T H E S E présentée

A L'UNIVERSITE DE PARIS-SUD

CENTRE D'ORSAY

pour obtenir

LE TITRE DE DOCTEUR INGENIEUR

par

René PERRIN

Spécialité : ELECTRONIQUE

Sous-spécialité : Traitement de l'Information

Sujet de la thèse :

ETUDE ET REALISATION D'UN SYSTEME EN TEMPS PARTAGE POUR LE LANGAGE APL SUR MINIORDINATEUR MITRA 15 CII

soutenue le 28 octobre 1977 devant la Commission d'Examen composée de :

M. DEBRAINE	Président
M. DONADIEU	Examinateur
M. de COSNAC	Examinateur
M. DELCOIGNE	Membre Invité

T H E S E présentée

A L'UNIVERSITE DE PARIS-SUD

CENTRE D'ORSAY

pour obtenir

LE TITRE DE DOCTEUR INGENIEUR

par

René PERRIN

Spécialité : ELECTRONIQUE

Sous-spécialité : Traitement de l'Information

Sujet de la thèse :

ETUDE ET REALISATION D'UN SYSTEME EN TEMPS PARTAGE POUR LE LANGAGE APL SUR MINIORDINATEUR MITRA 15 CII

soutenue le 28 octobre 1977 devant la Commission d'Examen composée de :

M. DEBRAINE	Président
M. DONADIEU	Examinateur
M. de COSNAC	Examinateur
M. DELCOIGNE	Membre Invité

T H E S E présentée

A L'UNIVERSITE DE PARIS-SUD

CENTRE D'ORSAY

pour obtenir

LE TITRE DE DOCTEUR INGENIEUR

par

René PERRIN

Spécialité : ELECTRONIQUE

Sous-spécialité : Traitement de l'Information

Sujet de la thèse :

ETUDE ET REALISATION D'UN SYSTEME EN TEMPS PARTAGE POUI LE LANGAGE APL SUR MINIORDINATEUR MITRA 15 CII

soutenue le 28 octobre 1977 devant la Commission d'Examen composée de :

M. DEBRAINE	Président
M. DONADIEU	Examinateur
M. de COSNAC	Examinateur
M. DELCOIGNE	Membre Invité

AVANT - PROPOS

Je remercie Monsieur DEBRAINE, professeur à l'Institut National des Sciences et Techniques Nucléaires, de m'avoir fait l'honneur de présider le jury de cette thèse et pour l'interêt qu'il a porté à ce travail.

Je remercie Monsieur DONADIEU, Directeur de l'Institut Universitaire de Technologie d'ORSAY, d'avoir bien voulu accepter de faire partie de ce jury.

Monsieur de COSNAC, adjoint au chef de Service SERF, a fixé le cadre de cette étude et m'a fait profiter de ses conseils et de son expérience tout au long de ce travail ; qu'il trouve ici le témoignage de ma gratitude.

La réalisation est issue d'un travail d'équipe. Je tiens à remerci pour leur participation active les agents du C.E.A. :

Monsieur BASTIEN
Monsieur BRAS
Monsieur SEVIN
et Monsieur GUEGUEN, agent de la CISI/GIXI.

Que Monsieur DELCOIGNE, Ingénieur à la CISI/GIXI, trouve ici l'expression de ma profonde reconnaissance pour son aide.

Enfin, je remercie Monsieur WEILL, chef des SES, et Monsieur AMRAM, chef du SERF pour leur accueil, ainsi que le personnel du SES.

- 1 -

INTRODUCTION

I - GENERALITES.

L'APL est un langage interprétatif très conversationnel. Un utilisateur de l'APL devant sa console a le comportement suivant : Après un temps de réflexion de plusieurs dizaines de secondes, il lance l'exécution de transactions courtes en moyenne (quelques secondes) ainsi l'ordinateur au travers de ses principales ressources, Unité Centrale et périphérique, est-il sous-employé. La conception d'un système en temps-partagé s'impose naturellement afin de répartir les dites ressources entre plusieurs usagers.

Chaque usager bénéficie des temps-morts offerts par chacun des utilisateurs en cours de session.

L'objectif du système est de faire travailler plusieurs usagers simultanément en APL à la différence des systèmes d'exploitation que l'on rencontre sur d'autres machines et qui sont capables de traiter des travaux de types très différents et très indépendants. Nous verrons que la réalisation faite implique nécessairement la notion de partage de ressources logiques entraînant la notion d'intercommunication.

Le schéma proposé peut être affiné dans le sens ou le temps consacré à la frappe des instructions APL peut être inclus dans le temps de réflexion à condition de gérer totalement la commande, en plus de la gestion de tous les utilisateurs du système.

