable PL/1
  - de type "chaîne de caractères":
    la chaîne de caractères est obtenue grâce aux indications du "DOPE-VECTOR"
  - de type "valeur arithmétique" :
    Traitement identique à celui nécessité par le littéral L.A.M.P.E. de même type, si un modèle de données est mentionné. Sinon recours à une conversion.

- un littéral L.A.M.P.E.
  - de type "chaîne de caractères":
    Aucun traitement n'est nécessaire. La valeur est directement utilisable.
  - de type "valeur arithmétique" :
    La chaîne de caractères équivalente est obtenue grâce à l'utilisation de l'instruction "PUT-STRING".

REMARQUES

- En L.A.M.P.E., afin d'autoriser toutes les écritures émises par PL/1 et, compte tenu du fait qu'un littéral ne participe à aucun autre traitement que ceux nécessaires à son édition, une constante arithmétique peut indifféremment s'exprimer sous la forme :
  * d'une constante chaîne de caractères: '3.141593'

* d'une constante numérique : 3.141593 ou 314.1593E+02

L'édition est identique dans les deux cas. Seule la durée du traitement diffère puisque la chaîne de caractères représentative de la constante numérique nécessite un traitement confié à l'instruction PUT STRING.

- Le choix entre la génération d'une instruction PUT STRING et d'une instruction d'affectation est dicté tant par la nature de la valeur traitée que par le contexte. (dépendance ou non d'un modèle de données).
  Si un modèle de données n'est pas associé :
  * à une constante numérique , il est déterminé, implicitement, grâce aux indications fournies par la Cellule-Donnée.
  * à un scalaire PL/1, le problème est différent. L'absence d'indications, dans la cellule Donnée, implique une conversion.

EXEMPLE

```
                    ⎧ buffer=valeur_pl_1;/*l'instruction d'affectation
                                          assure la conversion */
DIRECTIVE
     valeur_pl_1 ;  ⎩ CALL T_SIGN(buffer);/*récupération de la chaîne des
                                          caractères significatifs*/
```

2.3.1.4. Les propriétés communes aux diverses expansions

a/ Les expansions se chargent de confier au compilateur PL/1 l'accomplissement des traitements qui lui incombent:

- Vérification de la portée des déclarations L.A.M.P.E.

La concrétisation d'un objet se ramène à mettre en oeuvre les Dispositifs appropriés du Système M.P.E., en leur communiquant comme base de travail, les Vecteurs-Informations représentatifs du dit objet.

Cette génération se traduit, dans l'expansion, par des instructions CALL ou des instructions d'affectation mentionnant les identificateurs qui figurent dans les déclarations PL/1 des Vecteurs-Informations concernés.

Nous sommes donc ramenés à un problème PL/1, puisqu'il appartient au compilateur PL/1 de vérifier, dès lors, la portée des déclarations PL/1 générées.

EXEMPLE :

Texte source

```
BEGIN;
   BEGIN;
      DCL aediter PROPOSITION FIXE;
      FIN aediter;
   END;
   PUT LAMPE (aediter);
END;
```

Texte généré

```
BEGIN;
   BEGIN;
      DCL aediter ...;/*Vecteurs-
      DCL ...........;  Informations */
      DCL ...........;
      P_OOGaediter PROC;
   END;
END;

CALL P_OOGaediter ;/*activ.init.*/
CALL T_FORM(ident des V.I....)/*Formation
CALL T_IMPOR(ident des V.I,...)/*Impres-
                                  sion
END ;
```

N.B. Dans le cas présent, la concrétisation de l'objet "aediter" se révèlera impossible. La déclaration de l'objet apparaît, en effet, dans un bloc interne à celui où figure l'instruction "PUT LAMPE".

- Interprétation des Modèles de données

L.A.M.P.E. utilise, sans aucune restriction, les modèles de données disponibles en PL/1. Bien qu'analysé par le pré-compilateur, ils sont fournis au compilateur PL/1 dont les procédures de la "PL/1-LIBRARY" en assure le traitement.

L'instruction "PUT STRING" de PL/1 permet de récupérer, dans un buffer, la chaîne de caractères délivrée qui, dès lors, est exploitable par les procédures-outils de L.A.M.P.E. D'où l'utilisation de cette instruction dans les expansions.

EXEMPLE :

Texte source	Texte généré
DIRECTIVE valeur_pl_1 (P'$9V.99');	PUT STRING(buffer) EDIT(valeur_pl_1)(P'$9V.99'); CALL T_SIGN(buffer)/*extraction de la chaîne significative*/

b/ Les expansions fournissent au Système M.P.E. le maximum d'indications

Il est intéressant d'assumer, lors de la pré-compilation le plus grand nombre possible de traitements; ce qui permet de minimiser la gamme et le volume des procédures-outils nécessaires,et d'augmenter d'autant l'espace alloué à l'exécution du programme PL/1.

D'où la distinction entre:

- les générations "immédiates" déduites directement de l'interprétation d'une fraction de la liste de cellules.
- les générations "différées" nécessitant l'accomplissement d'un traitement, avant que de procéder à la génération du texte PL/1. Tel est le cas, par exemple, des Vecteurs-Informations Directeurs relatifs au corps des déclarations d'objets éditables (Cf.§2.3.3.2.).

c/ Les expansions sont constituées d'unités syntaxiques issues d'un nombre limité de filières.

Les unités syntaxiques constitutives de chaque instruction ou déclaration PL/1 générées sont des chaînes de 3I caractères au plus.

Compte tenu des techniques d'obtention qu'elles impliquent, elles se répartissent en trois catégories :

- Les chaînes de caractères invariables: les mots-clés PL/1.
- Les chaînes de caractères transcrites du texte L.A.M.P.E.: certains identificateurs et littéraux.
- Les chaînes de caractères découlant d'un traitement:
  - * les valeurs numériques.
  - * les identificateurs propres à l'expansion. Ils ne doivent être à l'origine d'aucune interférence avec ceux figurant dans le texte-source.

D'où leur principe de composition :

Identificateur { de variables ou de procédures PL/1:"α_β γ <texte>"
Identificateur { de procédures-outils "T_xxxx"

Avec :

- α = Signification de l'identificateur { V=Variable / P=Procédure / E=Etiquette
- β = Numéro d'ordre de la génération { 00= Un seul exemplaire / 01 à 99 } Numéro de la génération du même type
- γ = Propriété de la variable ou de la procédure
- Texte :
  * l'identificateur mentionné dans le texte L.A.M.P.E. et ramené à une longueur ≤ 28 caractères.
  * un identificateur totalement généré (principalement utilisé pour les variables communes au programme).

2.3.2. Le module DECOMPILATEUR

Le DECOMPILATEUR reçoit,en entrée,des listes de cellules dressées en mémoire à l'issue de l'Analyse Syntaxique. En retour, après les avoir sélectionnées, il communique, une à une, les cellules au GENERATEUR de texte PL/1.

Avant que de passer le contrôle au GENERATEUR de texte PL/1, le DECOMPILATEUR a la charge :

* de vérifier la validité sémantique du texte L.A.M.P.E.
* d'effectuer les corrélations nécessaires entre les représentations des divers éléments du texte L.A.M.P.E.

2.3.2.1. La vérification de la validité sémantique du texte L.A.M.P.E.

La vérification de la validité sémantique du texte L.A.M.P.E. se trouve facilitée du fait de la représentation sous forme de listes de cellules. Dans les cas les plus courants, il s'agit de vérifier la conformité des indications portées par la cellule sélectionnée. Dans les cas les plus complexes,

nécessitant des retours arrière , la sélection aisée des cellules et sous-listes de cellules est un facteur appréciable tant de rapidité que de simplicité.

A titre indicatif, citons deux exemples:

a) Compatibilité entre paramètres spécifiés dans l'en-tête et paramètres mentionnés dans le corps d'une déclaration.
A cet effet, il suffit de s'assurer de la concordance entre les listes de Cellules-Données déduites respectivement de l'en-tête et du corps de la déclaration, à condition qu'elles représentent les mêmes paramètres.

b) Compatibilité entre arguments et paramètres
A cet effet, il suffit de vérifier la concordance entre les listes de Cellules-Données représentatives des arguments et celles correspondantes, représentatives des paramètres de l'objet désigné.
Il est à noter que les listes de Cellules-Données déduites de l'en-tête des déclarations n'ont pas la même organisation dans les deux cas.

* Dans le premier cas, elles doivent exprimer l'interdépendance entre les paramètres; condition essentielle pour la communication des arguments.

* Dans le second cas, elles traduisent l'ordre dans lequel se présentent les paramètres; condition essentielle pour la communication des arguments.

2.3.2.2. Les corrélations entre éléments du texte L.A.M.P.E.

La préparation à la génération du texte PL/1 se traduit généralement soit par une modification du chaînage des cellules appartenant à une même liste, soit par l'acquisition nécessaire d'informations puisées dans des listes disjointes.

Deux applications :

a/ Communication Arguments/Paramètres
Aux Cellules-Données représentant des paramètres mentionnés dans le corps de la déclaration sont substituées par simple modification des pointeurs, celles représentant les paramètres spécifiés dans l'en-tête.
Puis, à chacune de ces dernières sont communiquées les informations puisées dans la Cellule-Donnée représentative de l'argument correspondant.

b/ Préparation à l'expansion des déclarations d'objets éditables composés

Tout objet éditable composé peut répondre à une description complexe. Son expansion est, à ce titre, condensée au maximum (Cf.§2.3.3.2.2.). Il s'agit d'une expansion différée puisqu'elle nécessite la mise en oeuvre d'un traitement particulier dénommé "Restru ct uration" (Cf.Annexe 7) et conditionné par les propriétés des opérandes.

2.3.3. Le module GENERATEUR de texte PL/1.

Le GENERATEUR traduit en instructions et déclarations PL/1 les cellules communiquées par le DECOMPILATEUR.

Décrire le fonctionnement du GENERATEUR se ramène à décrire les techniques d'expansion. A cet effet, nous conviendrons d'étudier, en un premier temps, les propriétés générales caractérisant l'élaboration du texte PL/1, puis d'entrer dans le détail en étudiant l'expansion des déclarations et de l'instruction L.A.M.P.E.

REMARQUES

- Les déclarations de mode Proposition et Représentation, définissant des objets éditables, ont des expansions similaires. Ce qui nous amène à fusionner leur étude.
- Les déclarations de mode Modèle et Fichier, bien que définissant des objets non-éditables, présentent une différence fondamentale; la première concerne un objet participant à l'Edition, la seconde, un objet servant de base à l'Edition. Ce qui nous amènera à les étudier séparément en raison de la diversité de leur expansion.
- Le terme Vecteur-Information auquel nous ferons fréquemment appel sera remplacé par un sigle:
  * "V.I.D", pour Vecteurs-Informations Directeurs
  * "V.I.B", pour Vecteurs-Informations de Base
  * "( )",précise le type du Vecteur-Information.

2.3.3.1. L'élaboration du texte PL/1

2.3.3.1.1. La composition physiqu e des expansions

Toute instruction ou déclaration PL/1 se présente comme une chaîne de caractères née du regroupement et de la séparation par des "blancs", d'unités syntaxiques bien définies :

* mots-clés PL/1
* entiers et identificateurs extraits des cellules
* entiers et identificateurs propres à l'expansion.

Technique employée :

- tronquer les chaînes de caractères obtenues de façon à ce qu'elles soient physiquement conformes aux restrictions du compilateur PL/1. Ce dernier accepte, en effet, des chaînes de 80 caractères (ou image de carte), mais ne prend en compte que la zone s'étendant du 2e au 72 e caractère, puisque l'option par défaut est SORMIN (2;72).

- Générer sur le champ l'expansion correspondant à l'ensemble des cellules qui lui sont fournies.

Toute instruction et déclaration PL/1 se répartit en un ou plusieurs enregistrements du fichier "Texte pré-compilé" identifiés, chacun, par un numéro d'ordre déduit de celui fourni par la cellule-lien.

Un programme de tri assure la répartition convenable des expansions au sein du fichier "Texte pré-traité" avant qu'il ne soit soumis au compilateur PL/1 (Cf.§2.4.2.).

2.3.3.1.2. La composition logique des expansions

En vertu des principes généraux exposés au §2.3.1., le texte généré ne fait que traduire :

- des organisations, en mémoire centrale, regroupant les Vecteurs-Informations appropriés.
- la composition et l'initialisation des Vecteurs-Informations utilisés.
- la mise en oeuvre, le cas échéant, de procédures cataloguées telles que :
  * procédures-outils indépendantes
  * procédures-outils constitutives des Dispositifs Formation et d'impression.

Technique employée :

Elle est basée sur le respect de deux considérations essentielles:

a/ L'expansion des déclarations L.A.M.P.E. ne doit pas augmenter considérablement la place en mémoire centrale, lors de l'exécution du programme prétraité.

D'où la nécessité :

- de minimiser le nombre de Vecteurs-Informations présents dans chaque expansion.
- d'éviter de représenter les Vecteurs-Informations par des structures PL/1 basées ou contrôlées [BE-3-1][LA-2-1]

b/ La traduction en PL/1 des diverses descriptions L.A.M.P.E. s'exprime sous forme d'une corrélation logique entre les Vecteurs- Informations.

Il suffit à cet effet, d'établir un lien entre les variables PL/1 concernées. Ce qui s'exprime par la simple initialisation ou restauration d'indice et, quelquefois, de pointeurs. Un tel traitement doit être assuré différemment en fonction du contexte. Il est, de ce fait, traduit sous forme :

* d'instructions d'affectations insérées dans les séquences d'initialisation, s'il s'agit de V.I.B. représentatifs de données fixes.
* D'appels de procédures-outils, s'il s'agit de V.I.B. représentatifs de données passées en arguments ou de V.I.D.

2.3.3.2. L'expansion des déclarations des objets éditables

2.3.3.2.1. Les difficultés à surmonter

- Choisir une méthode de codification des requêtes permettant, au Dispositif de Formation, d'effectuer les manipulations requises par la détermination de la Mise-en-Pages.
- Exprimer la corrélation que le programmeur peut établir, sans restriction, entre divers objets propres à L.A.M.P.E.
- Tenir compte de la paramétrisation éventuelle.
- Respecter la portée des déclarations d'objets propres ou extérieurs.

- Traduire le type de déclaration PL/1 auquel s'apparente la déclaration L.A.M.P.E.

- Considérer la diversité des objets traités par une requête.

  En effet :

  Une directive peut porter
  - sur un identificateur d'objet : "extérieur" / "propre" (à L.A.M.P.E.)
  - sur l'identificateur d'un paramètre concernant un objet. : "extérieur" / "propre" (à L.A.M.P.E.)

2.3.3.2.2. L'organisation en mémoire centrale

L'organisation en mémoire centrale des divers Vecteurs-Informations correspondant à la déclaration d'un objet éditable L.A.M.P.E. est conforme au schéma de la page suivante.

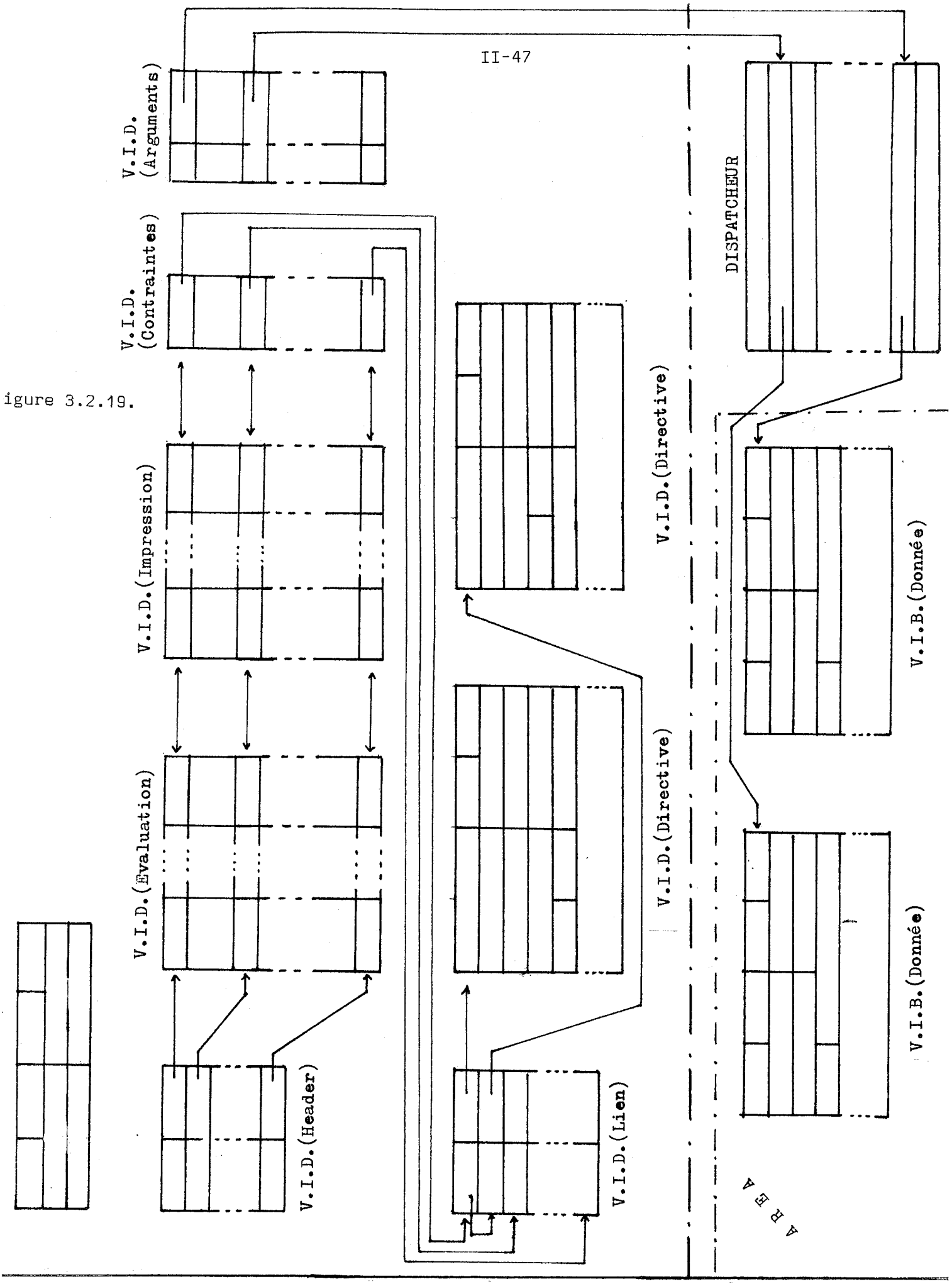
a/ Motivation d'une telle organisation

Les déclarations d'objets L.A.M.P.E. sont successivement prises en considération, lors d'une concrétisation, par les deux Dispositifs du système M.P.E. Comme nous le verrons plus en détail au chapitre 3 :

- Le Dispositif de Formation se charge de déterminer la meilleure répartition de plusieurs surfaces rectangulaires fixes (représentatives des objets) au sein d'une ou de plusieurs surfaces rectangulaires également fixes (les Pages Physiques).
- Le Dispositif d'Impression se charge d'implanter les surfaces représentatives des objets aux emplacements qui leur sont, dès lors, assignés dans les Pages Physiques et d'y ranger les informations à éditer.

Parmi les informations fournies par les déclarations, ce sont les directives qui sont à l'origine des principales difficultés de la "Mise-en-Pages".

En effet le Dispositif de Formation des Pages-Physiques doit prendre toutes les décisions nécessaires, quant à la disposition des informations instructives des divers opérandes ; opération qui peut être consécutive à l'accomplissement de découpes, si l'opérande traité fixe la Représentation d'un objet de base composé (cf. première partie § 2.3.5).


V.I.D. (Arguments)
V.I.D. (Contraintes)
V.I.D. (Impression)
V.I.D. (Evaluation)
V.I.D. (Header)
V.I.D. (Directive)
V.I.D. (Directive)
V.I.D. (Lien)
DISPATCHEUR
V.I.B. (Donnée)
V.I.B. (Donnée)
AREA
ZONE PROGRAMME
Génération en lieu et place du texte L.A.M.P.E.
ZONE COMPLEMENTAIRE

igure 3.2.19.

La détermination de ces découpes ne peut être envisagée que lorsque la surface disponible sur la Page Physique courante a été évaluée; cela, en fonction de toutes les contraintes exprimées par les directives et portant sur l'objet traité ou sur ceux désignés dans la même requête. Cette surface est obtenue en déduisant de la surface de la page courante, la surface des rectangles représentatifs des objets astreints à figurer sur cette même page; ce qui implique que le Dispositif de Formation ne prenne pas nécessairement en compte les objets dans l'ordre où ils sont mentionnés dans les requêtes. La solution adoptée, dont nous présentons les aspects théoriques et pratiques en annexe 7, consiste à paramétrer les requêtes de façon à ce que, sans modifier leur structure, on puisse exprimer sous une forme plus apte à la prise de décision les diverses contraintes qu'elles traduisent.

Ce traitement, dénommé restructuration, motive l'existence des divers V.I.D. Outre son rôle essentiel dans la technique de Mise-en- pages, il présente l'avantage de minimiser le volume du Système M.P.E. du fait qu'il est assumé dans la pré-compilation.

b/ Rôle des différents Vecteurs-Informations

- Les V.I.D. (Evaluation) et V.I.D. (Réalisation) traduisent la représentation paramêtrée des diverses requêtes figurant dans le corps de la déclaration de tout objet éditable.
- Le V.I.D. (Header) permet de délimiter, au sein des V.I.D.(Evaluation et V.I.D.(Réalisation), les représentations paramétrées de chaque requête.
- Le V.I.B. (Directives) regroupe toutes les indications sur les directives spécialisées figurant dans le corps de la déclaration.
- Le V.I.B.(Donnée) contient la chaîne de caractères à éditer, représentative de chaque littéral L.A.M.P.E. ou variable PL/1 figurant, en tant qu'opérande, dans le corps de la déclaration.

  Un même littéral ou une même variable PL/1 pouvant entrer dans la composition de plusieurs objets éditables, le V.I.B. représentatif doit être mis en commun.

A cet effet, les V.I.B.(Donnée) :

* sont des structures PL/1 "basées",
* sont regroupés dans la zone "complémentaire" du programme pré-traité, en raison de la portée nécessairement globale de leur déclaration.
* sont désignés individuellement par des variables de type "Pointeur" regroupées en une table : le DISPATCHEUR.

- Les V.I.D. (Contraintes) et V.I.D.(Arguments) établissent la corrélation entre les représentations paramétrées de chaque requête, et celles de leurs éléments constitutifs respectifs (Directives spécialisées et Opérandes).

REMARQUE

La diversification des Vecteurs-Informations, implantés dans la zone programme, est essentiellement motivée par la volonté d'établir un juste compromis entre la minimisation du texte PL/1 généré et l'occupation mémoire.

2.3.3.2.3. Le texte PL/1 généré

Nous allons apporter des explications sur le rôle et les modalités d'obtention des expansions de tout objet éditable.