II - PRELIMINAIRE A L'ETUDE D'UN SYSTEME EN TEMPS-PARTAGE.

Un système en temps partagé permet :

- Un dialogue entre l'homme et la machine.
- L'amélioration du rendement de la machine et de l'installation.

Ce résultat est obtenu en attribuant la ressource Unité Centrale à chaque usager du système, grâce à un programme d'ordonnancement qui décide à quel utilisateur doit s'effectuer un passage donné.

Si la tâche se met en attente d'une fin d'entrée-sortie, le contrôle passe à la tâche suivante et la tâche interrompue est mise en file d'attente pour un service futur.

Néammoins les impératifs d'ordre humain font qu'il est demandé un temps de réponse le plus faible possible à l'instruction traitée. Ceci exige qu'un quantum de temps limite une situation de monopole des ressources critiques (non partageables).

Il est difficile de simuler un système en temps partagé, mais nous pouvons présumer que les taux de charges de l'Unité Centrale et de ses périphériques deviendront acceptables. L'ordinateur traitant un langage suffisamment conversationnel chaque usager aura l'impression de disposer de la machine pour lui seul, s'il bénéficie d'un temps de réponse individuellement satisfaisant.

Cette approche simpliste, quoique traditionnelle, ne fait pas apparaître les phénomènes de fond, mis en jeu par la concurrence de toutes les tâches pour l'emploi des ressources critiques du système, ici, l'Unité Centrale et le disque.

C'est donc dans le cadre plus général de la théorie des files d'attentes que se fondent la conception, l'évaluation et la validité d'un système en temps partagé. Le fait de disposer d'un langage hautement conversationnel contribue à valoriser ces points de vue ; en cela, le langage APL s'inscrit naturellement dans une configuration en temps partagé. Jusqu'ici dévolu à des installations puissantes en raison de sa complexité, ce langage est maintenant disponible sur des miniordinateurs dont le Mitra 15 de la CII fait partie.

L'adaptation de l'interpréteur du langage APL 15 à un environnement en

temps partagé, nécessite de nombreux compromis sur l'utilisation et la gestion des ressources de l'ordinateur. Nous montrerons néammoins que le système proposé utilise l'équipement avec des performances intéressantes.

Nous présenterons tout d'abord le système en temps-partagé tel qu'il est disponible actuellement sans justifier les solutions adoptées, celles-ci étant exposées dans les chapitres qui suivront la description détaillée du logiciel du Mitra 15.

PRESENTATION DU SYSTEME EN TEMPS PARTAGE

I - ENVIRONNEMENT.

Les utilisateurs du langage APL sur Mitra 15 travaillent en mode conversationnel et peuvent accéder sous contrôle du système à l'équipement externe. Des commandes adéquates permettent de lancer en parallèle un autre traitement, programme sous forme d'IMT exécutable, dont les éventuelles communications de données sont réglées par fichiers standard CII accessible à l'APL par le jeu des fichiers externes. Ces commandes assurent le lancement des programmes dans la même partition mémoire ou sous forme d'une tâche supplémentaire introduite dans l'ordonnancement.

II - GESTION DES RESSOURCES.

II.1. Unité Centrale.

L'Unité Centrale est la ressource principale du système en temps partagé. Elle est retirée au programme actif ou rendue au système lorsque celui-ci remplit une des conditions suivantes :

- Lancement d'une entrée-sortie dont le temps est égal ou supérieur à celui demandé par un échange de tâches contrôlant l'UC.
- Epuisement d'un quantum de temps UC alloué à la tâche par l'allocateur.
- Utilisation de primitives bloquantes.

La notion importante pour la politique d'ordonnancement repose sur l'évaluation des temps d'oisiveté du système. Il existe une file d'exécution (attente de la libération de l'UC) gérée par le planificateur des travaux qui donne le contrôle à une tâche devenue "éligible" par un mécanisme de type "FIFO". La tâche reçoit un quantum de temps invariable mais facilement

adaptable au profil du site d'exploitation APL. L'évaluation nous montrera comment tenir compte de ce paramètre.

Le nombre de postes partageant l'unité centrale est au maximum de 4, mais le système peut être dimensionné à la demande.

II.2. La Mémoire Centrale.

L'organisation de la mémoire centrale se déduit du fait qu'un seul utilisateur se trouve en mémoire centrale à un instant donné. Le planificateur assure donc le va et vient des tâches dans une partition fixe.