Pour plus amples détails, nous renvoyons le lecteur à l'annexe 8 qui se propose :

* de schématiser les principes d'obtention du texte PL/1
* d'appliquer ces principes à un exemple général complexe
* de montrer la progression de l'expansion
* de présenter le texte pré-traité.

a/ Génération des déclarations PL/1

Les déclarations PL/1, définissant les vecteurs Informations représentatifs de chaque objet éditable, apparaissent en lieu et place de la déclaration L.A.M.P.E. Elles décrivent des variables PL/1.
La normalisation des identificateurs générés (Cf §2.3.1.4.c.) permet à une procédure-outil d'assurer la corrélation nécessaire entre les variables PL/1 générées.

b/ Génération des séquences d'instructions

b-1 Respect du type de la déclaration

Les déclarations d'objets éditables L.A.M.P.E. étant assimilées à des déclarations PL/1 AUTOMATIC ou CONTROLLED (Cf Manuel de Référence §1.2.2.2.), deux modes d'initialisation des Vecteurs-Informations sont envisagés : le mode séquentiel et le mode contrôlé.
Ainsi,

- relèvent du premier mode, les déclarations d'objets Représentation. La mise en oeuvre de la séquence d'instructions générée est assurée une fois prises en compte les déclarations faisant partie de l'expansion.

- relèvent du deuxième mode, les déclarations Proposition. La mise en oeuvre de la séquence d'instructions générée est réalisée sur l'initiative de l'instruction PUT LAMPE. Ce qui implique que la séquence prenne la forme d'une procédure PL/1 (Cf §2.3.1.2.-a).

b-2 Respect de la paramétrisation

Tout objet paramétré peut être concrétisé plusieurs fois dans le même programme avec des arguments différents.
Dans le cadre de l'implémentation, il est essentiel de distinguer les opérandes fixes et les opérandes passées en arguments.
A cet effet,

- la séquence d'instructions relative au traitement des opérandes fixes est unique et invariable (Cf annexe 8).

- la séquence d'instructions relative au traitement des opérandes passés en arguments, est générée en autant de versions qu'il y a d'appels différents de l'objet paramétré. Les identificateurs des diverses versions de la procédure représentative de cette séquence d'instructions, se différencient par la valeur numériqu qu'ils comportent (Cf.§2.3.1.4.c).

b-3 Respect de la portée des déclarations

L'implantation des séquences d'instructions n'est pas rigide. Elle est conditionnée par le respect, en toutes circonstances, de la portée des déclarations PL/1 générées, ou existantes, et décrivant les objets auxquels s'adressent les requêtes

A cet effet :

- la séquence d'instructions traitant les opérandes fixes figure en lieu et place de la déclaration L.A.M.P.E. dont elle est déduite (Cf Annexe 8)

- la séquence d'instructions traitant les opérandes passés en arguments précède l'expansion de l'instruction PUT LAMPE courante (Cf Annexe 8)

c/ Principales fonctions des instructions PL/1 générées

c-1 Allocation, Initialisation et Corrélation des V.I.B. (Donnée)

- l'allocation consiste à désigner une structure basée, au sein d'une "AREA", à l'aide d'un "pointeur" rangé dans une table spécialisée le "Dispatcheur".

- L'Initialisation affecte, lors de l'exécution du programme, une valeur à chaque structure basée représentative d'un V.I.B.(Donnée)

- La Corrélation se traduit par l'inclusion, dans le V.I.D.(Arguments), d'un indice désignant, dans le "Dispatcheur", le pointeur relatif à la structure basée.

N.B. Les V.I.B.(Donnée) peuvent concerner des littéraux L.A.M.P.E. ou des variables PL/1 indifféremment. Cette distinction est à l'origine de la solution adoptée pour minimiser tant la mémoire occupée que le temps de traitement.

Ainsi,

- Dans le cas d'un littéral L.A.M.P.E. il est possible de savoir, dès la pré-compilation, qu'un V.I.B.(Donnée) a déjà été généré. L'expansion se traduit par une instruction assurant la corrélation avec le V.I.B. déjà alloué : "V_OOAident(α) = β;"
  avec : α = entier désignant un élément du V.I.D.(Argument)
  β = entier désignant un élément du Dispatcheur.

- Dans le cas d'une variable PL/1, c'est seulement lors de l'exécution du programme que l'on peut gérer l'allocation des V.I.B.(Donnée); opération qui ne se révèle possible que pour les variables passées en arguments.

D'où :

pour chaque argument {
```
  ALLOCATE V_OOGdonnée SET(V_OOGP IN V_OOGarea)
<instructions d'initialisation de V_OOGp→V_OOGdonnée)
V_OOGp = V_OOAident(nbr)=T_LIBR;/*récupération*/
V_OOGdispatch(V_OOGi)=V_OOGp;/*corrélation*/
```

expansion de l'instruction "PUT LAMPE" { . . .

Pour chaque argument {
```
FREE V_OOGdispatch(V_OOAident(nbr))→V_OOGdonnée;
V_OOGdispatch(V_OOAident(nbr))=0;/*libération*/
```

c-2/ Activation de la préparation à la Représentation

Seule l'expansion d'un objet Représentation concernant un objet de base composé peut donner lieu à l'opération de "Préparation à la Représentation". Il s'agit d'activer des procédures-outils conduisant au rangement, sur support externe ou en mémoire centrale, selon un ordre pré-déterminé, des informations constitutives de l'objet de base. Préalablement mises sous la forme de chaînes de caractères, ces données sont prêtes à l'Edition. Le choix de la mémoire de rangement est fonction du volume de l'objet de base. Cette technique d'organisation des données, permet d'établir la corrélation physique entre les composants de l'objet de base et de l'objet Modèle approprié; corrélation indispensable à toute Représentation et impliquée par la directive ASSOCIER.

c-3/ Corrélation des V.I.D. avant l'activation des dispositifs

L'activation des dispositifs du Système M.P.E. implique la communication des Vecteurs-Informations à concrétiser. Comme tout objet L.A.M.P.E. peut être composé à partir d'autres objets L.A.M.P.E., il est nécessaire d'établir entre les V.I.D. une corrélation physique déduite de la corrélation logique exprimée par le programmeur.
Les V.I.D. n'étant pas représentés par des variables basées PL/1 sont fusionnés avant d'être passés en arguments aux dispositifs.
A cet effet, l'activation d'une procédure suffit à initialiser une structure PL/1 unique allouée dynamiquement (Annexe 7).

Notons que ce regroupement des Vecteurs-Informations ramène la corrélation physique, à l'affectation d'une valeur entière aux variables non initialisées du V.I.D.(Arguments). Cette valeur désigne, par son numéro d'ordre dans la structure, le Header de l'objet concerné.

C-4/ Activation des séquences d'instructions générées

Les séquences d'instructions soumises à une activation contrôlée se présentant sous la forme d'une procédure PL/1, leur mise en oeuvre est classique.

d/ Distinctions entre expansions des objets Représentation et Proposition.

Les expansions que nous venons de décrire s'appliquent aussi bien aux déclarations d'objets Proposition qu'aux déclarations d'objets Représentation.

Néanmoins, dans ce dernier cas, les directives s'adressent non pas à des objets considérés dans leur totalité, mais à des sous-ensembles d'objets (de base) identifiés à l'aide des niveaux du Modèle associé.

A ce titre, l'expansion doit matérialiser une corrélation

* entre les composants de l'objet de base et de l'objet Modèle.
* entre les directives et les composants de l'objet de base par l'entremise des niveaux du Modèle.

Ce qui motive :

* la présence de la séquence de "Préparation à la Présentation"
* le rôle des valeu r s rangées dans le V.I.D.(Arguments), les quelles désignent une position, non pas du Dispatcheur, mais du V.I.D. (Structure) de l'objet Modèle (Cf §2.3.3.3.2).

2.3.3.3. L'expansion de l'objet Modèle

2.3.3.3.1. Les difficultés à surmonter

- Choisir une méthode de codification de la forme structurée permettant l'établissement d'une corrélation entre composant du Modèle et composant de l'objet de base soumis à une représentation.

- Tenir compte de la paramétrisation éventuelle et respecter, dans ce dernier cas, la portée des déclarations de variables PL/I passées en argument.

2.3.3.3.2. L'organisation en mémoire centrale

L'organisation, en mémoire centrale, des divers Vecteurs-Informations est conforme au schéma de la page suivante. Elle est motivée par les divers types de corrélation qu'implique la conception et l'utilisation d'un objet Modèle.

a/ Corrélation entre les composants d'un même niveau

Chaque composant de la la forme structurée décrivant un objet Modèle est constitué de littéraux L.A.M.P.E. , de variables PL/1 et de Modèles de données.

En vertu des principes généraux d'une expansion,

* Les littéraux L.A.M.P.E. engendrent des V.I.B.(Donnée)
* Les variables PL/1 engendrent des V.I.B.(variable)
* Les Modèles de Données sont mentionnés dans les instructions.

En conséquence,

- Les V.I.B.(Donnée) et les V.I.B.(variable) sont implantés dans l'Area figurant dans la zone complémentaire du programme pré-traité et sont désignés individuellement par un pointeur du Dispatcheur.

- Les V.I.B. représentatifs des composants d'un même niveau sont reliés entre eux grâce au V.I.D.(Lien) via le Dispatcheur.

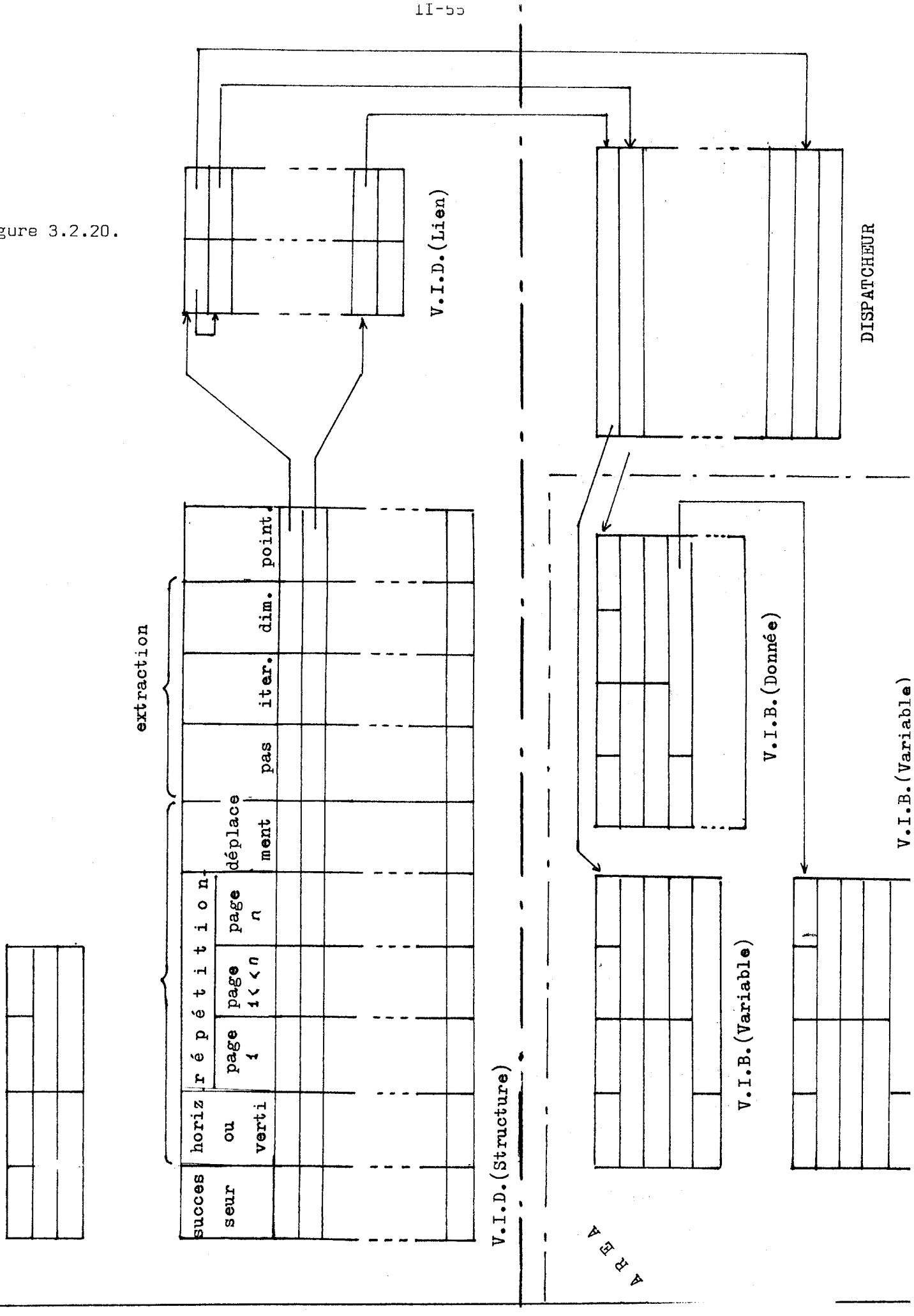
- Le V.I.D.(structure) fait la synthèse de toutes les informations nécessaires tant à la description de l'objet Modèle qu'à la concrétisation de la Représentation à laquelle il participe.

b/ Corrélation avec les Vecteurs-Informations de l'objet Représentation.

Un objet Modèle participe exclusivement aux Représentations d'objets de base. Dans la déclaration d'un objet Représentation, les directives s'adressent à chacun des niveaux de l'objet Modéle associé. Elles concernent, de ce fait, l'ensemble des composants de chaque niveau désigné qui, en vertu de l'organisation en mémoire adoptée, sont exprimés sous forme de V.I.B

Par suite, le V.I.D.(Arguments) figurant dans l'expansion de l'objet Représentation, peut désigner plusieurs V.I.B; ce qui est à l'origine du chaînage (vers le successeur) établi dans le V.I.D.(Structure).

figure 3.2.20.



énération en lieu et place du texte PL/I-L.A.M.P.E.

c/ Corrélation entre les composants du Modèle et de l'objet de base

L'indication du niveau de la forme structurée permet de sélectionner conjointement un sous-ensemble de composants de l'objet Modèle et le sous-ensemble correspondant des composants de l'objet de base à Représenter.
La corrélation nécessairement établie entre ces deux types d'informations d'origine différente ne devait, pour être satisfaisante, ne nécessiter qu'une zone mémoire réduite quel que soit le volume de l'objet de base concerné.

Ces considérations sont à l'origine,

- de la linéarisation des composants de l'objet de base dans le cadre de l'expansion de l'objet Représentation (rôle de la procédure-outil de Préparation à la Représentation §2.3.2.3.).
- De l'inclusion, dans le V.I.D.(structure) de trois paramétres (le pas, l'itération et les dimensions) destinés à selectionner,parmi les sous-ensembles de composants de l'objet de base, ceux correspondant à chaque niveau du Modéle associé.

2.3.3.3.3. Le texte PL/I généré.

a/ Génération des déclarations

Les déclarations décrivant les V.I.D. sont implantées en lieu et place de la déclaration de l'objet Modèle. Leur génération ne présente aucune particularité.

b/ Génération des séquences d'instructions

b-1 Respect de la paramètrisation

Le Modèle est un objet paramétrable. Il est possible de lui communiquer dynamiquement: littéraux, scalaire, entier fourni par une fonction.

Exemple: ASSOCIER aadjoindre (10,nombre,HBOUND(tableau));

Le principe d'implémentation est identique à celui adopté pour les objets éditables :

- Séquence unique d'instructions allouant et initialisant les V.I.D.(Donnée) et les V.I.B.(Variable) à partir des valeurs fixes figurant dans le corps de la déclaration.
- Séquences d'instructions, en nombre proportionnel au nombre d'appels allouant et initialisant les V.I.B.(Variable) à partir des valeurs passées en arguments.

b-2/ Respect de la portée des déclarations

Comme pour les objets éditables, la portée des déclarations est vérifiée par le compilateur PL/1 dans la mesure où les séquences d'instructions sont convenablement implantées :

- Séquence traitant les valeurs fixes, implantées à la suite des déclarations des V.I.D.
- Séquences traitant les valeurs passées en arguments, implantées en lieu et place de la requête d'appel. Elles sont donc incluses dans chaque expansion d'un objet Représentation.

c/ Principales fonctions des instructions PL/1 générées

Les instructions générées ont le même rôle que celles participant aux expansions des objets éditables.

Elles assurent d'une part l'allocation, l'initialisation, la corrélation des V.I.B.(Donnée) et V.I.B.(Variable), d'autre part, la mise en oeuvre des séquences d'instructions générées ou des procédures-outils appropriées.

2.3.3.4. L'expansion de l'objet Fichier

2.3.3.4.1. Les difficu l tés à surmonter

- Exprimer, par la nature de l'expansion, que cet objet de base doit être placé sur le même plan que les autres objets de base fournis par PL/1.
- Tenir compte des diverses possibilités offertes (cf. Manuel d'utilisation § 2.1.2.4) pour exprimer la dimension d'un fichier.

2.3.3.4.2. L'organisation en mémoire centrale

A la différence des objets propres à L.A.M.P.E., en raison de l'orientation adoptée précédemment, aucun Vecteur-Information n'est généré.

A partir du corps de la déclaration, est engendrée une structure PL/1. Les renseignements impliqués par l'objet Fichier sont alors puisés comme pour d'autres objets de base, dans les Dopes-Vectors PL/1.

L'avantage de cette solution est de confier au compilateur PL/1 la délicate tâche d'interpréter les divers attributs PL/1 utilisables dans le corps de la déclaration d'un objet fichier.

2.3.3.4.3. Le texte PL/1 généré

a/ Génération des déclarations

Il est procédé à la déclaration d'une seule variable PL/1 :

* ayant pour identificateur celui attribué à l'objet Fichier
* décrivant une structure PL/1 obtenue par transposition des renseignements puisés dans le corps de la déclaration L.A.M.P.E.

b/ Génération des séquences d'instructions

Une ou deux séquences d'instructions sont générées, selon que l'attribut PERMANENT n'est pas ou est mentionné.

En effet :

- une séquence facultative, tributaire de la présence de l'attribut PERMANENT, range les Enregistrements sur un support externe.

  Incluse dans le programme pré-traité immédiatement après la déclaration de la structure PL/1, elle comporte :

  * L'ouvertue d'un fichier sur un support externe
  * Des boucles "DØ" imbriquées, en nombre égal à celui des sélections dont peut être, éventuellement, déduite la dimension.
  * Une instruction d'affectation ayant pour membre gauche l'identificateur de l'objet Fichier et, pour membre droit, l'appel de la fonction d'initialisation.
  * Une instruction "PUT" assurant le transfert de l'Enregistrement (la structure PL/1) vers le support externe.
  * La fermeture du fichier sur le support externe.

- Une séquence obligatoire fournissant chaque enregistrement sous la forme d'une chaîne de caractères. Ces enregistrements sont extraits du fichier sur support externe, ou fournis par la fonction d'initialisation, selon la présence ou non de l'attribut PERMANENT.

Incluse dans l'expansion de l'objet Représentantion, fixant l'aspect de l'objet Fichier, elle comporte :

* Des boucles "DØ" répondant aux mêmes principes de génération que dans la séquence précédente.
* Des instructions d'extraction des enregistrements (soit à partir de la fonction d'initialisation, soit à partir du fichier sur le support externe).
* Des instructions transformant chaque élément de l'enregistrement, en une chaîne de caractères.

2.3.3.5. L'expansion de l'instruction L.A.M.P.E.

2.3.3.5.1. Les difficultés à surmonter

Respecter la libre possibilité, donnée au Programmeur, d'adopter pour la concrétisation d'un objet, un support autre que le papier d'imprimante et tel, par exemple, que la bande perforée, la bande ou le disque magnétique.

2.3.3.5.2. La structure de l'Expansion

L'instruction L.A.M.P.E. regroupe deux types d'indications ;

* Les unes relatives à la concrétisation (désignation de l'objet et modalités de l'Edition).
* Les autres relatives au support, éventuellement transitoire, de l'Edition (fichier décrit en J.C.L.)

L'expansion est constituée de la déclaration d'un V.I.D., en zone complémentaire, et, en lieu et place de l'instruction L.A.M.P.E., d'instructions PL/1. Parmi ces dernières :

Les unes régissent la Formation des pages physiques,
Les autres assurent l'impression, ou plus précisément, le rangement des lignes constitutives de chaque page, sur le support mentionné dans l'instruction L.A.M.P.E.

a/ L'expansion systématique

Le V.I.D. représentatif de l'instruction se présente sous la forme d'un tableau dont le nombre de lignes est tributaire des modalités de l'Edition.

Déclaré en zone complémentaire, il est alloué dynamiquement. Il regroupe les résultats destinés au Processus de Décision, et obtenus au terme des traitements assumés par les autres Processus constitutifs du Dispositif de Formation des Pages-Physiques (cf § 3.3).

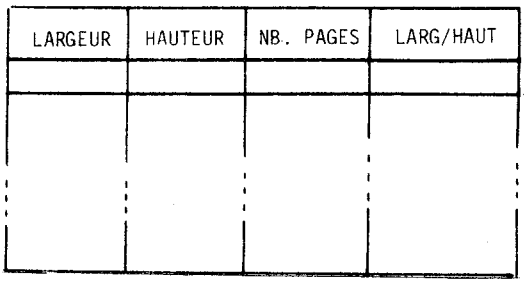


figure 3.2.21.

V.I.D.(Instruction)

b/ L'expansion tributaire des modalités de l'édition

Cette expansion a pour objectif de mettre en oeuvre les Dispositifs de Formation et d'Impression.

b-1/Influence des modalités de l'édition sur les dispositifs

- "valeur initiale"
- "pas"
- "valeur limite"

} influent ... sur la mise en oeuvre et le fonctionnement du dispositif de Formation.

- "itération" influe ... sur le fontionnement du dispositif d'Impression.
- "sérialisation" influe { sur le Dispositif de Formation (la hauteur des pages est diminuée de deux lignes) ; sur le Dispositif d'Impression(en raison de l'Implantation de "**PAGE: **")

b-2/ Mise en oeuvre de Dispositifs de Formation

Le Dispositif de Formation, traitant la représentation paramétrée et restructurée des requêtes, est mis en oeuvre en premier lieu. Il a la charge (Cf §3.3.) de déterminer :

* Le nombre de pages nécessaires à la concrétisation de l'objet et de compléter, de ce fait, le V.I.D.(instruction)
* Les découpes optima d'un objet de base, et de ranger les indications résultantes en mémoire.

b-3/ Mise en oeuvre du Dispositif d'Impression

Le Dispositif d'Impression, traitant la représentation paramétrée, et non restructurée des requêtes, est mis en oeuvre après le dispositif de Formation en raison de l'incidence qu'ont, sur son fonctionnement, les résultats acquis par ce dernier. Trois fonctions principales (Cf §3.4.) lui incombent :

* Le calcul des coordonnées du point d'implantation, dans chaque page, des informations constitutives des objets concernés.
* L'élaboration, une à une, en mémoire centrale, des pages à éditer et harmonisation de leur composition.
* L'exécution d'opérations de compactage et de décompactage de façon à prendre en compte, au moindre frais, la demande d'itération (Cf.§3.4.2.).

c/ Expansion tributaire du support de l'Edition requis.

La réalisation physique des imprimés se ramène à communiquer chacune des lignes constitutives de la page élaborée en mémoire centrale (Cf.§3.2.) sur le support (fichier au sens OS 360) explicitement mentionné dans l'instruction "PUT LAMPE". Une procédure, générée systématiquement dans la zone complémentaire du programme, assure la constitution du "Buffer". La transmission de celui-ci, au fichier retenu s'effectuant grâce à des instructions PL/1 du type :

"PUT FILE (nom du fichier) EDIT(buffer)(X(cadrage),A(largeur));"

Ces instructions sont étiquetées de façon à ce qu'un aiguillage, prenant en compte le code d'identification du Fichier sélectionne l'instruction appropriée.

Texte généré en zone complémentaire du programme.

```
P_OGEDIT : PROC ;

   Séquence d'instructions immuables sélectionnant
   chaque élément de la zone de travail (Cf.§3.2.)

   Séquence conditionnée par le nombre de supports
   d'édition requis
      → aiguillage
      → instructions "PUT FILE" (  )..."
        étiquetées et générées en nombre égal au
        nombre de supports distincts mentionné
        dans le programme-source.

END ;
```

2.3.3.5.3. Le texte PL/1 généré

Seules des instructions PL/1 sont élaborées.

- Sont générées en lieu et place de l'instruction L.A.M.P.E.
  * l'instruction d'allocation du V.I.D.(instruction)
  * les instructions de mise en oeuvre des Dispositifs
- sont générées, en zone complémentaire du programme :
  * les instructions constitutives de la procédure "P_OGEDIT" précitée.

2.4. LA CLASSIFICATION ET LA REPARTITION DU TEXTE PRE-COMPILE

Le texte pré-compilé regroupe le texte source, le texte PL/1 généré et les messages d'erreurs. En vertu de l'architecture et du fonctionnement du pré-compilateur, l'expansion PL/1 et les diagnostics sont insérés, dans le fichier "Texte pré-compilé", dès leur génération.

- Afin de respecter la structure propre à tout programme PL/1, les déclarations et instructions PL/1 générées ne sont pas toutes implantées, comme nous avons pu nous en rendre compte, en lieu et place de la déclaration ou de l'instruction L.A.M.P.E. dont elles sont issues.

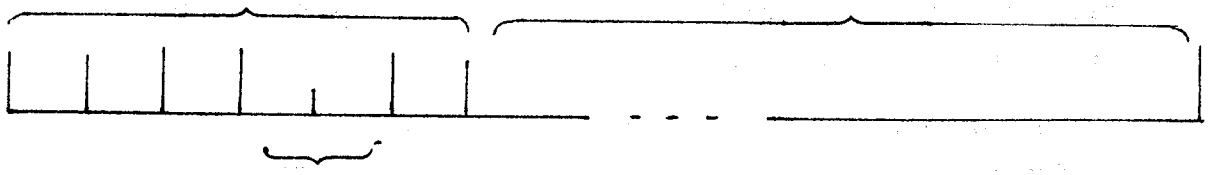
Pour remédier à l'insertion systématique et immédiate du texte généré, une opération de classification s'impose. Elle est confiée au module "TRI".

- Afin d'affecter aux fonctions qui leur sont imparties, les différents types de texte fusionnés au sein du fichier "Texte pré-compilé", une opération de répartition s'impose. Elle est confiée au module ORDONNATEUR.

2.4.1. L'organisation du texte pré-compilé sur son support physique.

Le texte pré-compilé est structuré de la façon suivante en mémoire annexe.

Chaque enregistrement du fichier, appartenant à un bloc physique de 1350 octets, comporte au maximum 135 octets répartis comme l'indique le schéma suivant :



a/ Le corps de l'enregistrement regroupe indifféremment soit du texte source (instructions et déclarations PL/1-L.A.M.P.E., commentaires), soit du texte généré (instructions et déclarations PL/1, diagnostics résultant de la pré-compilation).
Un enregistrement est créé pour toute nouvelle instruction ou déclaration (source ou générée), pour toute nouvelle image de carte, pour chaque diagnostic.

b/ L'en-tête de l'enregistrement contient les indications suivantes :

- L'indicateur de corrélation logique, décrivant la nature du corps de l'enregistrement.
  I = Instruction-source, C=commentaire, M=messages, G=génération,
  [illegible] = suite du corps de l'enregistrement précédent.

  N.B. La distinction entre instruction L.A.M.P.E. et instruction PL/1 source n'apparait pas dans l'indicateur de corrélation logique (en voir l'explication §2.1.3.b.).

- L'indicateur de corrélation physique, décrivant l'aspect externe du texte source, ou généré, compte tenu du support impliqué (cartes de 80 colonnes pour les instructions, déclarations et commentaires, ligne de 132 caractères pour les diagnostics).
  D = début de carte ou de ligne C = continuation

- La dimension indique la longueur, en caractères, de l'enregistrement
- Le numéro de l'instruction, codé sur 16 bits, permet d'établir une corrélation entre toute instruction L.A.M.P.E., les diagnostics qui la concernent et le texte PL/1 équivalent.
- Le numéro de l'enregistrement codé sur 8 bits permet de respecter l'ordre lexicographique des informations résultant de la pré-compilation de chaque instruction L.A.M.P.E.

2.4.2. Le module TRI

Le module TRI assure la mutation du texte "pré-compilé" en un texte "pré-traité".

Son rôle se borne à l'accomplissement d'un tri portant sur deux arguments :

- en mineur: le numéro de l'enregistrement
- en majeur : le numéro de l'instruction

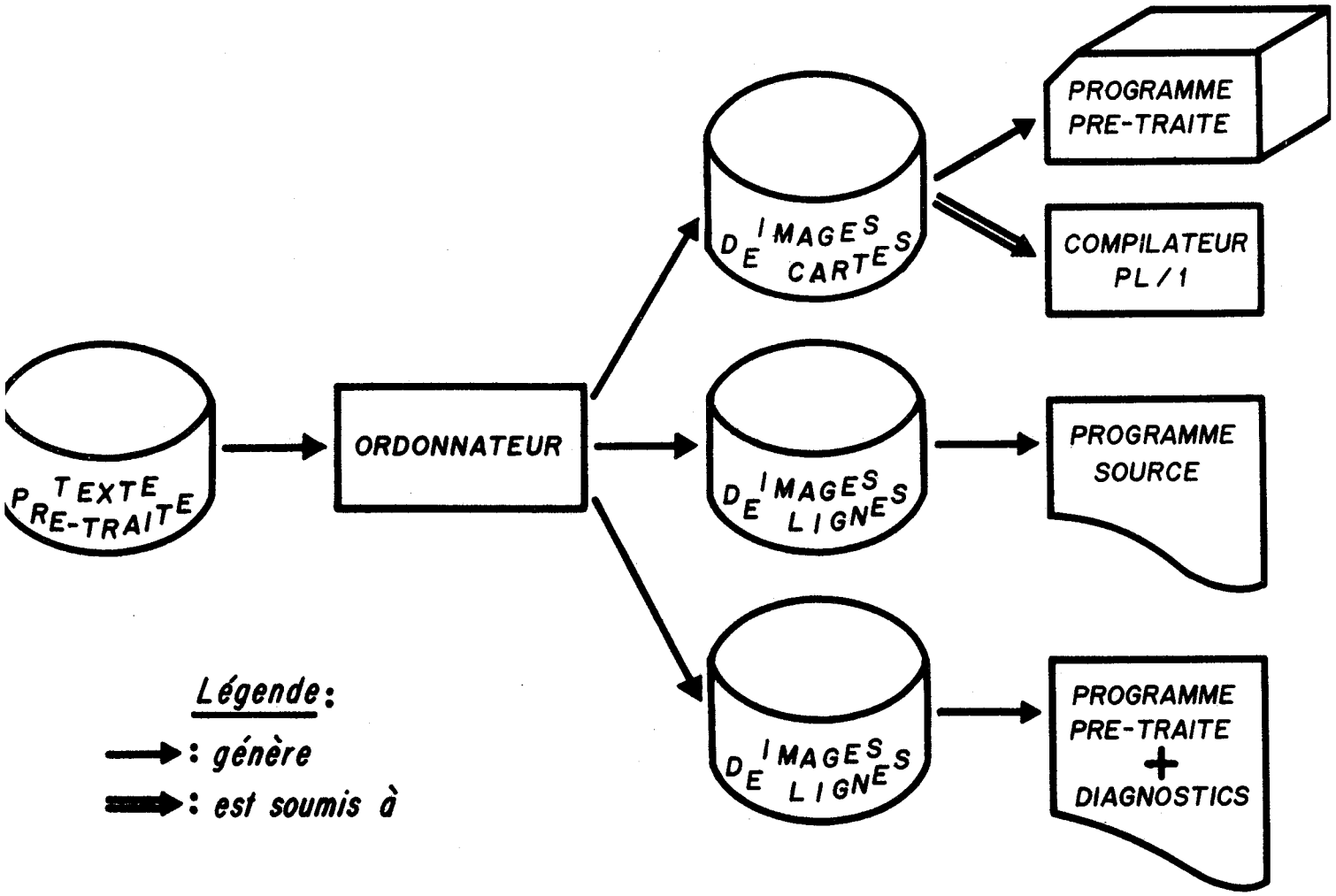
L'indication de ces arguments, ne pouvant pas être incorporée aux expansions J.C.L., se trouve rangée dans une "librairie" spécialisée: le "SYSINLIB".

Afin d'optimiser les traitements assumés par le pré-processeur, le module de Tri n'est mis en oeuvre que dans la mesure où le texte L.A.M.P.E., dont la présence a été décelée lors de l'Analyse Lexicographique, a été soumis effectivement à l'Analyse Syntaxique.

2.4.3. Le module ORDONNATEUR

L'ordonnateur a pour rôle :

* de fournir l'image du programme-source.
* de fournir l'image du programme pré-traité
* de regrouper les instructions et déclarations du programme pré-traité puis d'assurer leur communication directe, ou différée, au compilateur PL/1.



Compte tenu de l'organisation du texte pré-traité sur le support physique, il est aisé pour l'ordonnateur d'accomplir sa tâche en fonction des indications mentionnées par le programmeur, dans la carte "EXEC".

a/ Pour communiquer au compilateur PL/1, le programme pré-traité :

* seules sont prises en considération les instructions PL/1.
* les instructions L.A.M.P.E. sont exclues,ou placées en commentaire,selon que le paramètre TRACE a ou non été mentionné.

b/ Pour éditer la liste du programme-source, seuls sont pris en compte les enregistrements identifiés par les indicateurs de corrélation logique "C" ou "I".

c/ Pour éditer la liste du programme pré-traité, tous les enregistrements sont pris en compte.

TROISIEME PARTIE

C H A P I T R E III

LE SYSTEME M.P.E.

Le Système M.P.E. regroupe un ensemble de routines assurant les divers traitements qui, à partir des données à éditer et en fonction des indications apportées par le texte L.A.M.P.E., va assurer l'élaboration des imprimés finaux et en réaliser l'édition ou le stockage.

3. . LA CONSTITUTION DU SYSTEME M.P.E.

Les routines constitutives du Système M.P.E., dénommées "Procédures-Outils", sont, du point de vue technique, des procédures cataloguées figurant dans une bibliothèque privée (Cf.Figure 3.1.1.). Cette bibliothèque est l'homologue de celle propre à PL/1 et son utilisation est requise pour l'exécution de tout module-objet obtenu à l'issue d'une compilation PL/1.

Ces procédures-outils ont été rédigées, pour la plupart, en PL/1 et cataloguées au terme d'une compilation indépendante. Les autres procédures ont été écrites en Assembleur 360. Il en a été ainsi, lorsque les possibilités offertes par PL/1 s'avéraient incapables d'assurer les traitements exigés. Citons, à titre indicatif le traitement consistant à récupérer dans les "Dopes Vectors", dressés par le compilateur PL/1, les renseignements concernant certaines des variables traitées (les structures, par exemple).

Les Procédures-Outils rangées dans la bibliothèque ne sont pas toutes requises lors d'un traitement L.A.M.P.E. C'est ainsi que, dans le cadre des diverses étapes permettant d'engendrer les modules exécutables (Cf.§3.1. 1.) à partir d'un programme-source PL/1-L.A.M.P.E., il appartient à "l'Editeur de liens" de sélectionner les seules routines nécessaires.

Les procédures-outils, en fonction de leur rôle, se répartissent en deux catégories :

- Les unes peuvent être qualifiées d'indépendantes, en ce sens que leur activation suffit à fournir des résultats directement exploitables. Tel est le cas des procédures traitant les variables PL/1; traitement qui ne peut être assumé que lors de l'exécution du programme objet. Les opérations ainsi accomplies consistent toujours à ranger les valeurs sélectionnées soit sur un support annexe, soit dans les Vecteurs-Informations de base correspondant

- Les autres peuvent être qualifiées dépendantes en ce sens que les résultats qu'elles fournissent sont partiels et sont généralement exploités par des Procédures-outils complémentaires. Ces routines dépendantes sont considérées comme les unités de traitement constitutives des "Processus" requis par les "Dispositifs" auxquels le système M.P.E. fait appel (Cf 1e partie §2.1.3.).

3.2. LA ZONE DE TRAVAIL

En vertu de l'idée maitresse initialement adoptée, le programmeur a été libéré des lourdes contraintes imposées par l'établissement de la maquette de chacune des pages à éditer. Ce travail est dévolu au Système M.P.E. et, plus particulièrement, au dispositif de Formation des Pages Physiques.

Le domaine d'application des Procédures-Outils requises à cet effet est une zone de la mémoire centrale représentative de la Page-Physique telle qu'elle a été définie par l'instruction L.A.M.P.E.

3.2.1. La représentation de la zone de travail en mémoire centrale

A l'image de la Page-Physique, la zone de travail doit se décomposer en lignes et chacune de celles-ci, en caractères. Le nombre de lignes et le nombre de caractères se déduit des indications explicitement mentionnées par l'utilisateur, dans l'instruction L.A.M.P.E.

Chaque caractère de la zone de travail doit pouvoir être sélectionné et traité individuellement. PL/1 apporte, dans ce domaine, toute l'efficacité qu'il manifeste dans le traitement des chaînes de caractères. La zone de travail est décrite comme un tableau dont le nombre d'éléments est égal au nombre de lignes de la Page-Physique, et dont chaque élément est une chaîne de caractères en nombre égal à celui fixé dans l'instruction L.A.M.P.E.

3.2.2. Le repérage des informations constitutives d'un objet

La disposition des informations constitutives d'un objet se ramène à l'implantation des rectangles dans lesquels elles sont inscriptibles (Cf.première partie Cf.§2.1.3.).

L'implantation d'une information, au sein de la zone de travail, sera définie si l'on connait les dimensions du rectangle pouvant la contenir et si l'on sait en repérer la position. La détermination des dimensions ne présentant aucune difficulté, pour repérer la position d'un tel rectangle il suffit de considérer l'un quelconque des points appartenant à la région qu'il occupe.

Nous avons admis, en règle générale, de prendre toujours comme repère, le sommet supérieur gauche du rectangle (ou "point d'implantation") Sa position est, dès lors, caractérisée par ses coordonnées dans un espace orthonormé adoptant pour axes respectivement le côté supérieur et le côté latéral gauche de la zone de travail. L'origine des axes est dénommée "point de référence". Les unités sont: le caractère, pour l'axe horizontal, et la ligne, pour l'axe vertical.

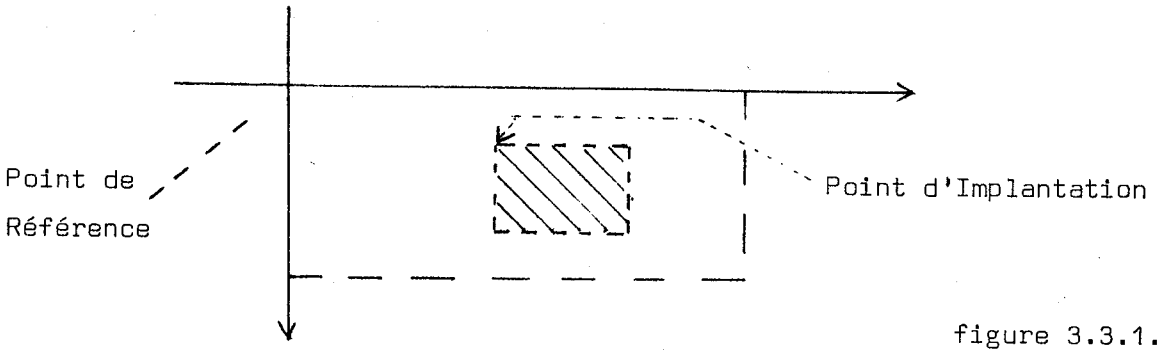


figure 3.3.1.

Compte tenu de la représentation, en mémoire centrale, de la zone de travail, les coordonnées du point d'implantation seront respectivement précisées par le numéro de l'élément dans le tableau (indice) et par la position du caractère dans l'élément. (Valeur entière utilisable dans une fonction "SUBSTRING").

3.2.3. La notion de surface réceptrice

On ne peut envisager de disposer le rectangle représentatif d'une information dans la zone de travail que dans la mesure où l'on connaît la surface disponible au moment où l'on envisage ce traitement. Dénommée "zone réceptrice", cette surface est toujours, comme nous le verrons plus loin, rectangulaire.

Par analogie, avec tous rectangles supportant les informations constitutives d'un objet, la zone réceptrice est caractérisée par ses dimensions et par les coordonnées de son sommet supérieur gauche, dénommé "Curseur"

3.3. LE DISPOSITIF DE FORMATION DES PAGES-PHYSIQUES

Le Dispositif de formation des Pages-Physiques reçoit, en entrée, les informations fournies par les différents Vecteurs-Informations (Cf.§2.3.1.3.3

* Les V.I.D. lui communiquent, sous forme paramétrée, les différentes indications mentionnées dans les requêtes.
* Les V.I.B. lui apportent toutes précisions utiles sur la nature, l'organisation et le contenu des objets de base.

Le dispositif, en retour, au terme des traitements auxquels il procède, délivre des valeurs numériques rangées soit directement dans les Vecteurs-Informations concernés, soit dans des tables dressées en mémoire.

Cette solution a l'avantage d'être rapide puisqu'elle fait largement appel à la technique d'affectation de valeurs à des variables PL/1 simples ou composées.

Le dispositif a la charge, comme son nom l'indique, de constituer les Pages-Physiques. A cet effet, il met en oeuvre chacun des trois Processus qui le composent : les Processus d'Evaluation, d'Adaptation et de Décision.

Cette première étape dans l'élaboration des imprimés doit surmonter les divers problèmes que posent la coexistence d'objets de nature et de dimensions différentes. Dans ce but, le dispositif de formation interprète les requêtes sous leur forme restructurée(Cf.Annexe 6).

3.3.1. LE PROCESSUS D'EVALUATION

Le processus d'évaluation a pour rôle de déterminer la surface réceptrice. Comme nous ne traitons pas un problème de découpes et que nous cherchons non pas à minimiser les espaces vides au sein des pages éditées, mais à les répartir de façon à rendre les imprimés facilement interprétables, nous admettons, par convention, que la surface réceptrice est toujours rectangulaire.

En vertu des considérations précédentes (Cf.§3.2.2.), une telle surface est isolée dès que l'on connait les coordonnées du curseur d'une part,

et ses dimensions d'autre part.

Sachant qu'initialement, le point de référence et le Curseur sont confondus, une simple incrémentation suffit à déduire les coordonnées courantes de ce dernier, de celles dont il était doté immédiatement avant.

Sachant que la zone réceptrice est rectangulaire et connaissant les coordonnées du Curseur, la détermination des dimensions est simple, même lorsque les Requêtes traitées mentionnent des objets à mettre en évidence. Le Schéma de la page suivante indique, dans ce cas, en faisant appel à des configurations précises, certaines éventualités.

Néanmoins, il est fréquent, qu'à un instant donné, plusieurs surfaces réceptrices puissent être retenues. Alors un choix s'impose. Un algorithme, basé sur des considérations empiriques, assure cette tâche et prend une décision arbitraire en cas de conflit.

Il s'ensuit que les résultats obtenus dans les premiers temps d'exploitation de notre produit pourront prêter à critique. Mais, en vue d'affiner rapidement et aisément les algorithmes du processus d'Evaluation, entre autres, nous avons conçu le système M.P.E. de façon à ce que des réglages puissent être progressivement effectués (Cf.§3.5.).

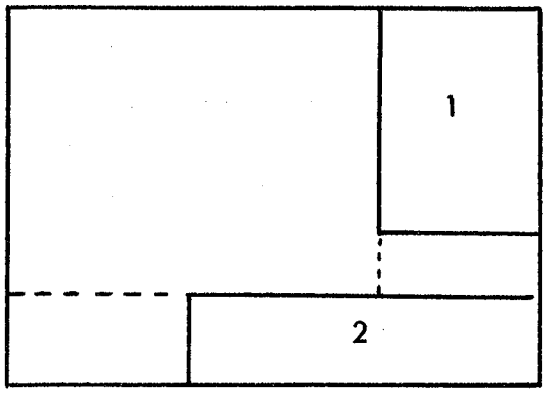
3.3.2. Le processus d'Adaptation

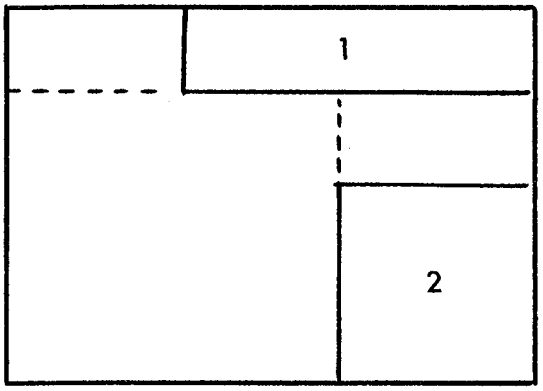
Le processus d'Adaptation a pour rôle de déterminer la découpe optimu d'un objet éditable fixant la représentation d'un objet de base composé. Seule l'occurrence d'un tel objet, dans les requêtes traitées, motive la mise en oeuvre du processus.

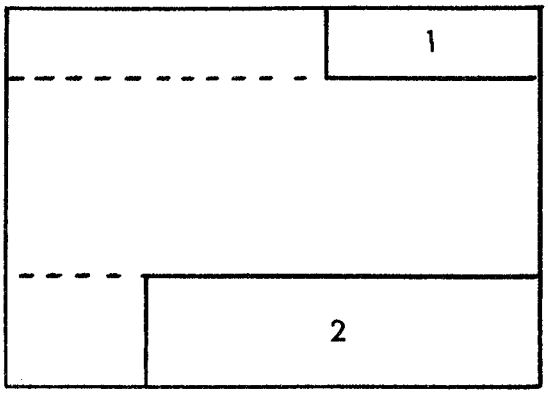
Il exprime le résultat du traitement auquel il procède, sous la forme de valeurs numériques stockées dans le V.I.D. concerné (V.I.D. structure) et destinées à des prises en compte par le Processus de Décision, en premier lieu, puis par les constitutifs du Dispositif d'Impression.