Les entrées-sorties de tâches sont assurées par le biais des tampons résidents lorsqu'il s'agit :

- De transactions des différents claviers.
- De transactions effectuées sur des périphériques non partageables et de caractéristiques lentes (temps de transaction de l'ordre de la commutation des tâches).

II-3. Les ressources physiques.

L'APL travaille sur son environnement physique par le jeu des fichiers externes. Le système en temps-partagé assure cette communication et contrôle la gestion des ressources critiques (périphérique non partageables), l'accès aux périphériques partageables (de type fichier) n'étant pas contrôlé explicitement par le système.

III - LE PROGRAMME INTERPRETEUR APL 15.

Le programme interpréteur APL 15 gère la mémoire laissée disponible par le logiciel global : l'espace de travail.

Il comporte une zone de code et une zone de données. Plusieurs tâches peuvent utiliser le code à condition de le rendre réentrant. Le programme APL étant partiellement en recouvrement, les branches seront considérées comme des données propres à la tâche contrôlant l'unité centrale. L'interpréteur travaille sur une mémoire virtuelle, chaque utilisateur aura donc sa propre mémoire virtuelle, fichier standard SGF15.

IV - LES DONNEES.

La protection est assurée par les conventions usuelles des fichiers standard gérés par le processeur UFG15.

- au niveau de la déclaration,
- au niveau de l'assignation,
- au niveau de l'ouverture,
- au niveau de la lecture/écriture.

De plus, des primitives de type sémaphore, permettent de régler les problèmes d'accès en simultanéité aux fichiers.

V - TERMINAUX CONVERSATIONNELS.

V.1. Liaison.

La liaison entre les terminaux et l'ordinateur se fait par lignes asynchrones. Le "handler" pilote, destiné aux lignes asynchrones, peut gérer jusqu'à 64 lignes simultanément. On dispose de lignes en "full-duplex" de vitesse adaptables (de 300 à 9600 bauds). L'émission-réception s'effectue en code ASCII.

La conversion de code ASCII/EBCDIC (code interne de la machine) est faite par le handler.

V.2. Les terminaux *

Les terminaux ont vis à vis du système une égale priorité ; gérés en modes alphanumérique et graphique, ils sont de deux types :

- Visualisation (9600 bauds).
- Machine à écrire avec mémoire tampon en sortie (1200 bauds).

L'acquisition se fait caractère par caractère et l'écho est réalisé par programme. Les terminaux conversent en code APL au niveau du clavier. (représentation des caractères).

VI - ETATS D'UNE TACHE.

Les principaux états d'une tâche peuvent être schématisés par la figure 2.1 Une tâche peut être active, en attente de fin de transaction sur l'équipement, suspendue derrière un sémaphore et en attente de l'unité centrale (en sommeil sur disque). A la fin de la session, le numéro de la tâche disparait de l'ordonnancement. La procédure de démarrage de la tâche, lors de la connection, remet le numéro de la tâche en attente de l'Unité Centrale.

* **SES/SERF/PUB/74.205.**
Notice d'utilisation de la console de visualisation TEKTRONIX 4013 associée au Mitra 15 CII. G. BERNEDE.