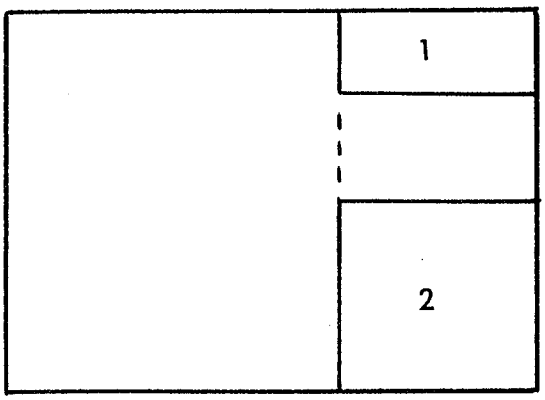
3.3.3. Le Processus de Décision

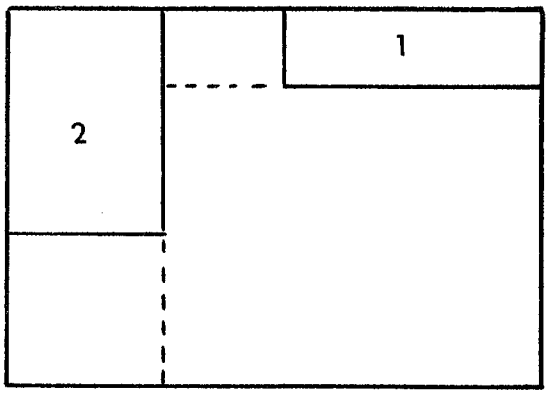
Le processus de Décision, compte tenu des résultats obtenus à l'issue de la mise en oeuvre des deux Processus précités, détermine, conformément aux principes énoncés (Cf.première partie §2.3.5.2.), la solution apparaissant comm la plus favorable. Cette solution, exprimée sous forme paramétrée, est soit

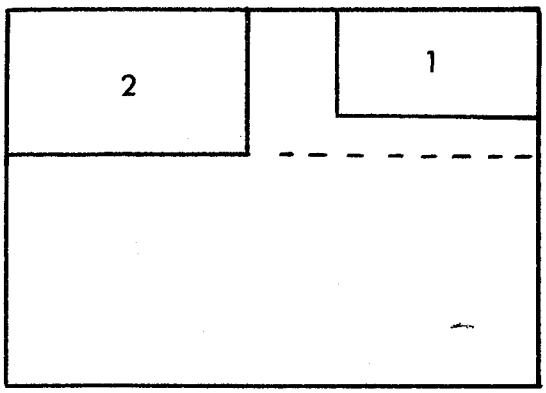


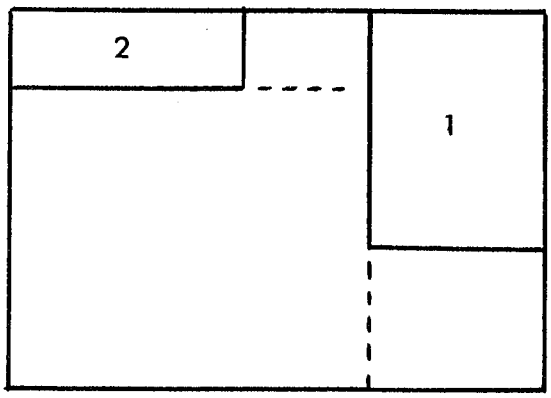












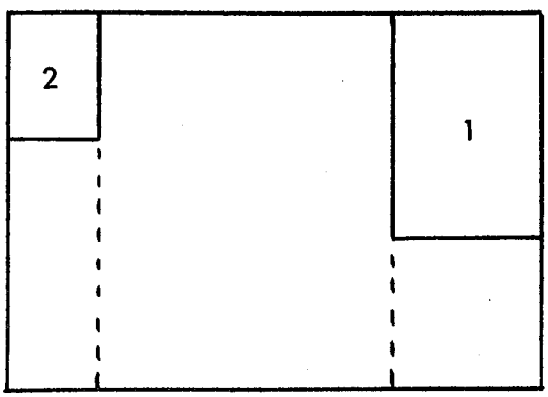


figure 3.3.2.

mise à la disposition du dispositif d'Impression en vue de la génération des imprimés, soit rangée en mémoire centrale. Dans ce dernier cas, elle est consulti par l'utilisateur qui, grâce à l'aspect algorithmique de L.A.M.P.E. entérinera ou non ce choix.

3.4. LE DISPOSITIF D'IMPRESSION

Le Dispositif d'Impression reçoit, en entrée, les mêmes informations que le dispositif de Formation, mais cependant complétées par les résultats des traitements précédents.

Ce Dispositif fournit, en retour, les pages éditées ou à la demande, leur représentation sur support extérieur dans le but d'en différer l'Edition. Pour assumer cette tâche, trois Processus sont mis en oeuvre : respectivement, les Processus d'Evaluation (Cf.§3.3.1.), de Rangement et d'Edition Cette ultime étape dans l'élaboration des imprimés impose aux Dispositifs mis en oeuvre de prendre en considération les requêtes sous la forme que leur a donnée le Programmeur et non sous la forme restructurée.

SCHEMA DE PRINCIPE DU DISPOSITIF D'IMPRESSION

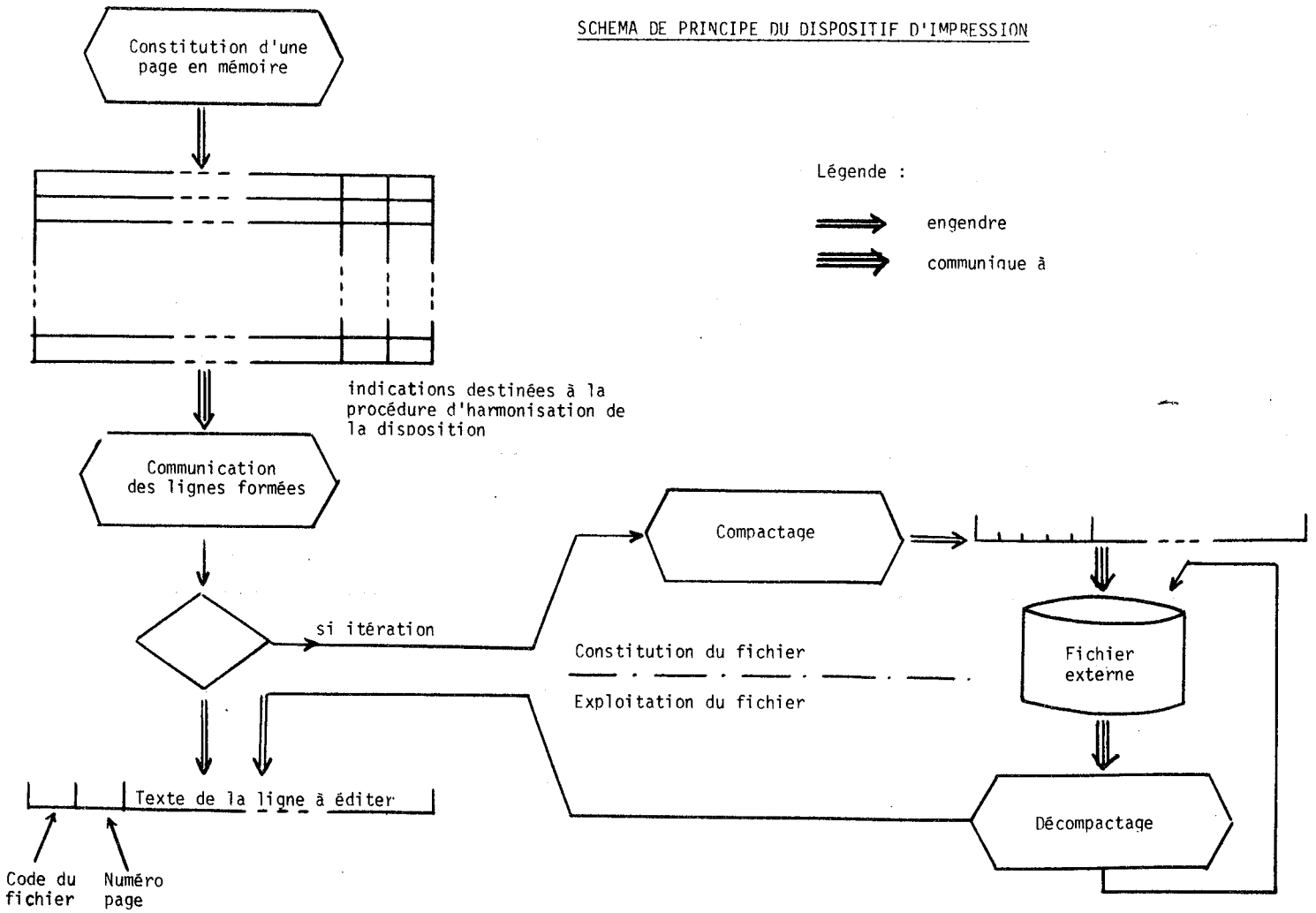


figure 3.3.3.

3.4.1. Le Processus de rangement

Le Processus de Rangement a pour but d'implanter, dans la zone de travail, les informations constitutives des objets désignés.

Ce traitement n'impliquerait pas de commentaires, si la disposition des informations n'était pas influencée par les directives de mise en valeur (Cf.Annexe 6-A-a).

A cet effet, la technique adoptée résulte de la transposition de la notion, bien connue, présidant à l'implantation d'un programme en mémoire centrale. Ainsi, sont successivement calculées les adresses relatives et absolues ou, plus précisément les coordonnées relatives et absolues du Point d'Implantation.

- Les coordonnées relatives s'identifient à celles du curseur.
- Les coordonnées absolues s'identifient aux coordonnées relatives si l'objet n'est pas soumis à des directives de mise en valeur. Dans le cas contraire, il y a incrémentation d'une valeur calculée.

L'adresse absolue, une fois déterminée, l'implantation des informations est immédiate.

3.4.2. Le Processus d'Edition

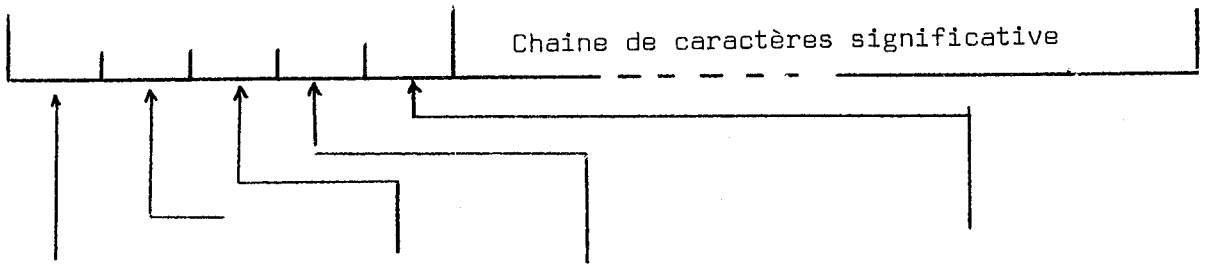
Le Processus d'Edition est mis en oeuvre chaque fois qu'une Page-Physique est constituée. Il a pour but de répartir les zones vides de façon à donner à l'imprimé un aspect plus harmonieux.

A ce stade, la zone de travail est prête à être reproduite sur un support (papier ou magnétique) Etant donné que la zone de travail a été définie comme un tableau PL/1, chaque élément de celui-ci est passé en donnée aux instructions "PUT EDIT" générée lors de l'expansion de l'instruction L.A.M.P.E.(Cf.§2.3.3.5.2.c).

En outre,c'est au processus d'Edition qu'il appartient de prendre en compte la demande de répétition de l'Edition formulée par le programmeur dans l'instruction L.A.M.P.E. Dans cette éventualité, chaque élément des diverses versions du tableau,représentatif de l'Edition, est compacté puis rangé sur un fichier de manoeuvre (voir Figure 3.3.1.).

Les opérations de décompactage et de communication du buffer aux instructions "PUT EDIT" sont alors soumises à itération.

N.B. Constitution d'un enregistrement du fichier de manoeuvre.



Code du fichier support.	numéro de la page.	numéro de la ligne.	nombre de caractéres significatifs.	nombre de "blancs" supprimés à gauche.

3.5. LES REGLAGES

Le système M.P.E. a été conçu, à la fois, pour prouver aux utilisateurs potentiels l'efficacité des méthodes adoptées, et pour parvenir à baser la répartition des informations, non sur des critères empiriques, mais, sur des règles jusqu'ici inconnues.

Dans ce but, l'exploitation de ce produit devrait nous fournir les indications escomptées dont on pourrait immédiatement vérifier la validité, en adaptant les dispositifs à ces nouveaux critères. C'est la raison pour laquelle, les différents processus sont tributaires de paramètres jouant le même rôle que les réglages dont sont pourvues la plupart des réalisations humaines.

C O N C L U S I O N

A l'issue de cet exposé, au cours duquel nous avons essayé de mettre en relief les éléments marquants de notre travail, il nous semble important de prendre un certain recul, de façon à évaluer l'apport de cette réalisation sur le plan pratique et sur le plan théorique.

De par son aspect pratique, ce travail a été profitable à deux titres. Il nous a permis de mettre en application et d'approfondir nos connaissances dans l'implémentation des langages évolués et, plus particulièrement, dans la conception de pré-compilateurs.

Il nous a aussi, et surtout, donné la possibilité d'acquérir une expérience précieuse, dans l'utilisation des performances des ordinateurs actuels, pour améliorer la diffusion des résultats témoignant de l'accomplissement d'un traitement. Il s'agit là d'un problème important, à notre avis trop longtemps et trop injustement délaissé par les Informaticiens; probablement du fait qu'il concerne un domaine marginal (les Entrées/Sorties) et que les résultats obtenus ne sont souvent pas à la mesure des difficultés surmontées.

Néanmoins, dans le cas présent, cette expérience a été et sera, nous l'espérons, d'autant plus riche en enseignements que nous sommes confrontés à un problème réel, imposant, aujourd'hui plus encore qu'hier, des solutions fiables et efficaces. Il suffit, pour se rendre compte de cette nécessité, de prendre conscience de l'anachronisme qui se manifeste parmi les unités périphériques connectées à un ordinateur de grande puissance. C'est ainsi, qu'aux périphériques évolués, synthèse des derniers progrès de la Science et de la Technique, s'opposent notamment les imprimantes. Répondant à une conception dépassée, elles se caractérisent tant par une efficacité relative que par la

consommation déraisonnable de papier à laquelle elles incitent.

La même contradiction qui se manifeste dans le matériel, se ressent également au niveau de son utilisation. Les efforts demandés au programmeur pour réaliser une édition, même simple, étaient, jusqu'à présent, sans commune mesure avec ceux requis pour décrire des algorithmes complexes.

Cette constatation est d'autant plus lourde de conséquences que l'ordinateur attire désormais des adeptes, de formations différentes, ne recherchant en lui qu'un outil efficace et d'emploi aisé.

Ce sont ces considérations qui nous ont amené à préférer, à la mise en service rapide d'une réalisation hybride, la conception, par étapes successives, d'un produit fini et fiable destiné à attirer à lui une large couche d'utilisateurs potentiels.

A cet effet, nous nous sommes, d'une part, appesanti sur l'élaboration et la mise au point d'algorithmes découlant de l'étude théorique entreprise. D'autre part, tenant compte des techniques les plus récentes, nous avons largement fait appel à l'organisation modulaire, dans la conception de nos programmes et avons essayé de doter l'implémentation de notre langage de bases saines et solides; cela,en prévision des évolutions inéluctables qu'impliquera le passage de cette réalisation, de son stade de prototype au stade opérationnel.

De par son aspect théorique ce travail s'est révélé enrichissant et passionnant.

Sur le plan personnel, il nous a donné l'occasion de faire appel à des connaissances théoriques diversifiées et de nous intéresser également à des domaines n'ayant aucun rapport avec l'Informatique, mais, seuls capables de fournir les notions essentielles requises,en raison de l'orientation donnée à l'étude entreprise. Il nous a également permis de connaitre la satisfaction de réussir à surmonter, pas à pas, les difficultés inhérentes à la résolution de tout problème d'analyse; difficultés d'autant plus grandes que nous ne possedions, initialement, que fort peu de renseignements sur les travaux menés, surtout outre Atlantique, dans le domaine de la Mise-en-Pages et de l'Edition par Ordinateur.

Nous fûmes ainsi amené , vu l'étendue du champ d'application :

- à délimiter notre étude en nous intéressant, dans l'immédiat, à la Mise en Pages et à l'Edition de données fournies par un langage largement diffusé (PL/1) et propice aux traitements les plus variés.

- à fixer les étapes des évolutions prévisibles.

- à concevoir, en un premier temps, un prototype dont la mise en application puisse offrir la double perspective de sensibiliser, par les avantages qu'il propose, de nombreux Programmeurs et de nous fournir, en retour, les précieux renseignements conditionnant les améliorations possibles.

Sur le plan général, nous retirons de cette réalisation un certain nombre d'aspects positifs qui se trouvent inclus dans les trois citations suivantes :

- Elaboration d'une nouvelle conception de l'Edition assistée par ordinateur, conférant à ce dernier un rôle important, du fait qu'il lui est confiée la double charge de déterminer la Mise-en-Pages en fonction des directives du Programmeur, et, de constituer les imprimés définitifs.

- Elaboration d'un langage, spécialisé de par les traitements qu'il autorise, algorithmique de par sa faculté de ramener la description d'une Edition à une forme facilement interprétable, évolué de par la nature des concepts sur lesquels il se fonde.

En effet, à l'image des langages de haut niveau classiques, qui libèrent le Programmeur de la prise en considération des caractéristiques physiques du matériel utilisé, L.A.M.P.E. délivre l'utilisateur des contraintes nées de la prise en considération des dimensions des informations et de celles du support.

- Elaboration de techniques de Mise-en-Pages ayant conduit à la conception d'algorithmes satisfaisants, de par leurs résultats et leur faible consommation tant en espace mémoire qu'en temps machine.

Mais de toute réalisation humaine n'émanent rarement que des aspects positifs et il nous est possible de regretter certaines imperfections de notre produit "Langage-Système" :

- Le langage L.A.M.P.E., s'il permet au Programmeur d'exprimer ses désiderata de façon simple et rigoureuse, ne lui offre pas encore la possibilité de prendre toutes les décisions en temps réel.

- Le système M.P.E., comme tous les systèmes asservis, est conçu pour fonctionner dans un contexte défini et dans des conditions précises dont on ne peut s'éloigner sans procéder à des modifications.

- L'unité périphérique utilisée pour l'édition des imprimés demeure, pour l'instant, l'imprimante, alors que l'utilisation de matériels répondant à des techniques plus poussées s'avère souhaitable et prometteuse.

Ce qui a été réalisé nous permet de mieux évaluer ce qui reste à faire, et ce sont ces aspects, moins favorables de notre solution, qui devront en tout premier lieu être neutralisés. Telle est la motivation des étapes que nous envisageons dans l'évolution de notre prototype et dont le but est de doter l'ordinateur de moyens de communication, avec l'homme, dignes de ses performances.

C'est ainsi que les extensions nous paraissant pouvoir être envisagées sont de deux types :

Du point de vue théorique, il nous faudra viser, grâce à la théorie des graphes, à optimiser la répartition des informations, compte tenu des contraintes qui leur sont imposées par l'utilisateur.

A plus longue échéance, tendre , en nous basant sur les techniques présentement élaborées, à définir une théorie Mathématique de la Mise-en-Pages, seule apte à élargir le domaine d'application. Il sera, alors, possible de tenir compte de l'utilisation de différentes polices de caractères (photocomposeuse) et d'accéder à la réalisation des Mises en Pages les plus délicates, en l'occurrence celle des journaux quotidiens.

Du point de vue pratique, il nous apparait nécessaire, dans l'immédiat, de développer l'aspect conversationnel et de faire appel à l'utilisation d'un écran graphique, de façon à ce que les utilisateurs favorisés, disposant de ce matériel, puissent communiquer leurs décisions au Système M.P.E. au vu de résultats probants.

Dès que le stade de prototype sera dépassé, il serait souhaitable, d'une part, de réaliser l'implémentation à l'aide d'un langage de programmation système, de façon à améliorer les performances et, d'autre part, d'adapter L.A.M.P.E. à d'autres langages de haut niveau.

A plus long terme, sans doute faudra-t-il étudier la possibilité de contribuer, grâce à une version plus évoluée de L.A.M.P.E., à améliorer la qualité des imprimés délivrés par un ordinateur et à inciter les utilisateurs à n'éditer que les informations significatives, susceptibles d'être fréquemment consultées. Ce qui contribuera à développer l'utilisation des écrans graphiques et à envisager la commande directe, par le Système M.P.E., d'une photocomposeuse.

B I B L I O G R A P H I E

1- LES TECHNIQUES INFORMATIQUES APPLIQUEES A L'EDITION

[AI-1-1] AVA-INFORMATIQUE

"Esquif-6" : un Editeur d'états et de tableuax.
+ Note descriptive de la Société "AVA Informatique" (Paris

[BE-1-1] BERNS GERALD

"Format" : A text-Processing Program.
+ Communication of the ACM (mars 1969).

[CS-1-1] COMPAGNIE DES SOFTWARE

"Gutenberg" : un programme d'interrogation et d'édition de fichiers.
+ Note descriptive de la "Compagnie des Software" (Paris).

[IB-1-1] IBM

"Hyphenation/360" : System/360 Text Processor.
+ Program Description Manual, IBM (1967)
+ Operation Manual, IBM (1967).

[IB-1-2] IBM

"Text-360"
+ Reference Manual and Operating Guide.
Program N° 360D-29-4-001, IBM (1967).

[IB-1-3] IBM

"Text-processor Program Description Manual"
+ IBM System/360 Operating System H-20-0585-0.

[IB-1-4] IBM

"Report-Writer" Programmer's Guide.
+ IBM System/360 Operating System C28-6399-1.

[IB-1-5] IBM

"CMS Script User's Manual
+ IBM Control Program-67 / Cambridge Monitor System
6H-0860-0.

[IB-1-6] IBM

"R.P.G." Language specifications
+ IBM System/360. Operating System GC-24-3337-5.

[IB-1-7] IBM

"Report Program Generator". Program Logic
+ IBM System 1360. Operating System Y-26-3704-0.

[MM-1-1] MATHEWS & MILE JOAN

Computer Editing, Typesitting and Image Generation.
* AFIPS Conference Proceedings (1965)
(pages 389 à 397)

[PR-1-1] PROFIZI J.C.

Un langage de Mise_en_page et d'Edition. Etude de son implémentation.
* Projet de D.E.A. (I.M.A.G. Octobre 70)

[PR-1-2] PROFIZI J.C. & SEGUIN J.

"L.A.M.P.E.": Langage Algorithmique de Mise en Page et d'Edit
* Centre Scientifique IBM/Université de Grenoble.
Etude n°FF2.0137 (Avril 72)

[PR-1-3] PROFIZI J.C.

"M.P.E." Un Système de Mise en Page et d'Edition,
"L.A.M.P.E." Son Langage de Commande.
* R.A.I.R.O. Numéro: B-2 (Juin 73)

[TR-1-1] TRABAND P.

La composition programmée à l'Imprimerie Nationale.
* Article "Informatique et Gestion" (N°30)

[VE-1-1] VERGES A.C.

L'édition de textes basée sur la reconnaissance automatique de leurs structures.
* Canadian Computer Conference (Session 72)

[WA-1-1] WALTER GERALD

Electronic Typesitting Systems promise to have a revolutionary effect on the printing industry.
* Scientific American 220 (Mai 69)

[WB-1-1] WILLIAMS D. & BARTRAN PHILIP

"Compose-Produce": A user oriented report generator capability within the S.D.C. Time Shared Data Management.
* AFIPS Conference Proceedings (1967)
(Volume 30 pages 635 à 640)

2- LES "SYSTEMES" SPECIALISES IMPLIQUANT LA CONCEPTION D'UN LANGAGE DE COMMANDE APPROPRIE.

[BF-2-1] BRACCHI G. & FERRARI D.

A Language for Treating Geometric Patterns in a two dimensione space.
* Communication of the A.C.M. (Janvier 71)

[CK-2-1] CARDENAS ALFONSO & RARPLUS WALTER

A Language for Partial Differential Equation.
* Communication of the A.C.M. (Mars 70)

[LA-2-1] LAPLACE A.

"PL/1-FORMAC" Tome 1: Le langage.
* Publication I.R.M.A. Grenoble (1972)

[LA-2-2] LAPLACE A.

"FORmac DEsk CALculator": un outil de mise au point et d'aide au calcul formel sur ordinateur.
* Thèse de Doctorat d'Etat. Grenoble (Février 73)

[LA-2-1] LECARME O.

"Un système de programmation graphique conversationnelle".
* Thèse de Doctorat d'Etat. Grenoble (Septembre 70)

[SI-2-1] SIRET Y.

"Contribution au calcul formel sur ordinateur".
* Thèse de Doctorat d'Etat. Grenoble (Juillet 70)

3- OUVRAGES ET ARTICLES D'INTERET TECHNIQUE

[BE-3-1] BERTHAUD M.

Variable BASED et tâches en PL/1.
+ Etude n° FF2.0104.IBM Centre Scientifique Grenoble.

[GR-3-1] GRIFFITHS M.

Analyse déterministe et Compilateurs.
+ Thèse de Doctorat d'Etat. Grenoble (Octobre 69).

[GT-3-1] GRIFFITHS & TASSART

Discrimination of Key-words in PL/1
+ Note interne I.M.A.G.

[IB-3-1] IBM

PL/1(F) Language Reference Manual.
+ IBM System/360 - Operating System C.28-8201-3.

[IB-3-2] IBM

PL/1(F) Programmer's Guide.
+ IBM System/360. Operating System C.28-6594-8.

[IB-3-3] IBM

PL/1(F) Subroutine Library. Program Logic Manual.
+ IBM System/360. Operating System Y-28-6801-5.

[IB-3-4] IBM

PL/1(F) Compiler Program Logic Manual.
+ IBM System/360. Opertaing System GY-28-6800-4.

[IB-3-5] IBM

PL/1(F) Language Specifications.
+ IBM System/360. Order Number GY-33-6003-2.

[IB-3-6] IBM

Job Control Language.
+ IBM System/360. Operating System C-38-6539-7.

[IB-3-7] IBM

Supervisor and Data Management Services.
+ IBM System/360. Operating System C-28-6646.

[IB-3-8] IBM

Scientific Subroutine Package (PL/1).
+ IBM System/360. Operating System GH-20-0586-0.

[KN-3-1] KNUTH

The Art of Comuter Programming (volume 1).
+ Addison-Wesley (Editeur) (1968).

[LA-3-1] LAPLACE

"PL/1-FORMAC" tome 2 : Le pré-processeur.
+ Publication I.R.M.A. Grenoble (1972).

[SE-3-1] SEGUIN J.

"Compile Time Facilities PL/1". Extension de PL/1.
+ Publication I.M.A.G. Grenoble (Mars 1970).

[UR-3-1] URSCLER G.

"Concrete Syntas of PL/1".
+ IBM Laboratory Vienna. TR-25-096 (Juin 1969).

[WI-3-1] WILLIS B.

PL/1 in an extensible language environment.
+ Publication I.+.A.G. Grenoble (Sept. 1969).

[WI-3-2] WILLIS B.

A base language for PL/1.
+ Publication I.M.A.G. Grenoble (Avril 1970).

4- OUVRAGES ET ARTICLES D'INTERET GENERAL

[AR-4-1] ARSAC J.

La Science Informatique.
* Edition Dunod - Paris (1970)

[BA-4-1] BARGILLIAT

L'imprimerie au XX^{ème} Siècle.
* Presses Universitaires de France. Paris (1967).

[FR-4-1] FRANC G.

Comparaison des Software généraux de gestion.
* Article : "Informatique et Gestion" (Juin 71)

[GR-4-1] GRIES DAVID

Compiler Construction for Digital Computer.
* Edition John Wiley. New-York (1971)

[LA-4-1] LABORDERIE (DE)

Toute l'imprimerie.
* Edition Dunod. Paris (1970)

[LE-4-1] LECARME O.

Etude comparative des principaux langages de programmation.
* Thèse de Doctorat de 3e Cycle Grenoble (1965)

[LE-4-2] LECARME O.

Niveau de langages et puissance des outils.
* R.A.I.R.O. (Juin 70).

[MO-4-1] MOLES ABRAHAM

Art et Ordinateur.
* Edition Casterman (1971)

[PE-4-1] PECCOUD, DELOBEL, ARCHER

Contribution au développement de méthodes d'analyse des systèmes d'information.
* Publication I.M.A.G. (Juillet 69)
Contrat D.G.R.S.T. n°67-01-015.

ANNEXE-1

POSSIBILITES COMPAREES DE PL/1 ET DE L.A.M.P.E.

/* SOIT LA DECLARATION SUIVANTE : */

```
DCL   (*,*,*)  CONTROLLE ) ;
DCL   TABLE(3,4,2)  INIT  (' ... ') ;
```

Supposons que l'on désire représenter ce tableau de la façon suivante :

		COL-1	COL-2	COL-3	COL-4
COU-1	LIG-1				
	LIG-2				
	LIG-3				
COU-2	LIG-1				
	LIG-2				
	LIG-3				

/* EN ADMETTANT QUE LA REPRESENTATION DU TABLEAU SOIT INSCRIPTIBLE DANS UNE PAGE DE LISTING, LA CODIFICATION DE LA MAQUETTE PREALABLEMENT ETABLIE PERMETTRA LE PROGRAMME PL/1 SUIVANT : */

```
DCL   ZONE   CHAR(4) ;
PUT   EDIT   (('  !  ',' COL-',N)  DO  N = 1  TO  4 )
             (X(15), A(A(3),A(4),A(1),SKIP(1)) ;
PUT   EDIT   ('-' DO  I = 1  TO  41 ) ( 41 A(1),  SKIP ) ;
   I = 1  TO  2 ;  /*  I = COUCHES */
      J = 1  TO  3 ;  /*  J = LIGNES  */
      IF  J - ( 2 * ( J/2) ) = 0  THEN  ZONE  =  'COU' ;
                                  ELSE  ZONE  =  '    ' ;
      PUT  EDIT  ( ZONE , 1 , ' * LIG-' , J , ' ! ' , ( TABLE(J,K,I)
                 DO K = 1 TO 4 )  ( R ( MASQ1 )) ;
   ;
PUT   EDIT  ( '-' DO  I = 1  TO  41 ) ( 41  A(1) , SKIP ) ;
;
MASQ1 : FORMAT  (X(2) , A(4) , A(1) , A(7) , A(1) , A(3) , 2((F(10,3)), ' , ')
                F(10,3)), SKIP) ;
```

```
/* REALISATION DE LA MEME EDITION EN L.A.M.P.E. */

DCL   AADJOINDRE (I1,I2,I3)  MODELE , (I1,I2,I3)  EXTERIEUR ;
      1  'COU-'  (I3) ,
         2  'COL-'  (I2) ,
            3  'LIG-'  (I1)  (F(10,3)) ;
FIN :

DCL   AEDITER  REPRESENTATION  TABULE  (TABLE(*,*,*)) ;
      ASSOCIER  AADJOINDRE  (DIM(TABLE,1) , DIM(TABLE,2) , DIM (TABLE,3)) ;
      IMPLANTER(HAUT)  SEPARER('.')  AADJOINDRE.2 ;
      IMPLANTER(GAUCHE)
         SEPARER  AADJOINDRE.1  ET
         SEPARER('*')  DISPOSER  SEPARER  AADJOINDRE.3 ;
FIN  ;

PUT  LAMPE  (AEDITER) ;  /* L'EDITION SERA REALISEE QUE LE TABLEAU, DONT
                            ON A FIXE L'ASPECT, SOIT OU NON INSCRIP-
                            TIBLE DANS LA PAGE DE LISTING */

/* CITONS QUELQUES POSSIBILITES ANNEXES */

      /* EDITION , SANS AUCUNE MODIFICATION, SUR DES PAGES DE DIMENSIONS
         DIFFERENTES */

      PUT  LAMPE  (AEDITER)  (CARAC(50) , LIGNE(40)) ;

      /* EDITION, SANS AUCUNE MODIFICATION, QUELQUE SOIT LE VOLUME AFFECTE
         DYNAMIQUEMENT AU TABLEAU PL/1 */

      ALLOCATE  TABLE(10,5,3)  INIT(' ... ') ;
      PUT  LAMPE  (AEDITER) ;

/* INCLUSION DE CE TABLEAU DANS DES IMPRIMES REGROUPANT D'AUTRES INFORMATIONS
   JOINTES */

DCL   ENSEMBLE  PROPOSITION  FIXE ;
      DISPOSER  'VOICI UN EXEMPLE D'EDITION SIMPLE DECRITE A L'AIDE DE
                 L.A.M.P.E.'  PUIS  CENTRER  AEDITER  ;

PUT  LAMPE  (ENSEMBLE) ;
```

ANNEXE -2

QUELQUES APPLICATIONS DE L.A.M.P.E. A LA REPRESENTATION DE DONNEES.

Le tableau PL/I ci-contre, est supposé établir le bilan annuel des recettes et des dépenses d'une entreprises. Il peut se préter à diverses représentations. Les figures I, 2 et 3, en sont, à titre d'exemple, un témoignage.

NB : Aux seules fins de rendre l'application plus parlante, les données numériques constitutives du tableau PL/I sont remplacées par des lettres de l'alphabet.

A/ Représentation sans traits de séparation entre les sous-ensembles

		TRIMESTRE-I	TRIMESTRE-2	TRIMESTRE-3	TRIMESTRE-4
	DEPENSES	a	b	c	d
MOIS-I	RECETTES	a'	b'	c'	d'
	DEPENSES	e	f	g	h
	RECETTES	e'	f'	g'	h'
	DEPENSES	i	j	k	l
	RECETTES	i'	j'	k'	l'

Figure - I

TRIMESTRE-I			TRIMESTRE-2			TRIMESTRE-3			TRIMESTRE-4			
MOIS-I	MOIS-2	MOIS-3	MOIS-I	MOIS-2	MOIS-3	MOIS-I	MOIS-2	MOIS-3	MOIS-I	MOIS-2	MOIS-3	
e	i		b	f	j	c	g	k	d	h	l	DEPENSE
e'	i'		b'	f'	j'	c'	g'	k'	d'	h'	l'	RECETTE

Figure - 2

TRIMESTRE-I	MOIS-I		MOIS-2		MOIS-3	
	DEPENSES	a	DEPENSES	e	DEPENSES	i
	RECETTES	a'	RECETTES	e'	RECETTES	i'
TRIMESTRE-2	MOIS-I		MOIS-2		MOIS-2	
	DEPENSES	b	DEPENSES	f	DEPENSES	j
	RECETTES	b'	RECETTES	f'	RECETTES	j'

Figure - 3

B/ Représentation avec traits de séparation entre les sous-ensemble

TRIMESTRE-I			TRIMESTRE-2			TRIMESTRE-3			TRIMESTRE-4			
MOIS-I	MOIS-2	MOIS-3	MOIS-I	MOIS-2	MOIS-3	MOIS-I	MOIS-2	MOIS-3	MOIS-I	MOIS-2	MOIS-3	
a	e	i	b	f	j	c	g	k	d	h	l	DEPENS
a	e	i	b	f	j	c	g	k	d	h	l	RECETT

Figure-4

TRIMESTRE-I	MOIS-I		MOIS-2		MOIS-3	
	DEPENSES	a	DEPENSES	e	DEPENSES	i
	RECETTES	a	RECETTES	e	RECETTES	i
TRIMESTRE-2	MOIS-I		MOIS-2		MOIS-3	
	DEPENSES	b	DEPENSES	f	DEPENSES	j
	RECETTES	b	RECETTES	f	RECETTES	j

Figure - 5

Le traitements suivant est à l'origine des cinq Représentations pré-citées.

```
DCL  BILAN(3,4,2)  FIXED ;
 .
 .
 .       Traitement PL/I initialisant le tableau 'BILAN'
 .
 .
DCL  AADJOINDRE(M,N)  MODELE,  (M,N) EXTERIEUR ;
     I ('DEPENSES' , 'RECETTES') ,
       2 'TRIMESTRE-' (N) ,
         3 'MOIS-' (M)  F(5,2) ;
FIN .
```

C/ Description des diverses Représentations.

```
DCL  FIGURE I  REPRESENTATION  TABULE  (BILAN(*,*,*)) ;
     ASSOCIER  AADJOINDRE(3,4) ;
     IMPLANTER(GAUCHE)  AADJOINDRE.3  ET  AADJOINDRE.I  ;
     IMPLANTER(HAUT)  AADJOINDRE.2  ;
FIN  FIGURE_I ;

DCL  FIGURE_2  REPRESENTATION  TABULE  (BILAN(*,*,*)) ;
     ASSOCIER  AADJOINDRE(3,4) ;
     IMPLANTER(HAUT)  AADJOINDRE.2  PUIS  AADJOINDRE.3  ;
     IMPLANTER(DROITE)  AADJOINDRE.I  ;
FIN  FIGURE_2 ;

DCL  FIGURE_3  REPRESENTATION  ARBITRAIRE  (BILAN(I:3,I:2,*)) ;
     ASSOCIER  AADJOINDRE ;
     ENUMERER  AADJOINDRE.2  FIN  ET  ALIGNER  AADJOINDRE.3  FIN  PUIS
     ENUMERER  AADJOINDRE.I ;
FIN  FIGURE_3 ;

DCL  FIGURE_4  REPRESENTATION  TABULE  (BILAN(*,*,*)) ;
     ASSOCIER  AADJOINDRE(3,4) ;
     IMPLANTER(HAUT)  SEPARER('*')  AADJOINDRE.2  FIN  PUIS
                      SEPARER('-')  DISPOSER  SEPARER('.')  AADJOINDRE.3 ;
     IMPLANTER(DROITE)  SEPARER  AADJOINDRE.I ;
FIN  FIGURE-4 ;

DCL  FIGURE_5  REPRESENTATION  ARBITRAIRE  (BILAN(I:3,I:2,*)) ;
     ASSOCIER  AADJOINDRE(3,2) ;
     ENUMERER  SEPARER  AADJOINDRE.2  FIN  FIN  ET
     SEPARER('*')  ALIGNER  AADJOINDRE.3  FIN  FIN  PUIS
     ENUMERER  AADJOINDRE.I ;
FIN  FIGURE_5 ;
```

ANNEXE-3

APTITUDES DE PL/I-L.A.M.P.E. A LA RESOLUTION D'UN PROBLEME D'EDITION.

EXEMPLE GENERAL

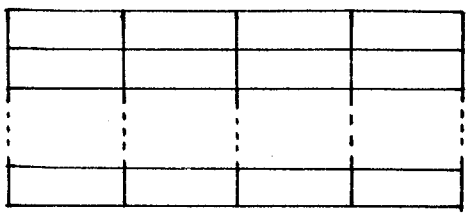
Supposons qu'un club de discophiles impose, à ses adhérents, de choisir mensuellement quatre disques, sur le catalogue qui leur est régulièrement adressé.

Au terme de chaque mois, les adhérents reçoivent, individuellement, leur facture accompagnée d'une liste des disques appartenant à la série qui les interesse et proposés à des prix exceptionnels.

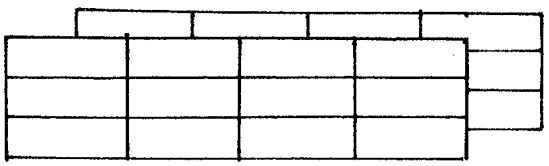
Admettons que le traitement PL/I nous fournisse, de façon itérative, pou chaque adhérents pris en compte, des renseignements dont certains sont répartis en deux tableaux :

- l'un dénommé "COMMANDES", de dimensions variables puisque récapitulant les commandes passées dans le mois échu,
- l'autre denommé "AVANTAGEUX", de dimensions fixes puisque regroupant les disques proposés à des conditions intéressantes.

COMMANDES



AVANTAGEUX



```
/* TRAITEMENT CONDUISANT A L'EDITION DES IMPRIMES ET ASSURE GRACE AU LANGAGE
   PL/I-L.A.M.P.E. */

          .
          .
          .        Traitement PL/I
          .
          .

DCL (NOM , PRENOM , RUE , VILLE , PAYS)  CHAR(25)  VARYING ; DCL ..... ;
DCL COMMANDES(N,4)  CHAR(4)  CONTROLLED ; DCL  AVANTAGEUX(3,4,2)  CHAR(4) ;
CLIENT_SUIVANT :  ......
                    .
                    .   Traitement PL/I assurant, entre autre, l'initialisation
                    .
                    .   des "objets" précédemment déclaré.
```

```
/* DECLARATIONS DES REPRESENTATIONS RELATIVES AUX VARIABLES COMPOSEES */

DCL  SPECIF(J)  MODELE,  J  EXTERIEUR ;
     I  ('REFERENCE' , 'PRIX' , 'NOMBRE' , 'MONTANT') ,
        2  'DISQUE-'(J)  A(4) ;
     FIN ;
DCL  TAB_COM  REPRESENTATION  TABULE  (COMMANDES(*,*,*)) ;
     ASSOCIER  SPECIF(N) ;
     IMPLANTER(HAUT)  SEPARER('.')  SPECIF.I ;
     IMPLANTER(GAUCHE)  SEPARER  SPECIF.2 ;
     FIN ;
DCL  COMPLEMENT  MODELE ;
     I  ('33 - TOURS' , '45 - TOURS'),
        2  ('DISPONIBLE' , 'DELAI - 8J' , 'DELAI - 15J' , 'DELAI - 21J'),
           3  ('*' , '* *' , '* * *')  a(4) ;
     FIN ;
DCL  TAB_AVA  REPRESENTATION  TABULE  (AVANTAGEUX(*,*,*)) ;
     ASSOCIER  COMPLEMENT ;
     IMPLANTER(GAUCHE)  SEPARER('.')  COMPLEMENT.I  ET
                                      SEPARER  DISPOSER  SEPARER  COMPLEMENT.3 ;
     IMPLANTER(HAUT)  SEPARER('.')  COMPLEMENT.2 ;
     FIN ;

/* DECLARATIONS DES "COMPOSITIONS" */

DCL  LISTE  PROPOSITION  FIXE ;
     DISPOSER  CENTRER  'LISTE DES DISQUES COMMANDES'  FIN  PUIS
               INSERER  DELIMITER  TAB_COM ;
     FIN ;
DCL  AFFAIRES  PROPOSITION  FIXE ;
     DISPOSER  'LISTE DES DISQUES OFFERTS A DES CONDITIONS EXCEPTIONNELLES'  PUIS
               INSERER  DELIMITER  TAB_AVA ;
     FIN ;
DCL  IDENTITE  PROPOSITION  FIXE ;
     DISPOSER  NOM  ET  PRENOM ;
     FIN ;
DCL  ADRESSE  PROPOSITION  FIXE ;
     DISPOSER  'ADRESSE'  PUIS
               DECALER  DISPOSER  NUMERO (F(3))  ET RUE  FIN
                        PUIS  VILLE  ET  PAYS ;
     FIN ;
```

```
DCL  MONTANT  PROPOSITION  FIXE ;
     DISPOSER  'MONTANT DES ACHATS :'  ET  TOTAL  ET  'FRANCS' ;
     FIN ;
DCL  NUM_CLIENT  PROPOSITION  FIXE ;
     DISPOSER 'NUMERO-CLIENT :'  ET  CODE ;
     FIN ;
DCL  NOTA_BENE  PROPOSITION  FIXE ;
    DISPOSER  'LE NOMBRE D'ASTERISQUES FIGURANT DANS LE TABLEAU, CI DESSUS',
              EST PROPORTIONNEL A L'IMPORTANCE DES REMISES CONSENTIES' ;
     FIN ;
DCL  ENTETE(SUIVANT)  PROPOSITION  FIXE , SUIVANT  EXTERIEUR ;
     DISPOSER  'PROPOSITIONS INTERESSANTES POUR LE MOIS DE '  ET  SUIVANT ;
     FIN ;
DCL  DATE(J,M,A)  PROPOSITION  FIXE ,  (J,M,A) EXTERIEUR ;
     DISPOSER  J  ET  '/'  ET  M  ET  '/'  ET  A ;
     FIN ;

/* DECLARATIONS DES "IMPLANTATIONS" */

DCL  DES_FAC(MOIS)  PROPOSITION  LIBRE ,  MOIS  EXTERIEUR ;
     IMPLANTER(TITRE)  CENTRER  'FACTURE DU MOIS DE'  ET  MOIS ;
     IMPLANTER(PRIORITE)  INSERER  DELIMITER  DISPOSER  IDENTITE  PUIS  ADRESSE ;
     IMPLANTER(BAS)  MONTANT ;
     DISPOSER  NUM_CLIENT ;
     DISPOSER  DISPOSER  'DATE DE COMMANDE :'  ET  DATE(JOUR_C,MOIS_C,AN_C)  FIN
               PUIS  'DATE D'EXPEDITION :'  ET  DATE(JOUR_E,MOIS_E,AN_E) ;
     DISPOSER  LISTE ;
     FIN ;
DCL  DES_EXCE(PERIODE)  PROPOSITION  LIBRE , PERIODE  EXTERIEUR ;
     IMPLANTER(TITRE)  ENTETE(PERIODE) ;
     IMPLANTER(HAUT)  SOULIGNER  IDENTITE  FIN  ET  ADRESSE ;
     DISPOSER  NUM_CLIENT ;  DISPOSER  AFFAIRES ;  DISPOSER  NOTA_BENE ;
     FIN ;

/* DECLARATIONS  DES  "REPARTITIONS" */

DCL  FACTURES(MOIS)  PROPOSITION  RESTRICTIVE  (DES_FAC),  MOIS  EXTERIEUR ;
     DISPOSER  IDENTITE  AVEC  ADRESSE  AVEC  LISTE  AVEC  MONTANT  AVEC
               'FACTURE DU MOIS DE '  AVEC  MOIS ;
     FIN ;
```

```
DCL  AVANTAGES(PERIODE)  PROPOSITION  RESTRICTIVE  (DES_FAC) ,  MOIS EXTERIEUR ;
     DISPOSER  IDENTITE  AVEC  AFFAIRES  AVEC  NOTA_BENE  AVEC  ENTETE(PERIODE)
               AVEC  ADRESSE ;
     FIN ;

/* ALGORITHME DE TRAITEMENT */

IF  CLIENT_CONCERNE  =  I :
THEN  IF  COMPATIBILITE  (FACTURES(MOIS_COURANT))  (CARACTERE(30),LIGNE(50))
      THEN  BEGIN ;
            PUT  LAMPE  (FACTURES(MOIS_COURANT))  (CARACTERE(30),LIGNE(50)) ;
            GOTO  SUITE ;  END ;
      ELSE  IF  COMPATIBILITE (FACTURES(MOIS_COURANT)) (CARACTERE(100),LIGNE(50))
            THEN  BEGIN ;
                  PUT  LAMPE  (FACTURES(MOIS_COURANT)) (CARACTERE(100, PAS(10)),
                               HAUTEUR(50)) ; GOTO  SUITE ;  END ;
            ELSE  .
                  .
                  .    Traitement PL/I.
                  .
                  GOTO  CLIENT_SUIVANT ;
SUITE : IF  COMPATIBILITE  (AVANTAGES(MOIS_SUIVANT))  (CARACTERE(100),LIGNE(50))
        THEN  BEGIN ;
              PUT  LAMPE  (AVANTAGES(MOIS_SUIVANT))  (CARACTERE(100),LIGNE(50))
              GOTO  CLIENT_SUIVANT ;  END ;
        ELSE  PUT  LAMPE  (DES_EXCE(MOIS_SUIVANT))  (CARACTERE(100),LIGNE(50))
GOTO  CLIENT_SUIVANT ;
```