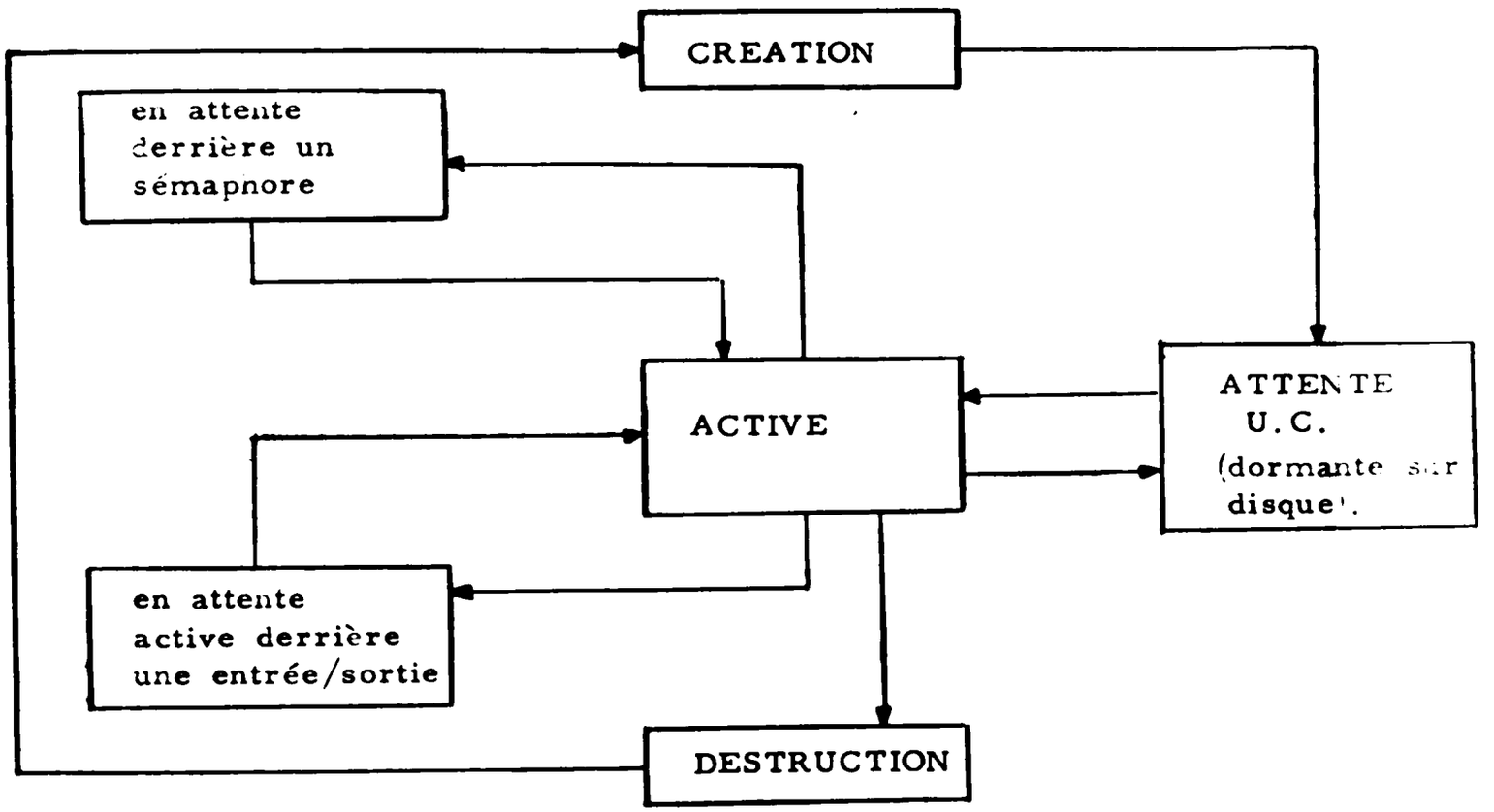


Fig. 2.1.

VII - CHEMINEMENT D'UN MESSAGE.

VII.1. Gestion des données.

A chaque terminal est associé un tampon dimensionné selon des critères adaptés au terminal. Les transactions d'entrée ou de sortie y transitent.

La tâche est en sommeil si l'usager est en entrée et si une autre tâche est prête et éligible en mémoire centrale. Une fin de transaction est détectée par le code "Retour chariot" ou "Line feed" dans certains cas, elle provoque le transfert du message dans l'espace de travail lorsque la tâche revient en mémoire.

Les caractères entrants sont contrôlés, puis transcodés en code interne APL pour permettre un certain compactage du message.

VII.2. Caractéristique de la transaction.

La gestion des terminaux s'effectue par sollicitation du niveau d'interruption supportant le processeur d'entrée-sortie après avoir précisé les modalités de la transaction. Le processeur gère totalement la transaction et le terminal à partir des directives données par l'APL. Ceci peut s'exprimer par le schéma 2.2. :