FACTURE DU MOIS DE JUIN

DUPONT JEAN
ADRESSE
3 BD. JOFFRE
CARPENTRAS FRANCE

NUMERO-CLIENT : 35036

DATE DE COMMANDE : 06/06/72
DATE D'EXPEDITION : 10/06/72

MONTANT DES ACHATS : 152 FRANCS

FACTURE DU MOIS DE JUIN

DUPONT JEAN
ADRESSE
3 BD. JOFFRE
CARPENTRAS FRANCE

LISTE DES DISQUES COMMANDES

	REFERENCE	PRIX	NOMBRE	MONTANT
DISQUE - 1				
DISQUE - 2				
DISQUE - 3				
DISQUE - 4				
DISQUE - 5				
DISQUE - 6				
DISQUE - 7				

MONTANT DES ACHATS : 152 FRANCS

PROPOSITIONS INTERESSANTES POUR LE MOIS DE JUILLET

DUPONT JEAN ADRESSE
3 BD. JOFFRE
CARPENTRAS FRANCE

NUMERO-CLIENT : 35036

LISTE DES DISQUES OFFERTS A DES CONDITIONS EXCEPTIONNELLES

		DISPONIBLE	DELAI - 8J
33 - TOURS	*		
	**		
	***		
45 - TOURS	*		
	**		
	***		

LE NOMBRE D'ASTERISQUES FIGURANT DANS LE TABLEAU, CI-DESSUS, EST PROPORTIONNEL A L'IMPORTANCE DES REMISES CONSENTIES

PROPOSITIONS INTERESSANTES POUR LE MOIS DE JUILLET

DUPONT JEAN ADRESSE
3 BD. JOFFRE
CARPENTRAS FRANCE

NUMERO-CLIENT : 35036

LISTE DES DISQUES OFFERTS A DES CONDITIONS EXCEPTIONNELLES

		DELAI - 15J	DELAI - 21J
33 - TOURS	*		
	**		
	***		
45 - TOURS	*		
	**		
	***		

LE NOMBRE D'ASTERISQUES FIGURANT DANS LE TABLEAU, CI-DESSUS, EST PROPORTIONNEL A L'IMPORTANCE DES REMISES CONSENTIES

ANNEXE-4

SYNTAXE DE L.A.M.P.E.

PROGRAMME → {ETIQUETTE}* PROCEDURE OPTIONS (MAIN) ;
CORPS - PROCEDURE END ;

CORPS_PROCEDURE → {DECLARATION PL/1 | DECLARATION L.A.M.P.E.|
INSTRUCTION - PL/1 | INSTRUCTION L.A.M.P.E.}*

DECLARATION PL/1 → cf bibliographie [UR-3-1]

INSTRUCTION PL/1 → cf bibliographie [UR-3-1]

DECLARATION L.A.M.P.E. → DECLARE ENTETE-DECLARATION CORPS-DECLARATION
DELIMITEUR-DECLARATION

ENTETE DECLARATION → IDENTIFICATION CARACTERISTIQUES [SPECIFICATION
[, SPECIFICATION]*] ;

IDENTIFICATION → IDENTIFICATEUR [(PARAMETRE [, PARAMETRE]*)]

PARAMETRE → IDENTIFICATEUR

CARACTERISTIQUES → MODE [ATTRIBUT] [(REFERENCE-OBJET)]

MODE → {FICHIER | MODELE | REPRESENTATION | PROPOSITION | EXTERIEU

ATTRIBUT → {FIXE | LIBRE | RESTRICTIF | LINEAIRE | TABULE |
STRUCTURE | ARBITRAIRE | PERMANENT}

REFERENCE-OBJET → IDENTIFICATEUR [PRECISION]

PRECISION → ({ARGUMENTS | MODELE-DONNEES | SELECTIONS}) |
. {IDENTIFICATEUR | CONST-ENTIERE}

ARGUMENTS → {LITTERAL | IDENTIFICATEUR} [, {LITTERAL |
IDENTIFICATEUR}]*

MODELE-DONNEES → DATA-FORMAT (cf bibliographie [UR-3-1])

SELECTIONS → SELECTION [, SELECTION]*

SELECTION → {REPERE | (REPERE {,REPERE}*) }

REPERE → [BORNE :] INDICE

BORNE → SCALAIRE

SCALAIRE → {CONST-ENTIERE | VARIABLE}

VARIABLE → IDENTIFICATEUR

INDICE → BORNE | *

SPECIFICATIONS → {IDENTIFICATION | (IDENTIFICATION {, IDENTIFICATION}*) M

CORPS-DECLARATION → {{REQUETE}* | FORME-STRUCTUREE }

REQUETE → COMMANDE ;

COMMANDE → [ETIQUETTE] DIRECTIVE OPERANDES [DELIMITEUR-COMMANDE]

ETIQUETTE → IDENTIFICATEUR :

DIRECTIVE → {{DISPOSER | CENTRER | DECALER | ALIGNER | ENUMERER }
|{SOULIGNER | DELIMITER | INSERER | IMPLANTER |
SEPARER} [RESTRICTION]}

RESTRICTION → ({CONST-CHAINE-CARACTERES | INDIC-POSITION [,INDIC-POSITION

INDIC-POSITION → {HAUT | BAS | DROITE | GAUCHE | ENTETE | PRIORITE}

OPERANDES → OPERANDE [OPE-MISE-RELATION OPERANDE]*

OPERANDE → REFERENCE-OBJET | LITTERAL | COMMANDE

OPE-MISE-RELATION → { ET | PUIS | AVEC }

DELIMITEUR-COMMANDE → FIN [IDENTIFICATEUR]

FORME-STRUCTUREE → CONST-ENTIERE ELEMENTS

ELEMENTS → [(DIMENSIONS)] RENSEIGNEMENTS

DIMENSIONS → DIMENSION [, DIMENSION]*

DIMENSION → {(SELECTIONS) | [BORNE :] BORNE}

RENSEIGNEMENTS → {QUALIFICATION [MODELE-DONNEES] | MODELE-DONNEES
[QUALIFICATION] | TYPE}

QUALIFICATION → CONST-CHAINE-CARACTERES [(DIMENSIONS) [CONST-
CHAINE-CARACTERES]]

TYPE → DATA-ATTRIBUTE PL/1 (cf bibliographie [UR-3-1])

DELIMITEUR-DECLARATION → FIN [IDENTIFICATEUR] ;

INSTRUCTION-L.A.M.P.E. → PUT ENTETE-INSTRUCTION CORPS-INSTRUCTION ;

ENTETE-INSTRUCTION → [(FILE IDENTIFICATEUR)] LAMPE

CORPS-INSTRUCTION → (REFERENCE-OBJET) [(MODALITE-EDITION [, MODALITE-EDITION]$^{0|\cdots|8}$)]

MODALITE-EDITION → {SERIALISATION | ITERATION (SCALAIRE) | CONTRAINTES-DIM}

CONTRAINTES-DIM → {CARACTERE | LIGNE} (SCALAIRE) [, PAS (SCALAIRE) [, A (SCALAIRE)]]

SYNTAXE COMMUNE A PL/1 ET A L.A.M.P.E.

LITTERAL → {CONST_CHAINE_CARACTERES | CONST_ARITHMETIQUE}

CONST_CHAINE_CARACTERE → {CARACTERE }*

CARACTERE → {LETTRE |CHIFFRE |CARACTERE_SPECIAL |AUTRE CARACTERE EBCDIC}

CONST_ARITHMETIQUE → CONST_REELLE [I]

CONST_REELLE → {[+|-] CONST_DECIMALE | CONST_BINAIRE}

CONST_DECIMALE → CONST_ENTIERE[.[CONST_ENTIERE]][E[+|-]CONST_ENTIERE]

CONST_ENTIERE → [CHIFFRE]$^{1|\cdots|10}$

CONST_BINAIRE → {BIT}*[.{BIT}*][E[+|-] CONST_ENTIERE]

IDENTIFICATEUR → LETTRE [LETTRE | CHIFFRE | _]$^{0|\cdots|30}$

ANNEXE-5

EXPANSION J.C.L.

```
//        EXEC    PL1FPCLG
//SYSIN   DD  *
                  .
                  .     PROGRAMME-SOURCE
                  .
/*
```

Les quelques instructions de J.C.L. précédentes, qui suffisent à provoquer le traitement de tout programme PL/1-L.A.M.P.E., mettent en oeuvre la séquence mentionnée sur la page suivante et représentative de l'une des sept expansions figurant dans la "Proclib".

```
//*
//* MISE EN OEUVRE SOIT DE LA PRE-COMPILATION SOIT DE LA ROUTINE ASSURANT
//* EXCLUSIVEMENT LA RECONNAISSANCE DES INSTRUCTIONS L.A.M.P.E. SELON LE
//* PARAMETRE MENTIONNE DANS PRETRA
//*
//PRETRA   EXEC   PGM=IELAMPE,PARM=&P
//SYSPRESPC DD    DSN=&RESPC,DISP=(MOD,PASS),UNIT=PDISK,
//                SPACE=(1350,(60,50),,CONTIG),DCB=BLKSIZE=1350
//*
//*TEST DU TEXTE PRE_COMPILE
//*
//TEST     EXEC   PGM=IERRCOOO,COND(8,LT,PRETRA)
//SYSOUT   DD     UNIT=(CTC,,DEFER),LABEL=(,NL)
//SYSPRINT DD     UNIT=(CTC,,DEFER),LABEL=(,NL)
//SYSLMOD  DD     UNIT=PDISK,SPACE=(TRK,(10,10,1))
//SYSLIN   DD     UNIT=PDISK,SPACE=(80,(10,10))
//SORTLIB  DD     DSN=SYS1.SORTLIB,DISP=SHR
//SYSUT1   DD     DSN=&SYSUT1,UNIT=PDISK,SPACE=(1000,(60,20))
//SORTIN   DD     DSN=&RESPC,DISP=(OLD,DELETE)
//SORTOUT  DD     DSN=&RESTRI,DISP=(MOD,PASS),UNIT=PDISK,
//                SPACE=(1350,(60,50),,CONTIG),DCB=BLKSIZE=1350
//SYSIN    DD     DSN=SYS1.SYSINLIB(SYSIN36),DISP=SHR
//SORTWK01 DD     UNIT=2314,SPACE=(TRK,&T,,CONTIG)
//SORTWK02 DD     UNIT=2314,SPACE=(TRK,&T,,CONTIG)
//SORTWK03 DD     UNIT=2314,SPACE=(TRK,&T,,CONTIG)
//SORTWK04 DD     UNIT=2314,SPACE=(TRK,&T,,CONTIG)
//SORTWK05 DD     UNIT=2314,SPACE=(TRK,&T,,CONTIG)
//SORTWK06 DD     UNIT=2314,SPACE=(TRK,&T,,CONTIG)
//*
//* RE-FORMATTAGE ET VENTILATION DU TEXTE FOURNI PAR LE PRE-COMPILATEUR
//*
//ORDO     EXEC   PGM=IEORDO,PARM='LIST_SOURCE,LIST_ATTEST
//SYSPRINT DD     UNIT=(CTC,,DEFER),LABEL=(,NL)
//SYSPUNCH DD     UNIT=(CTC,,DEFER),LABEL=(,NL)
//SYSIN    DD     DSN=&RESTRI,DISP=(OLD,DELETE)
//SYSUT1   DD     DSN=&OBJET,DISP=(MOD,PASS),UNIT=PDISK,
//                SPACE=(80,(250,250)),DCB=BLKSIZE=80
//*
//* COMPILATION PL/1 DU TEXTE OBJET FORMATE PAR L'ORDONNATEUR
//*
//PL1L     EXEC   PGM=IEMAA,REGION=100K,DPRTY=(1,2)
//SYSPRINT DD     UNIT=(CTC,,DEFER),LABEL=(,NL)
//SYSPUNCH DD     UNIT=(CTC,,DEFER),LABEL=(,NL)
//SYSLIN   DD     DSN=&LOADSET,DISP=(MOD,PASS),UNIT=PDISK,
//                SPACE=(80,(250,100)),DCB=BLKSIZE=80
//SYSIN    DD     DSN=&OBJET,UNIT=PDISK,DISP=(OLD,DELETE)
//SYSUT3   DD     DSN=&SYSUT3,UNIT=(PDISK,SEP=SYSPRINT),
//                SPACE=(80,(250,250)),DCB=BLKSIZE=80
//SYSUT1   DD     DSN=&SYSUT1,UNIT=(PDISK,SEP=(SYSUT3,SYSPRINT,SYSLIN)),
//                SPACE=(1024,(60,50),,CONTIG),DCB=BLKSIZE=1024
//LKED     EXEC   PGM=IEWL,PARM='MAP,LIST',COND=(9,LT,PL1L),REGION=96K,
//                DPRTY=(0,13)
//SYSLIB   DD     DSN=SYS1.PL1LIB,DISP=SHR
//         DD     DSN=SYS1.MPELIB,DISP=SHR
//SYSPRINT DD     UNIT=(CTC,,DEFER),LABEL=(,NL)
//SYSUT1   DD     DSN=&SYSUT1,UNIT=(PDISP,SEP=(SYSLMOD,SYSLIB)),
//                SPACE=(1024,(200,20)),DCB=BLKSIZE=1024
//SYSLIN   DD     DSN=&LOADSET,DISP=(OLD,DELETE),DCB=BLKSIZE=80
//         DD     DDNAME=SYSIN
//SYSLMOD  DD     DSN=&GOSET(GO),DISP=(MOD,PASS),UNIT=PDISK,
//                SPACE=(1024,(50,20,1))
//GO       EXEC   PGM=*.LKED.SYSLMOD,COND=((9,LT,LKED),(9,LT,PL1L))
//SYSPRINT DD     UNIT=(CTC,,DEFER),LABEL=(,NL)
```

ANNEXE-6

CODIFICATION DES UNITES SYNTAXIQUES

NATURE		CLASSE		INDICATEURS SEMANTIQUES	
code	signification	code	signification	code	signification
I	CARACTERE	IC	chiffre	I	peut figurer dans un identificateur
		II	lettre	2	ne peut pas figurer dans un identificateur
		I2	délimiteurs utilisés en L.A.M.P.E.		
		I3	délimiteurs non utilisés en L.A.M.P.E.		
		I4	non délimiteurs figurant ailleurs que dans une Cst. chaine		
		I5	non délimiteurs figurant seulement dans des Cst. chaines		
2	IDENTIFICATEURS		(voir pages suivantes)		
3	LITTERAL	3I	Cst. chaine	I	littéral de même type de même longueur mais distinct a été rencontré.
		32	Cst. entière		
				2	littéral de même type de même longueur mais distinct n'a pas été rencontré.

TYPE		INDICATEURS SEMANTIQUES		
code		code	signification	abréviation
20	identificateur d'objets	I	référence un objet L.A.M.P.E.	
		2	ne référence pas un objet L.A.M.P.E.	
21	indicateurs d'instruction	I	DECLARE	DCL
		2	PUT	PUT
22	indicateurs de fonction	I	NBPAGE	NBPAGE
		2	COMPATIBILITE	COMPA
23	indicateurs de commande	I	DISPOSER	DISPO
		2	SOULIGNER	SOULI
		3	DELIMITER	DELI
		4	INSERER	INSE
		5	CENTRER	CENTRER
		6	DECALER	DECA
		7	SEPARER	SEPA
		8	ALIGNER	ALI
		9	ENUMERER	ENU
		IO	IMPLANTER	IMPLAN
		II	ASSOCIER	ASSO
24	modes	I	FICHIER	FICHIER
		2	MODELE	MODE
		3	REPRESENTATION	REPRE
		4	PROPOSITION	PROPO
		5	EXTERIEUR	EXTE
25	attributs	I	LINEAIRE	LINE
		2	TABULE	TABU
		3	STRUCTURE	STRUCTU
		4	ARBITRAIRE	ARBI
		5	FIXE	FIXE
		6	LIBRE	LIBRE
		7	RESTRICTIF	RESTRI
		8	PERMANENT	PERM

26	restrictions	1	LAMPE	LAMPE
		2	HAUT	HAUT
		3	BAS	BAS
		4	DROITE	DROITE
		5	GAUCHE	GAUCHE
		6	ENTETE	ENTE
		7	PRIORITE	PRIO
27	arguments de l'édition	1	CARACTERE	CARAC
		2	LIGNE	LIGNE
		3	PAS	PAS
		4	A	A
		5	ITERATION	ITE
		6	SERIALISATION	SERI
28	éléments de démarcation	1	FIN	FIN
		2	ET	ET
		3	PUIS	PUIS
		4	AVEC	AVEC
29	symbole de base extérieur	1	IF	IF
		2	THEN	THEN
		3	EDIT	EDIT
		4	LIST	LIST
		5	DATA	DATA
		6	DIMENSION	DIM
		7	FIXED	FIXED
		8	FLOAT	FLOAT
		9	CHARACTER	CHAR
		⋮	⋮	⋮

A N N E X E 7

A- INFLUENCE DES DIRECTIVES SUR LES DISPOSITIFS DU SYSTEME M.P.E.

Pour assurer la concrétisation de tout objet éditable, les Dispositifs de Formation et d'Impression des Pages-Physiques doivent prendre en compte les requêtes mentionnées dans le corps de la déclaration de tout objet éditable.

En fonction de leur signification (Cf Manuel de Référence §1.4.4.) les directives influencent diversement les Dispositifs mis en oeuvre.

a/ En ce qui concerne la Directive générale

a-1

La Directive Générale influence le Dispositif de Formation, en raison des regroupements qu'elle réalise sur des opérandes d'une même requête. Ce qui entraine des difficultés dans la constitution des pages physiques du fait de l'incompatibilité possible entre les dimensions du support et celles de l'entité constituée par les objets regroupés.

Afin que dans ce cas, les décisions appropriées puissent être prises et que les découpes envisageables puissent être assumées (Cf première partie §2.3.6.),il est essentiel de connaitre la surface allouée, à l'objet traité, sur la page courante.

A cet effet, le Dispositif de Formation impose, pour le traitement des opérandes, un ordre préférentiel qui peut se résumer de la façon suivante :

Pour toute requête où figure un objet Représentation fixant l'aspect d'un objet de base composé, le dispositif doit prendre en compte :

* Le groupe auquel appartient l'objet Représentation, une fois déterminée l'implantation des objets éventuellement désignés dans des groupes disjoints.
* L'objet Représentation lui-même, une fois déterminée l'implantation des autres objets appartenant éventuellement au même groupe.

a-2

La directive générale influence le Dispositif d'Impression en raison de l'ordre chronologique qu'elle impose aux objets auxquels elle s'adresse.

A cet effet, le Dispositif doit prendre en compte :

* Les requêtes dans l'ordre où elles apparaissent dans le corps de la déclaration.
* Les groupes dans l'ordre où ils apparaissent dans la requête
* Les objets dans l'ordre où leurs identificateurs apparaissen dans chaque groupe.

b/ En ce qui concerne les directives spécialisées

b-1

Les directives spécialisées influencent le Dispositif de Formation, en raison des variations qu'elles font subir aux Rectangles-entités.

A cet effet, le Dispositif doit prendre en compte ces directives, avant que de déterminer l'emplacement des informations représentées par les opérandes concernés.

b-2

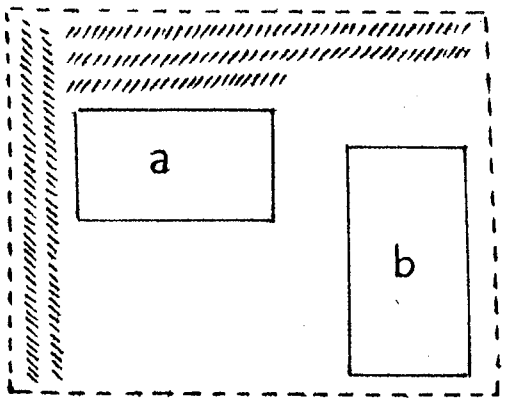
Les directives spécialisées influencent le Dispositif d'Impression en raison de leur incidence sur l'aspect de l'Edition.

A cet effet, le Dispositif doit prendre en compte ces directives, dans l'ordre où elles apparaissent dans les requêtes.

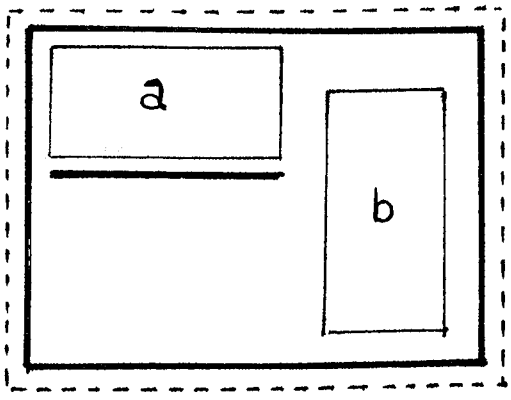
EXEMPLE

Soit la requête : "DISPOSER DELIMITER INSERER SOULIGNER a FIN ET DECALER b;

Dispositif de Formation



Dispositif d'Impression



Nous constatons :

- que les requêtes doivent se présenter sous leur structure initiale ou modifiée, selon celui des deux Dispositifs du Système M.P.E. auquel elles sont soumises.