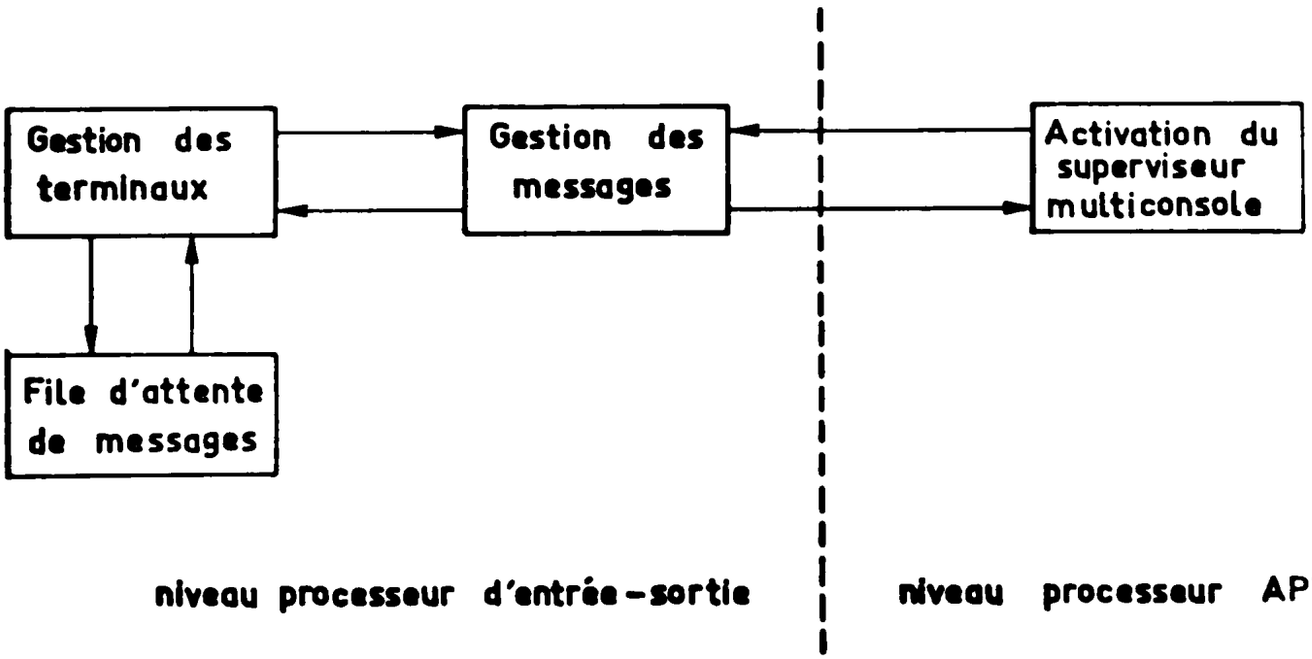


Fig. 2.2.

VIII - DIRECTIVES.

L'emploi dans un environnement en temps partagé du langage APL nécessite l'introduction de primitives destinées à l'accès de bibliothèques de façon simultanée par plusieurs utilisateurs. La synchronisation qui en résulte est assurée par le principe des sémaphores.

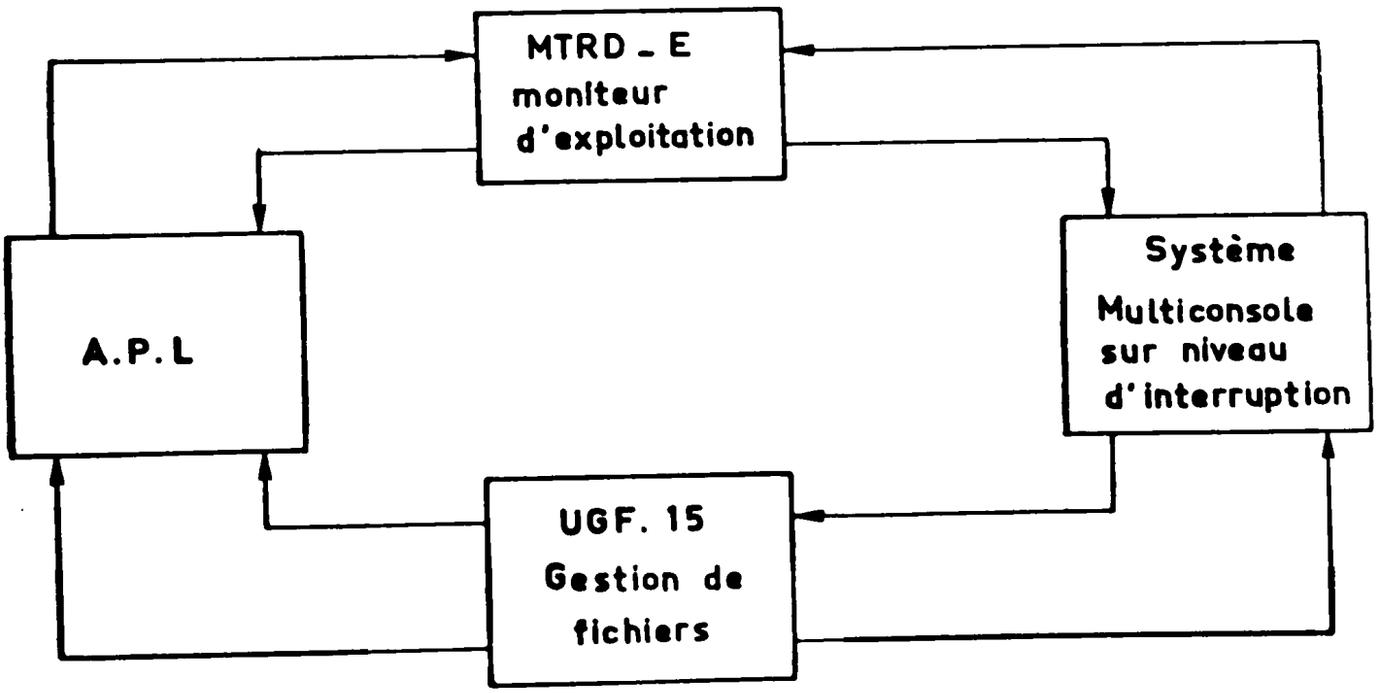
- Sémaphores accessibles et programmables par l'usager de façon à exploiter des données de façon concertée
- Sémaphores système inaccessibles à l'usager permettant à ce système en temps partagé de gérer ses tâches dans des sections critiques.

IX - ARCHITECTURE DU LOGICIEL.

Les différents logiciels mis en œuvre sont reliés entre eux.

- liaison du processeur APL et du système en temps-partagé.
- liaison du moniteur d'exploitation et du système en temps-partagé. Le dit système étant disposé sur un niveau d'interruption.

Ces interconnexions sont figurées sur le schéma 2.3.



Le moniteur d'exploitation MTRD-E version 6 a été transformé ponctuellement pour supporter le système en temps-partagé. Les modifications apportées sont reportées sur les versions MTRD-E 7 et MTRD-E 8 pour les ordinateurs de type Mitra 15 , Mitra 125 et MP 105.

L'encombrement est indiqué sur le schéma 2.4. pour une configuration à 4 utilisateurs adressant les périphériques du site (lecteur de cartes, ruban perforé, imprimante).

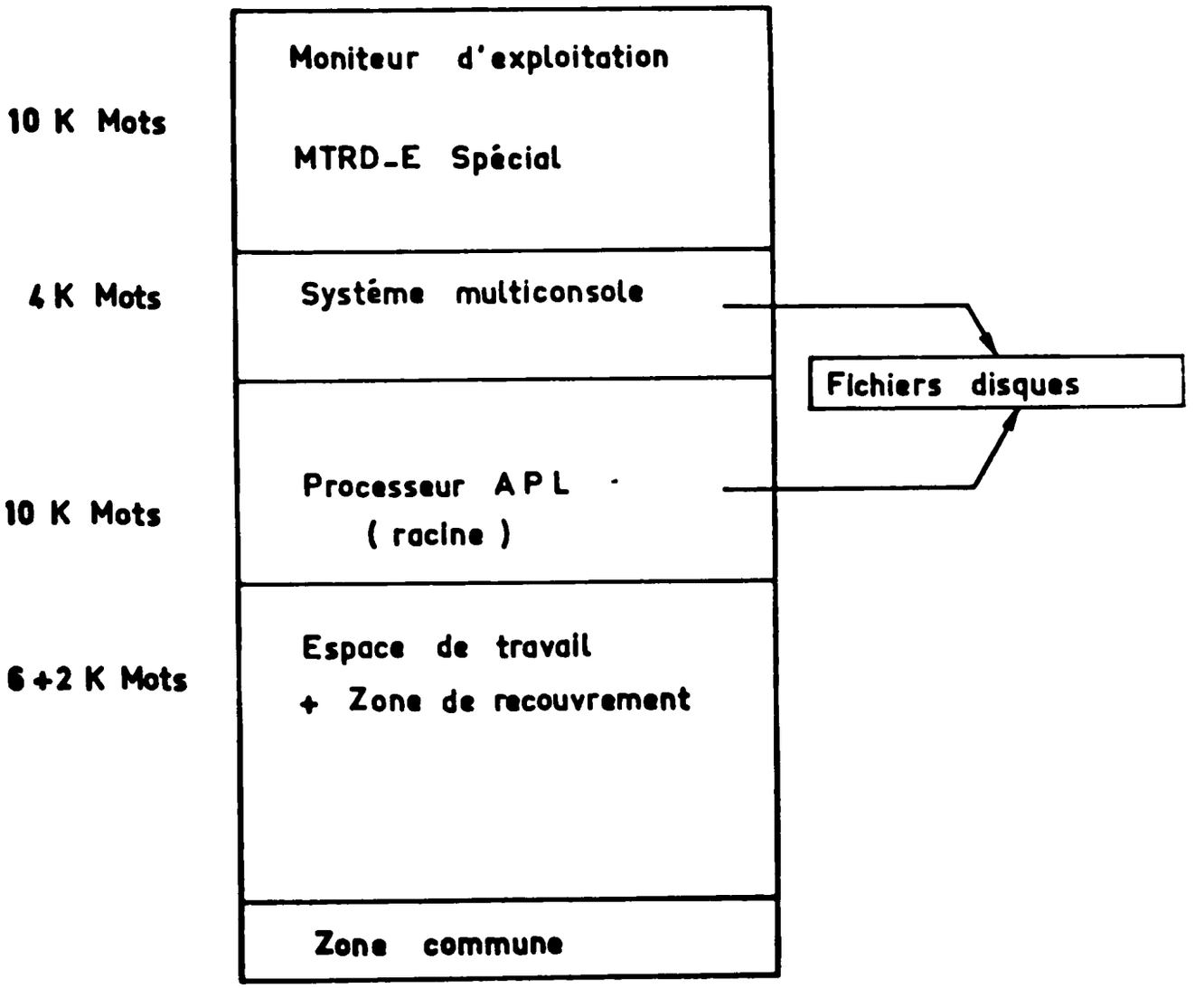


Fig. 2.4.

- 3 -

EVALUATION DU SYSTEME EN TEMPS - PARTAGE.

I - LES MODELES D'ARCHITECTURE.

L'attribution à des tâches concurrentes d'organes non partageables, fait intervenir la notion de file d'attente.

Nous verrons plus loin la modélisation relative à une description des phénomènes se passant à l'intérieur du système réel.

Il nous faut d'abord introduire quelques idées directrices sur l'architecture du système dépendant des ressources de l'ordinateur, de leur organisation et de leur gestion.

I.1. Allocation de la ressource unité centrale.

Une bonne distribution de la ressource Unité Centrale qui exécute les instructions des programmes permet de maintenir un taux d'utilisation satisfaisant par ailleur des autres ressources.