- que la modification de la structure des requêtes (ou Restructuration), motivée par l'occurrence d'un objet Représentation fixant l'aspect d'un objet de base composé, est possible dès la pré-compilation.

  Il s'agit d'un travail préliminaire fondamental dans l'accomplissement d'une Mise-en-Pages.

- que les résultats de la Restructuration doivent transparaitre dans l'expansion du corps de la déclaration. Ce qui implique la composition des Vecteurs-Informations générés.

B- REPRESENTATION PARAMETREE CONDENSEE DU CORPS DE LA DECLARATION D'UN OBJET EDITABLE.

a/ Représentation paramétrée

Le corps de la déclaration d'un objet éditable est traduit, à l'issue de l'Analyse Syntaxique, en une liste de cellules. Les Cellules-Opérandes, par leur présence et les renseignements qu'elles incluent, suffisent à exprimer, sous forme paramétrée, la structure des requêtes.

En effet,

Nous pouvons différencier, de par leurs actions sémantiques, les directives spécialisées de la directive générale.

Nous pouvons dissocier, implicitement, parmi les commandes, celles introduites par une directive spécialisée de celles introduites par une directive générale et dénommée groupe.

Nous pouvons considérer, par conséquent, le groupe comme étant, syntaxiquement (Cf Manuel de Référence §1.4.2.) et sémantiquement un opérande au même titre qu'un objet.

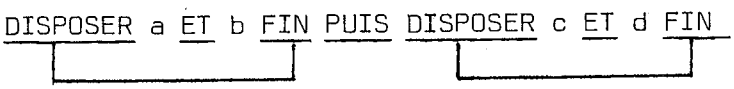
La paramétrisation du corps de la déclaration d'un objet éditable découle désormais de trois constatations.

- Dans le contexte du corps de la déclaration d'un objet éditable, pour caractériser un opérande, il est essentiel de définir à l'aide des indications exprimées sous la forme de "Rang","Niveau","Priorité", respectivement

* la requête à laquelle l'opérande appartient,
* le groupe dont fait partie l'opérande,
* le numéro d'ordre de l'opérande dans le groupe.

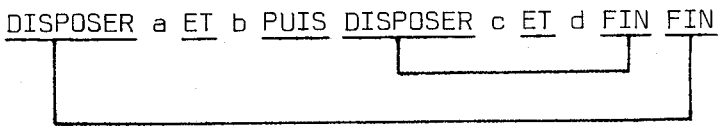
- Dans une requête, les groupes peuvent être adjacents et représenter des opérandes de même niveau.

EX.1

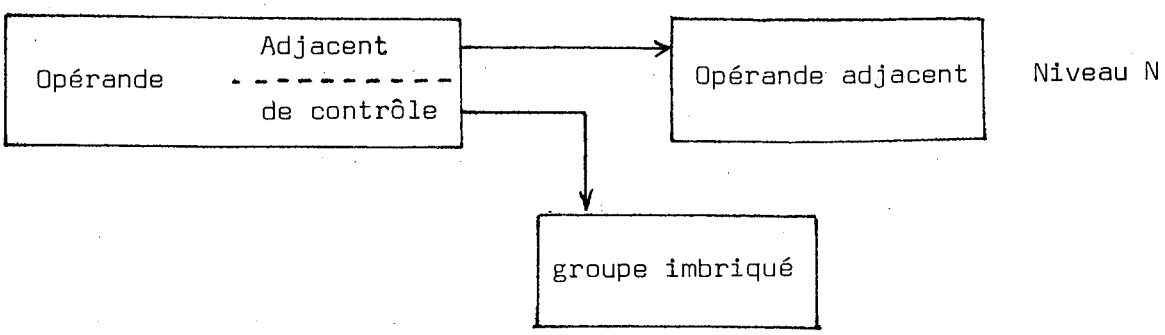


- Dans une requête, les groupes peuvent être imbriqués et représenter des opérandes de niveau différent.

EX.2



En conséquence, la structure significative de toute requête est basée sur les relations établies entre ses opérandes et résumées par le schéma suivant :



Ainsi :

- lorsque l'opérande est un groupe,

le terme "groupe adjacent" désigne un groupe séparé, par un opérateur de mise en relation, des groupes de même niveau (EX-1).

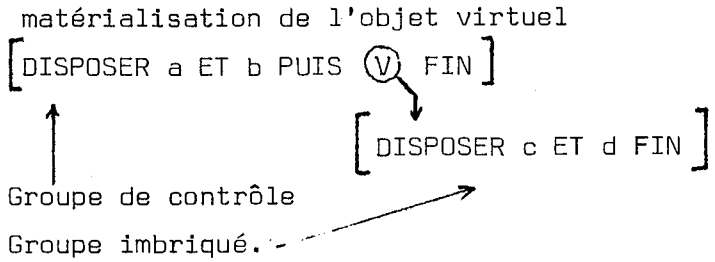
Le terme "groupe de contrôle" désigne un groupe, dont l'un des opérandes est lui-même un groupe. Ce qui vaut à ce dernier d'être dénommé "groupe imbriqué" (EX-2).

- lorsque l'opérande est un objet;

le terme "objet adjacent" désigne un objet séparé, par un opérateur de mise en relation, des objets de même niveau (EX-1 ou EX-2).

Le terme "objet de contrôle" désigne un "objet virtuel" appartenant à un "groupe de contrôle", et traduisant l'occurrence, à cet emplacement, d'un "groupe imbriqué". Ce qui permettra de faire subir à l'objet virtuel", et par suite au groupe qu'il représente, les mêmes traitements que ceux imposés aux autres objets du groupe.

EXEMPLE

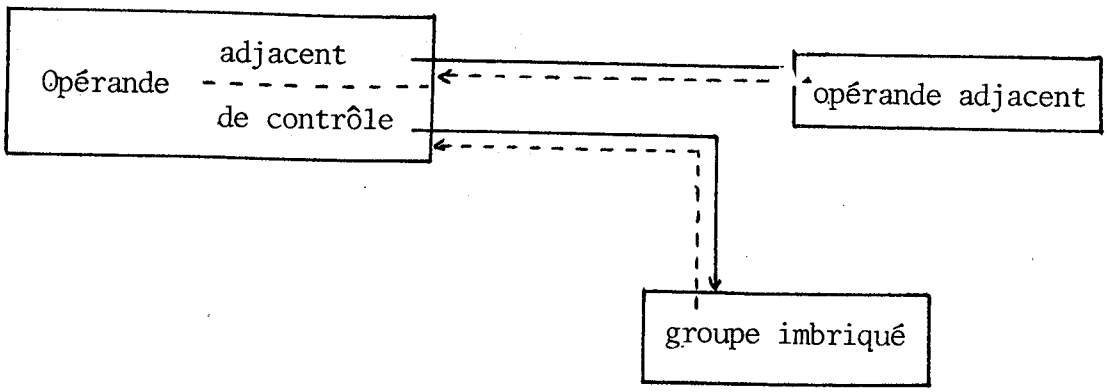


b/ Représentation paramétrée condensée

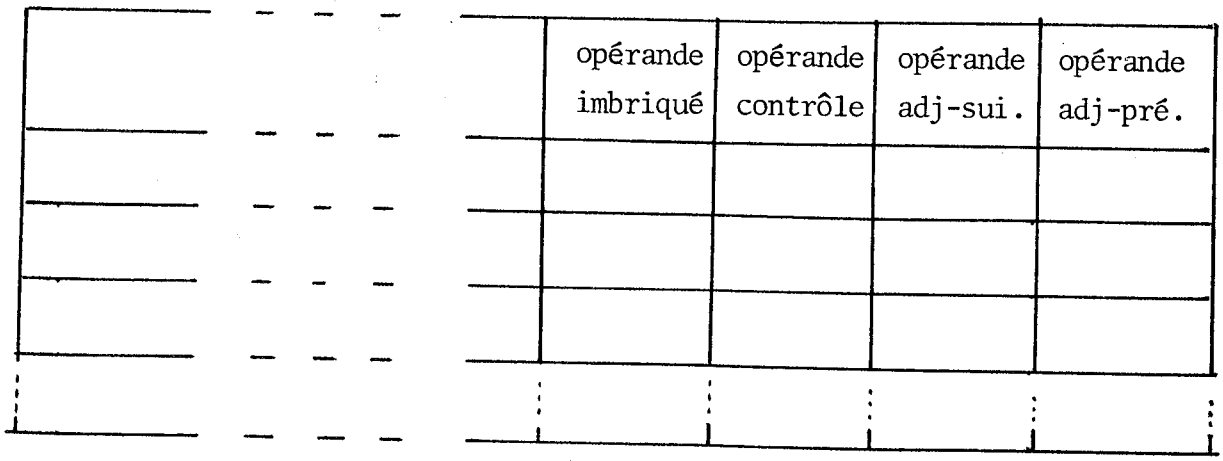
De façon à faciliter l'opération de Restructuration et à préparer la génération des V.I.D., les informations extraites de chaque Cellule-Opérande sont rangées dans un table dite "de structuration des requêtes".

Chaque élément de la table représente une Cellule-Opérande. Aux valeurs extraites des Cellules-Opérandes (paramètres et pointeurs vers les Cellules-Directives et Cellules-Données) s'ajoutent des pointeurs vers d'autres éléments de la table; ce qui permet d'exprimer les liaisons entre opérandes. En raison des fréquents retours arrière qu'impose la Restructuration, le chaînage établi est bi-univoque.

Schéma du chainage bi-univoque.



Matérialisation du chainage bi-univoque dans la table.



Algorithme assurant le chainage bi-univoque.

```
/* */
/* ALGORITHME DE CHAINAGE */
/* */
        CHAINAGE : PROC ;
        DCL (J,K) FIXED BIN(15) ;
        TR.NOUV_OPE(LIMIT) = TR.OPE(LIMIT) ;
        BOUCLE : DO J = HEADER(I,2) TO LIMIT - 1 ;
           TR.NOUV_OPE(J) = TR.OPE(J) ;
           IF TR.NIV(J) = TR.NIV(J+1)
              THEN BEGIN ;
                 TR.ADJ_SUI(J) = J+1 ; TR.ADJ_PRE(J+1) = J ; END ;
              ELSE IF TR.NIV(J) <  TR.NIV(J+1)
                 THEN BEGIN ;
                    TR.IMBR(J) = J+1 ; TR.CONT(J+1) = J ; END ;
                 ELSE BEGIN ;
                    INTERM = TR.NIV(J+1) ;
                    DO K = J TO 1 BY -1 ;
                       IF TR.NIV(K) = INTERM
                          THEN BEGIN ;
                             TR.ADJ_SUI(K) = J+1 ; TR.ADJ_PRE(J+1) =
                             GOTO HALT ;
        HALT : END BOUCLE ;
        END CHAINAGE ;
```

c/ Les techniques de Restructuration

Dans tout objet éditable composé, chaque requête décrit un Rectangle-Entité occupant au sein du support virtuel, requis par le mode de raisonnement propre à L.A.M.P.E., une surface de dimensions finies.

L'un quelconque des rectangles constitutifs du Rectangle-Entité peut résulter de la représentation d'un objet de base composé. Dans ce cas, conformément aux principes adoptés, le Dispositif assurant le passage du support virtuel (Page-Virtuelle ou Pages-Formelles) à des supports physiques de dimensions imposées (Pages-Physiques), implique, si besoin est, la découpe de l'objet de base composé.

Pour assurer cette découpe, il est nécessaire de connaitre la surface libre sur la page courante, une fois implantés tous les rectangles autres que celui résultant de la Représentation de l'objet de base composé. Il faut donc, sans altérer la logique de la requête, en modifier la structure.

Ainsi, la concrétisation de l'objet Représentation considéré doit être envisagée, en dernier, par le Dispositif de Formation. Tel est le but de la Restructuration.

La condition nécessaire pour que la logique de la requête ne soit pas altérée est que le Rectangle-Entité ait les mêmes dimensions, qu'il soit déduit de la requête avant ou après Restructuration (voir application page

A cet effet, il suffit de permuter, dans la requête, les groupes "contrôlant" l'objet fixant la représentation de l'objet de base composé, et de faire de même pour les éléments constitutifs de chaque groupe concerné.

Les diverses configurations possibles sont exprimées par le schéma de la page suivante.

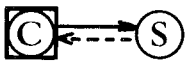
L'élément concerné :

* n'a pas de successeur

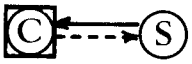
Aucun traitement.

* a un seul successeur

- n'a pas de predecesseur

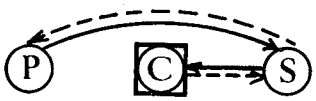


Traitement :

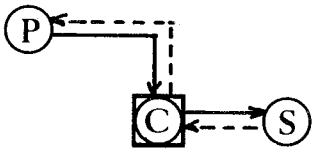


- à un prédécesseur adjacent

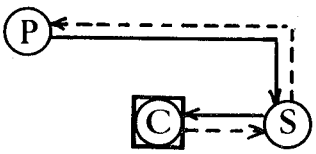
Traitement :



- a un prédécesseur de contrôle



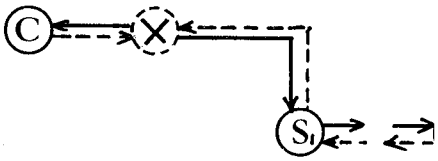
Traitement :



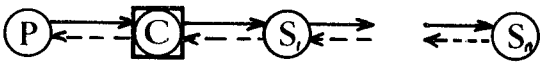
* a plus d'un successeur

- n'a pas de prédécesseur

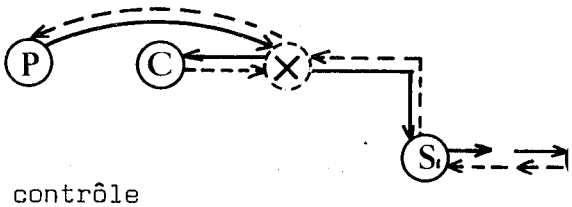
Traitement :



- a un prédécesseur adjacent

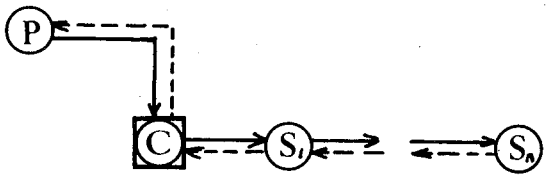


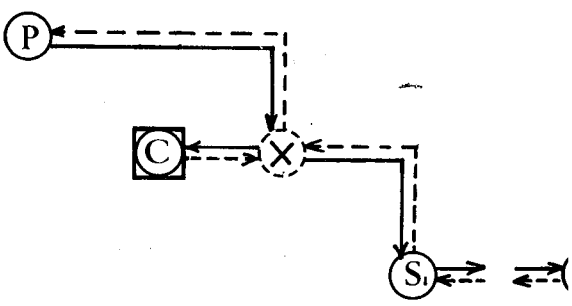
Traitement :



- a un prédécesseur de contrôle

Traitement :





Légende :

Successeur adjacent

Prédécesseur adjacent

Successeur imbriqué

Prédécesseur de contrôle

élément courant

élément "Prédécesseur"

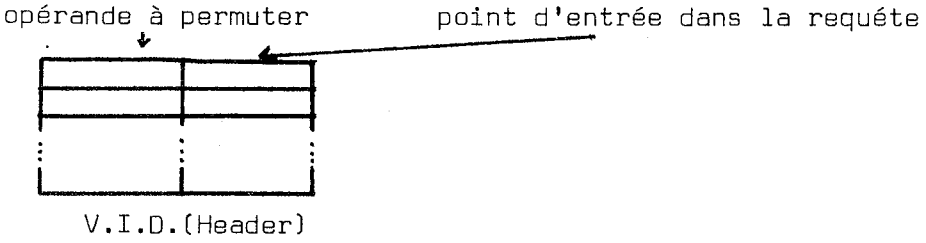
élément "Successeur"

élément "Prédécesseur" généré

Le traitement assurant les permutations, précédemment codées, est assuré par l'algorithme figurant page A-7-10. Ce traitement est particulièrement rapide en fonction de la paramétrisation de la requête et par suite, de son domaine d'application à la table de structuration des requêtes.

L'algorithme de permutation, dont le traitement est préparé par l'algorithme de chaînage (page A-7-6), est complété par l'algorithme de déchaînage (page A-7-11. Celui-ci ayant pour rôle, de donner à la requête restructurée une forme adaptée à la méthode de consultation propre aux Processus du système M.P.E.

A ce titre l'algorithme de déchaînage régit la constitution des V.I.D.



V.I.D.(Header)

niveau N.S.	opérateur N.S.	ordre

V.I.D.
(Evaluation)

niveau A.S.	opérateur A.S.	opérande imbriqué	opérande contrôle	opérande adj-sui	opérande adj-pre

V.I.D.
(Impression)

N.B.L'opération de Restructuration est applicable à toutes les requêtes figurant dans la déclaration d'un objet éditable composé.

```
/* */
/* ALGORITHME DE MODIFICATION DU CHAINAGE */
/* */
        PERMUT : PROC ( COURANT,TEST ) ;
        DCL (COURANT,TEST,I,J,K,L) FIXED BIN(15) ;
        NCOLOC = NCO : I = COURANT :
        START : IF TR.ADJ_SUI(I) ^= 0
           THEN BEGIN :
              J = TR.ADJ_SUI(I) ;
              IF TR.ADJ_SUI(J) = 0
                 THEN BEGIN ; /* IL EXISTE UN SEUL SUCCESSEUR
                                 ADJACENT */
                    IF TR.CONT(I) ^= 0
                       THEN BEGIN ; /* IL EXISTE UN PREDECESSEUR DE
                                       CONTROLE */
                          K = TR.CONT(I) ; TR.IMBR(K) = J ;
                          TR.CONT(J) = K ; TR.ADJ_PRE(J) = 0 ;
                          END ;
                       ELSE IF TR.ADJ_PRE(I) ^= 0
                          THEN BEGIN ; /* IL EXISTE UN PREDECESSEUR
                                          ADJACENT */
                             K = TR.ADJ_PRE(I) ; TR.ADJ_SUI(K) = J ;
                             TR.ADJ_PRE(J) = K ;
                             END ;
                          TR.ADJ_SUI(J) = I ; TR.ADJ_PRE(I) = J ;
                          TR.ADJ_SUI(I) = 0 ;
                          TR.NOUV_OPE(I) = TR.OPE(J) ;
                          TR.NOUV_OPE(J) = TR.OPE(I) ;
                 END ;
              ELSE BEGIN ; /* IL EXISTE PLUS D'UN SUCCESSEUR */
               NCOLOC,L = NCOLOC + 1 ;
                 IF TR.CONT(I) ^= 0
                    THEN BEGIN ; /* IL EXISTE UN PREDECESSEUR DE
                                    CONTROLE */
                       K , TR.CONT(L) = TR.CONT(I) ; TR.IMBR(K) = L ;
                       TR.CONT(I) = 0 ;
                       TR.NOUV_OPE(I) = TR.OPE(J) ; TR.NOUV_OPE(J) = ' ' ;
                       END ;
                    ELSE IF TR.ADJ_PRE(I) ^= 0
                       THEN BEGIN ; /* IL EXISTE UN PREDECESSEUR
                                       ADJACENT */
                          K , TR.ADJ_PRE(L) = TR.ADJ_PRE(I) ;
                          TR.ADJ_SUI(K) = L ; TR.NOUV_OPE(L) = TR.OPE(I) ;
                          TR.NOUV_OPE(I) = TR.OPE(J) ; TR.NOUV_OPE(J)=' ';
                          END ;
                       TR.ADJ_PRE(I) , TR.CONT(J) = L ;
                       TR.ADJ_SUI(I) , TR.ADJ_PRE(J) = 0 ;
                       TR.ADJ_SUI(L) = I ; TR.IMBR(L) = J ;
                    END ;
              END ; /* ON RECHERCHE L'ELEMENT CONTROLE DU BLOC */
               ITER : IF (TR.ADJ_PRE(I) ^= 0 & TR.ADJ_PRE(I) < NCO + 1)
                  THEN DO :
                     I = ADJ_PRE(I) ; GOTO ITER ;
                  END ;
               REFAIRE : I = I - 1 ;
               IF I ^= TEST - 1
                  THEN IF TR.IMBR(I) = 0
                     THEN GOTO REFAIRE ;

                     ELSE GOTO START ;
        CEFINI : END PERMUT ;
```

Algorithme de "Déchainage"

Principe

Alors que l'algorithme de "Chaînage" détermine les liens bi-univoques, en fonction de la valeur des niveaux et de l'ordre de consultation, l'algorithme de "Déchaînage" effectue le traitement inverse; en fonction des liens bi-univoques, tels qu'ils se présentent à l'issue de la permutation, il détermine l'ordre de consultation et les niveaux de la requête que le Dispositif de Formation devra prendre en considération.

Le principe est, de ce fait, directement déduit de celui adopté par l'algorithme de chaînage.