La règle de multiplexage de la ressource Unité Centrale est celle du quantum de temps, ici fixe. Les programmes étant exécutés à tour de rôle, dans l'ordre d'arrivée dans la file d'attente, et ce à chaque allocation de la durée du quantum jusqu'à l'achèvement de la session.

Un compromis doit être nécessairement trouvé entre le temps de réponse moyen accordé aux usagers et l'utilisation de la ressource Unité Centrale.
En effet :

a) Si le quantum de temps est trop court, le nombre de commutations de l'unité centrale est élevé et les opération de va et vient, de gestion des usagers conduit à une charge prohibitive.

b) Si le quantum de temps est trop élevé, le temps de réponse des transactions courtes tend à croître au détriment de l'opérateur. Les tâches necessitant n quanta de temps bloquant ces transactions.

Insistons sur le fait que le programme en cours rend la ressource Unité Centrale du système lorsqu'il n'en a plus besoin, c'est ce que nous désignerons sous le vocable de "critère de commutation de tâche". Le quantum de temps n'est là que pour reprendre exceptionnellement la ressource Unité Centrale au programme en cours et l'attribuer à une autre tâche prête.

I.2. Allocation de la ressource Mémoire Centrale.*

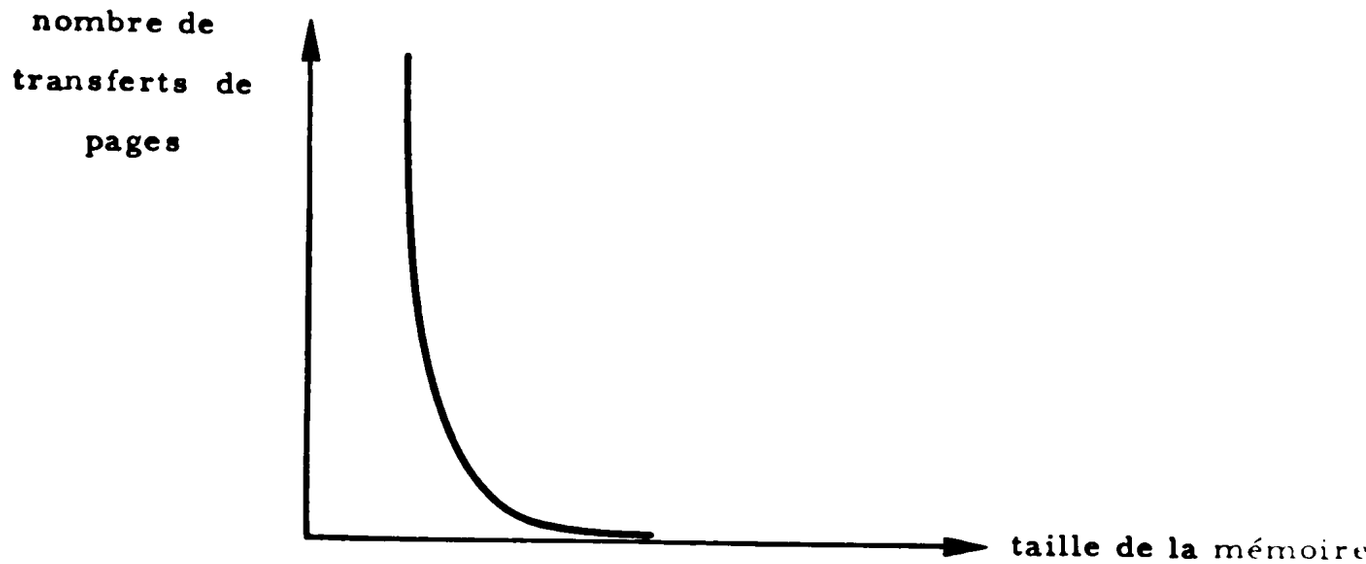
L'APL travaillant avec une mémoire virtuelle, l'allocation de la mémoire centrale est critique comme nous allons le voir.