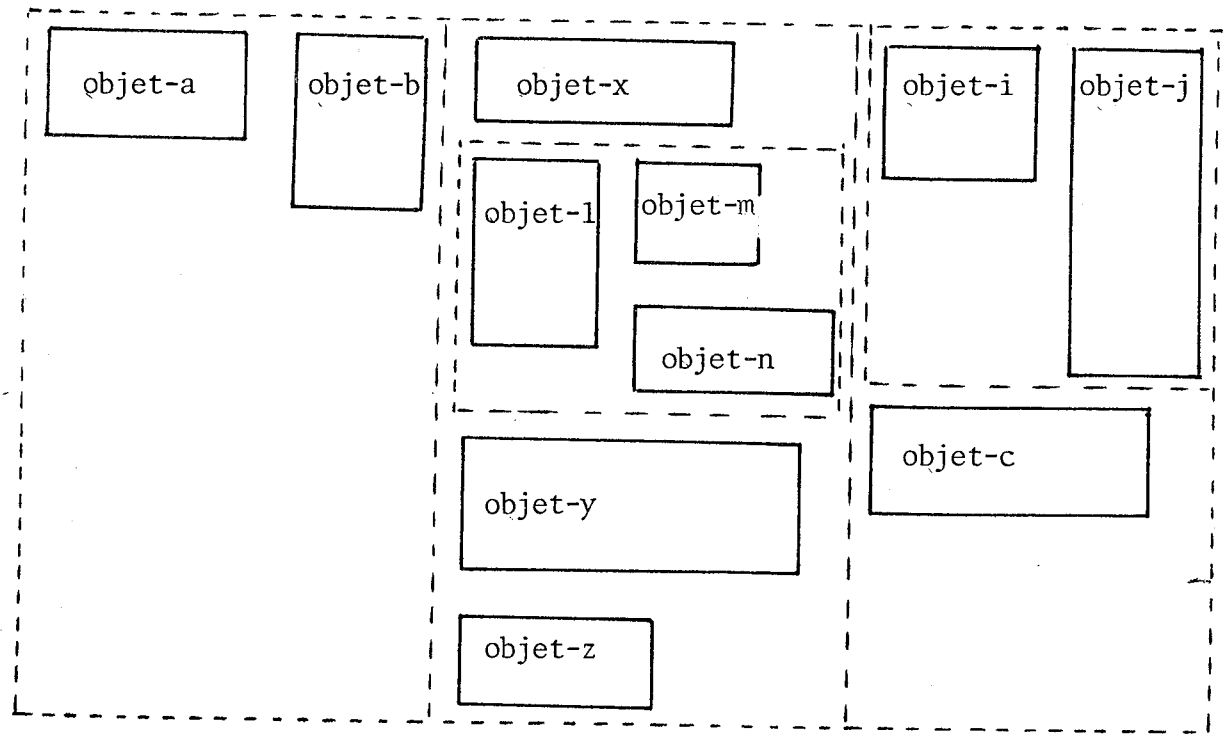
```
/* */
/* ALGORITHME DE NORMALISATION EN VUE DE L'EXPANSION */
/* */
        DECHAINAGE : PROC (ELEMENT1) ;
        DCL (ELEMENT1,I,J,SAUVE(10,2)) FIXED BIN(15) ;
        ISAUVE = 1 ; MNIVEAU = 0 ; I = ELEMENT1 ;
        UNDEPLUS : TR.NOUV_NIV(I) = MNIVEAU ;
        IF TR.IMBR(I) ^= 0
           THEN DO ;
              MNIVEAU = MNIVEAU + 1 ;
              IF TR.ADJ_SUI(I) ^= 0
                 THEN DO ; /* SI L'ELEMENT COURANT ADMET DES SUCCESSEURS
                              IMBRIQUES ET ADJACENTS, SAUVEGARDE DE L'INDICE
                              ET DU NIVEAU COURANTS */
                    SAUVE(ISAUVE,1) = I ; SAUVE(ISAUVE,2) = MNIVEAU-1 ;
                    ISAUVE = ISAUVE + 1 ;
                 END ;
              TR.ORDRE(I) = TR.IMBR(I) ; I = TR.IMBR(I) ;
              GOTO UNDEPLUS ;
              END ;
           ELSE IF TR.ADJ_SUI(I) = 0 /* SI L'ELEMENT COURANT N'ADMET AUCUN
                                        SUCCESSEUR, RECUPERATION DES DERNIERS
                                        INDICE ET NIVEAU SAUVEGARDES */
              THEN IF ISAUVE = 1
                 THEN GOTO TERM ;
                 ELSE DO ;
                    ISAUVE = ISAUVE - 1 ; J = SAUVE(ISAUVE,1) ;
                    MNIVEAU = SAUVE(ISAUVE,2) ;
                    TR.ORDRE(I) = TR.ADJ_SUI(J) ; I = TR.ADJ_SUI(J) ;
                    GOTO UNDEPLUS ;
                 END ;
              TR.ORDRE(I) = TR.ADJ_SUI(I) ; I = TR.ADJ_SUI(I) ;
              GOTO UNDEPLUS ;
        TERM : END DECHAINAGE ;
```

Exemple :

Soit la requête suivante:

```
DISPOSER  OBJET_A  ET  OBJET_B  ET  DISPOSER
 |                                    |  OBJET_X  PUIS  DISPOSER
 |                                    |                    |  OBJET_L  ET  DISPOSER
 |                                    |                    |                    |  OBJET_M
 |                                    |                    |                  FIN  PUIS  OBJE
 |                                    |                  FIN  PUIS  OBJET_Y  PUIS  OBJET_
 |                                  FIN  ET  DISPOSER
 |                                             |  OBJET_I  ET  OBJET_J
 |                                           FIN  PUIS  OBJET_C
FIN  ;
```

Le "Rectangle-Entité" correspondant a la configuration suivante:



Supposons que l'objet-m résulte de rep-ésentation d'un objet de base composé
Les pages suivantes décrivent la composition de la "Table de structuration de
requête et son évolution au cours des diverses phases précédents l'expansion.

INITIALISATION

HEADER

2 1

TABLE REQUETES	NUMERO	NIV_AS	NIV_NS	IMBRIQUE	CONTROLE	ADJ_SUI	ADJ_PRE	RESULTAT	OPE_AS	OPE_NS	
	1	0	0	0	0	0	0	0			
	2	1	0	0	0	0	0	0			--> OBJET_A
	3	1	0	0	0	0	0	0	E		--> OBJET_B
	4	1	0	0	0	0	0	0	E		
	5	2	0	0	0	0	0	0			--> OBJET_X
	6	2	0	0	0	0	0	0	P		
	7	3	0	0	0	0	0	0			--> OBJET_L
	8	3	0	0	0	0	0	0	E		
	9	4	0	0	0	0	0	0			--> OBJET_M
	10	3	0	0	0	0	0	0	P		--> OBJET_N
	11	2	0	0	0	0	0	0	P		--> OBJET_Y
	12	2	0	0	0	0	0	0	P		--> OBJET_Z
	13	1	0	0	0	0	0	0	E		
	14	2	0	0	0	0	0	0			--> OBJET_I
	15	2	0	0	0	0	0	0	E		--> OBJET_J
	16	1	0	0	0	0	0	0	P		--> OBJET_C
	17	0	0	0	0	0	0	0			
	18	0	0	0	0	0	0	0			
	19	0	0	0	0	0	0	0			
	20	0	0	0	0	0	0	0			

CHAINAGE

HEADER

2 1

TABLE REQUETES	NUMERO	NIV_AS	NIV_NS	IMBRIQUE	CONTROLE	ADJ_SUI	ADJ_PRE	RESULTAT	OPE_AS	OPE_NS	
	1	0	0	2	0	0	0	0			
	2	1	0	0	1	3	0	0			--> OBJET_A
	3	1	0	0	0	4	2	0	E	E	--> OBJET_B
	4	1	0	5	0	13	3	0	E	E	
	5	2	0	0	4	6	0	0			--> OBJET_X
	6	2	0	7	0	11	5	0	P	P	
	7	3	0	0	6	8	0	0			--> OBJET_L
	8	3	0	9	0	10	7	0	E	E	
	9	4	0	0	8	0	0	0			--> OBJET_M
	10	3	0	0	0	0	8	0	P	P	--> OBJET_N
	11	2	0	0	0	12	6	0	P	P	--> OBJET_Y
	12	2	0	0	0	0	11	0	P	P	--> OBJET_Z
	13	1	0	14	0	16	4	0	E	E	
	14	2	0	0	13	15	0	0			--> OBJET_I
	15	2	0	0	0	0	14	0	E	E	--> OBJET_J
	16	1	0	0	0	0	13	0	P	P	--> OBJET_C
	17	0	0	0	0	0	0	0			
	18	0	0	0	0	0	0	0			
	19	0	0	0	0	0	0	0			
	20	0	0	0	0	0	0	0			

PERMUTATION

HEADER

2 1

TABLE REQUETES	NUMERO	NIV_AS	NIV_NS	IMBRIQUE	CONTROLE	ADJ_SUI	ADJ_PRE	RESULTAT	OPE_AS	OPE_NS	
	1	0	0	17	0	0	0	0			
	2	1	0	0	0	0	17	0		E	--> OBJET_A
	3	1	0	0	17	4	0	0	E		--> OBJET_B
	4	1	0	5	0	13	3	0	E	E	
	5	2	0	0	4	6	0	0			--> OBJET_X
	6	2	0	7	0	11	5	0	P	P	
	7	3	0	0	6	8	0	0			--> OBJET_L
	8	3	0	9	0	10	7	0	E	E	
	9	4	0	0	8	0	0	0			--> OBJET_M
	10	3	0	0	0	0	8	0	P	P	--> OBJET_N
	11	2	0	0	0	12	6	0	P	P	--> OBJET_Y
	12	2	0	0	0	0	11	0	P	P	--> OBJET_Z
	13	1	0	14	0	16	4	0	E	E	
	14	2	0	0	13	15	0	0			--> OBJET_I
	15	2	0	0	0	0	14	0	E	E	--> OBJET_J
	16	1	0	0	0	0	13	0	P	P	--> OBJET_C
	17	0	0	3	1	2	0	0			
	18	0	0	0	0	0	0	0			
	19	0	0	0	0	0	0	0			
	20	0	0	0	0	0	0	0			

DECHAINAGE

HEADER

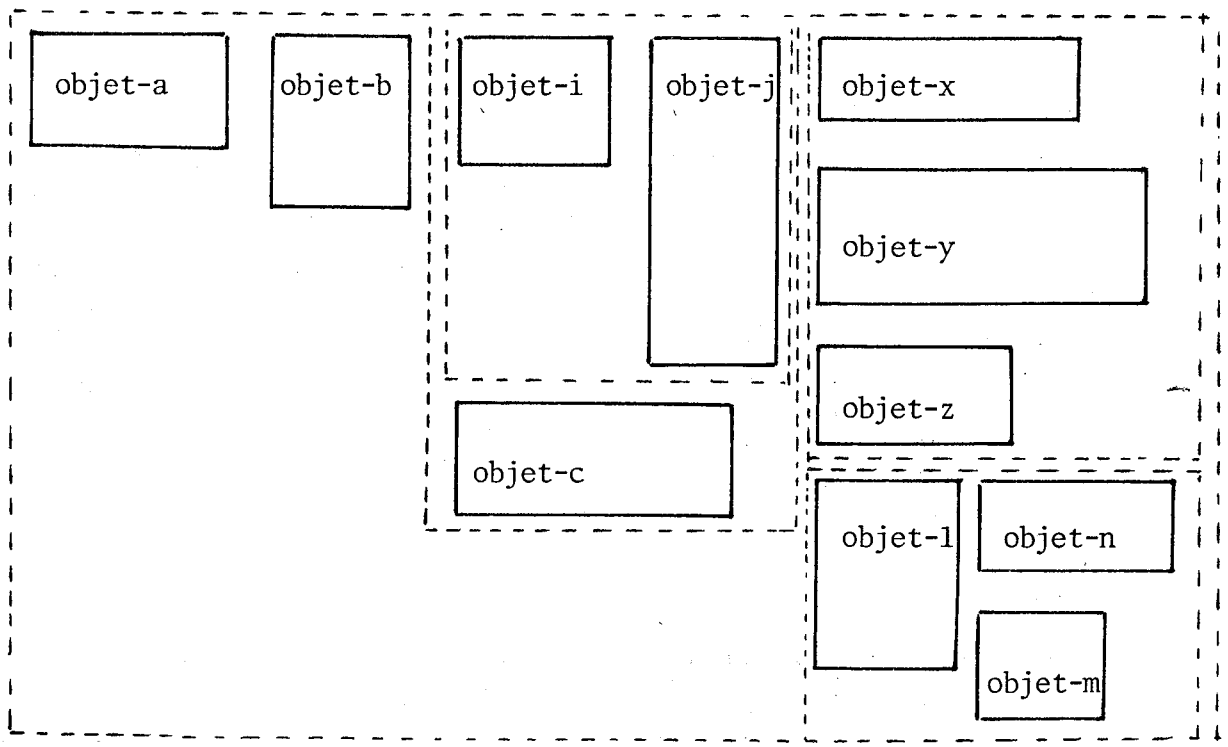
2 1

TABLE REQUETES	NUMERO	NIV_AS	NIV_NS	IMBRIQUE	CONTROLE	ADJ_SUI	ADJ_PRE	RESULTAT	OPE_AS	OPE_NS	
	1	0	0	17	0	0	0	17			
	2	1	1	0	0	0	17	0		E	--> OBJET_A
	3	1	2	0	17	4	0	4	E		--> OBJET_B
	4	1	2	5	0	13	3	5	E	E	
	5	2	3	0	4	6	0	6			--> OBJET_X
	6	2	3	7	0	11	5	7	P	P	
	7	3	4	0	6	8	0	8			--> OBJET_L
	8	3	4	9	0	10	7	9	E	E	
	9	4	5	0	8	0	0	10			--> OBJET_M
	10	3	4	0	0	0	8	11	P	P	--> OBJET_N
	11	2	3	0	0	12	6	12	P	P	--> OBJET_Y
	12	2	3	0	0	0	11	13	P	P	--> OBJET_Z
	13	1	2	14	0	16	4	14	E	E	
	14	2	3	0	13	15	0	15			--> OBJET_I
	15	2	3	0	0	0	14	16	E	E	--> OBJET_J
	16	1	2	0	0	0	13	2	P	P	--> OBJET_C
	17	0	1	3	1	2	0	3			
	18	0	0	0	0	0	0	0			
	19	0	0	0	0	0	0	0			
	20	0	0	0	0	0	0	0			

Aprés Restucturation la requéte revet la forme suivante:

```
DISPOSER
  OBJET_A  ET  OBJET_B  ET  DISPOSER
                              DISPOSER
                                OBJET_I  ET  OBJET_J
                              FIN  PUIS  OBJET_C
                            FIN  ET  DISPOSER
                                       OBJET_X  PUIS DISPOSER
                                                       OBJET_Y  PUIS  OBJET_Z
                                                     FIN  PUIS  DISPOSER
                                                                  OBJET_L  ET  OBJET_N  PUIS  DISPOSER
                                                                                               OBJET_M
                                                                  FIN                          FIN
                                       FIN
FIN ;
```

Le Rectangle-Entité, ainsi décrit, correspond à la configuration suivante:



Ses dimensions sont identiques à celles du Rectangle-Entité mentionné page A.8.12.
Seule la répartition des rectangles inscrits a changé.

```
DEBUT : BEGIN ;
        DCL mod(N) MODELE, N EXTERIEUR ; ..... ; FIN ;
        DCL (variable_x, variable_y, variable_z) FIXED BIN (15) ;
        DCL objet_d PROPOSITION FIXE ; SOULIGNER variable_y (F(5.2)) ; FIN ;
        BEGIN ;
             DCL statis (*,*,*) FIXED BIN (15) CONTROLLED ;
             GET EDIT (I) (F(5)) ; ALLOCATE statis (5,4,I) INIT (sp (I)) ; /* allocation et initialisation */
             BEGIN ;
                  DCL objet_x REPRESENTATION TABULE (statis(*,*,*))  ;
                       ASSOCIER mod (HBOUND (statis, 3) - LBOUND (statis, 3) +1) ; ..... , FIN ;
                  DCL objet_a (parm_k, parm_ℓ, parm_m, parm_n, parm_p) PROPOSITION FIXE, (parm_k, parm_ℓ (parm_p)) PROPO ;
                            parm_n PRESEN, (parm_m, parm_p) EXTERIEUR ;
                       DISPOSER parm_m puis parm_ℓ (parm_p) ET objet_c (parm_n) puis variable_x puis DECALER parm-k ;
                  FIN ;
                  DCL objet_c (parm_j) PROPOSITION FIXE, parm_j PRESENTATION ; DISPOSER DELIMITER INSERER parm_j ; FIN ;
                  BEGIN ;
                       DCL (variable_p, variable_m) FIXED BIN (15) ;
                       DCL objet_b (parm_i) PROPOSITION FIXE, parm_i EXTERIEUR ; DISPOSER variable_z ET SOULIGNER parm_i ; FIN
                       BEGIN ;
                            DCL variable FIXED BIN (15) ;
                            PUT LAMPE (objet_b (variable)) ;
                       END ;
                       PUT LAMPE (objet_a (objet_d, objet_b, variable_m, objet_x, variable_p)) ;
        END DEBUT ;
```

```
      INSTRUCTION   objet_b    objet_a    objet_c    objet_x    mod      objet_d                    EXPANSION PONCTUELLE
-------------------------------------------------------------------------------------------------------------------------------------------
(1)                 DCL des Vecteurs Informations de "lobjet_b"
                    P_OOGobjet_b : PROC ;
                         V_OOAobjet_b(2) = T_LIBR ; /* Indexation après recherche de la prémière position libre dans le DISPATCHER */
(2)                      Allocation + Initialisation du V.I.B. (Donnée) de "variable_z" ; /* Un aiguillage distingue Initialisation de
                                                                                                                    réinitialisation */
      CALL P_OOGobjet_b ;
      V_OOAobjet_b (3) = T_LIBR ;
      Allocation + Initialisation du V.I.B. (Donnée de "variable" ;
                 END ;
      CALL T_UNIP (V_OOHobjet_b, ..., V_OODobjet_b) ; /* préparation à l'appel des processus (à partir des V.I. d'un objet éditable) */
(3)   appel des processus d'Evaluation puis d'impression ;
-------------------------------------------------------------------------------------------------------------------------------------------
(1)                           DCL des Vecteurs Informations de "l'objet_a" ;
      CALL P_OOBobjet_a ;
      V_OOAobjet_a (2) = T_LIBR ;
(2)   Allocation + Initialisation du V.I.B. (Donnée) de "variable_m" ;
      CALL P_OOGobjet_b
      V_OOAobjet_b (3) = T_LIBR ;
      Allocation + Initialisation du V.I.B. (Donnée de "variable_p" ;
      CALL T_UNIQ (V_OOHobjet_b, ..., V_OODobjet_b) ;
                                      DCL des Vecteurs Informations de "l'objet_c" ;
                                                DCL des Vecteurs Informations de "l'objet_x" ;
(4)                                             Extraction et ordonnacement des composants de l'objet de base
                                                          DCL des Vecteurs Informations de l'objet : "mod" ;
                                                          DCL N FIXED BIN (15) INIT (HBOUND (STATIS, 3) - LBOUND (STATIS, 3) +1) ;
 *                                              Allocation et Intialisation des V.I.B. (Donnée) du Modèle
      CALL T_UNIP (V_OOHobjet_x,..., V_OODobjet_x) ;
      CALL T_UNIM (V_OO du modèle) ;
 *    CALL T_UNIP (V_OOH objet_c , ..., V_OODobjet_C) ;
                              P_OOGobjet_a : PROC ;
                                V_OOAobjet_a (5) = T_LIBR ;
                              Allocation et Initialisation du V.I.B. (Donnée de "variable_x" ;
      CALL P_OOGobjet_d ;
                                                                  DCL des Vecteurs Informations de "l'objet_d" ;
                                                                  P_OOGobjet_d ;
                                                                      V_OOAobjet_d (2) = T_LIBR ;
                                                                      Allocation et Initialisation du V.I.B. (Donnée) de
                                                                                                          "variable y"
                                                                  END ;
                              END ;
```

INSTRUCTION objet_b objet_a objet_c objet_x mod objet_d

```
CALL T_UNIP (V_OOHobjet_d, ..., V_OODobjet_d) ;
CALL T_UNIP (V_OOHobjet_a, ..., V_OODobjet_a) ;
appel des processus d'Evaluation puis d'impression ;
```

NB :

(1)

(2) à remplacer par
```
ALLOCATE V_OOGDONNEE SET (V_OOGP IN V_OOGAREA) ;
instructions d'initialisation de "V_OOGP → V_OOGDONNEE" ; (voir § 2.3.1.3.3.e)
V_OOGI = V_OOA ident (nbr) = T_LIBR ; /* récupération */
V_OOGDISPATCH (V_OOGI) = V_OOGP ; /* corrélation */
```

(3) à remplacer par la séquence mentionnée dans le § correspondant
```
FREE V_OOGDISPATCH (V_OOA ident (nbr)) → V_OOGDONNEE ;
V_OOGDISPATCH (V_OOAident (nbr)) = 0 ; /* libération */
```
si argument : séquence systématique
si identificateur : séquence conditionnée par une utilisation ultérieure ou non

en conséquence diviser la répartition INSTRUCTION en
- INS avant appel des processus ⇒ préparation et mise en oeuvre
- INS après appel des processus ⇒ libération

TEXTE PRE-TRAITE

```
DEBUT : BEGIN ;

        DCL des Vecteurs Informations de l'objet : "mod" ;
        DCL N   FIXED BIN (15) INIT (HBOUND (STATI, 3) - LBOUND (STATI, 3) +1) ;

        DCL (variable_x, variable_y, variable_z) FIXED BIN (15) ;

        DCL des Vecteurs Informations de "l'objet_d" ;
        P_OOGobjet_d
            V_OOAobjet_d (2) = T_LIBT ;
            Allocation et Initialisation du V.I.B. (Donnée) de "variable_y" ;
        END ;

        BEGIN

             DCL statis (*,*,*) FIXED BIN (15) CONTROLLED ;
             GET EDIT (I) (F(5)) ; ALLOCATE statis (5,4,I) INIT (SP (I)) ; /* allocation et initialisation */

             BEGIN ;

                  DCL des Vecteurs Informations de "l'objet_x" ;
                  Extraction et ordonnancement des composants de l'objet de base ;
                  Allocation et initialisation des V.I.B. (Donnée) du Modèle ;

                  DCL des Vecteurs Informations de "l'objet_a" ;
                  P_OOGobjet_a : PROC ;
                      V_OOAobjet_a (5) = T_LIBR ;
                      Allocation et Initialisation du V.I.B. (Donnée) de "variables_x" ;
                  END ;

                  DCL des Vecteurs Informations de "l'objet_c" ;

                  BEGIN ;

                       DCL (variable_p, variable_m) FIXED BIN (15) ;

                       DCL des Vecteurs Informations de "l'objet_b" ;
                       P_OOGobjet_b : PROC ;
                           V_OOHobjet_b (2) = T_LIBR ;
                           Allocation et Initialisation du V.I.B. (Donnée) de "variable_z" ;
                       END ;
                                  }
```

```
BEGIN ;

     DCL variable FIXED BIN (15) ;

     CALL P_OOGobjet_b ;
     V_OOAobjet_b (3) = T_LIBR ;
     Allocation + Initialisation du V.I.B. (Donnée) de "variable" ;
     CALL T_UNIP (V_OOHobjet_b, ..., V_OODobjet_b) ;
     appel des processus d'Evaluation puis d'Impression ;

END ;

CALL P_OOGobjet_a ;
V_OOAobjet_a (2) = T_LIBR ;
Allocation + Initialisation du V.I.B. (Donnée) de variable_m" ;
CALL P_OOGobjet_b ;
V_OOAobjet_b (3) = T_LIBR ;
Allocation + Initialisation du V.I.B. (Donnée) de "variable_p" ;
CALL T_UNIP (V_OOHobjet_b, ..., V_OODobjet_b) ;
CALL T_UNIP (V_OOHobjet_x, ..., V_OODobjet_x) ;
CALL T_UNIM (V_OO du Modèle)
CALL T_UNIP (V_OOHobjet_c, ..., V_OODobjet_c) ;
CALL P_OOGobjet_d ;
CALL T_UNIP (V_OOHobjet_d, ..., V_OODobjet_d) ;
CALL T_UNIP (V_OOHobjet_a, ..., V_OODobjet_a) ;
appel des processus d'Evaluation puis d'Impression ;
END DEBUT.
```