L'étendue de l'espace de travail en mémoire constitue une donnée importante pour le comportement de l'usager vis à vis de la mémoire virtuelle. En effet, les performances globales du système dépendent :

a) Du nombre de transferts entre la mémoire secondaire et la mémoire centrale.
b) De la vitesse de transfert de l'information.

Le comportement dynamique des programmes d'applications en APL sur le site relève des études classiques sur les structures paginées et se schématise ainsi :

* SES/SERF/PUB/76-114.
Gestion de la mémoire virtuelle de l'interpréteur APL/Mitra15 (version V0)
D. BRAS.



Pour s'exécuter de façon satisfaisante un programme APL doit donc disposer instantanément d'une partie de l'espace de travail nécessaire à son exécution en mémoire Centrale.

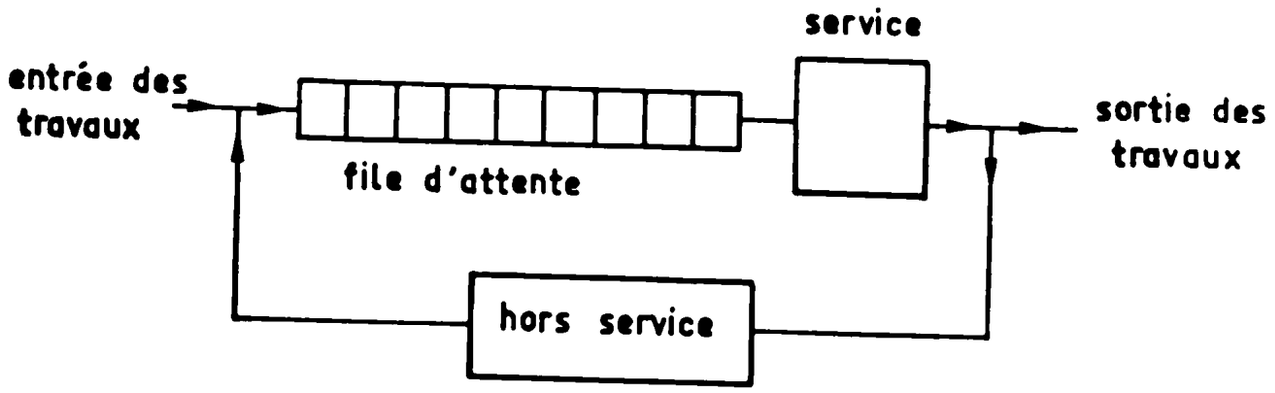
Dans ce cadre on ne pourra tolérer qu'un seul utilisateur en mémoire centrale et, de ce fait, toute référence à la mémoire virtuelle entrainera une période d'oisiveté de l'unité centrale comprise dans le traitement.

L'apparent désavantage de ne pas disposer d'une multiprogrammation au niveau de la mémoire centrale est en partie contrebalancé par la taille maximale de l'espace de travail en mémoire centrale, permettant de réduire les références à la mémoire virtuelle.

En d'autres termes, l'accès à la mémoire virtuelle sur le disque supportant les fichiers de va-et-vient, ne permet pas de prendre celui-ci comme critère de commutation de tâches.

Le comportement du système en temps-partagé peut donc être décrit par le modèle suivant qui garantit que tout travail est servi au bout d'un temps fini. Le service est assuré par un seul serveur et la stratégie

d'allocation ne concerne qu'un usager à la fois.



Modèle d'allocation de l'Unité Centrale

I.3. Gestion de la mémoire auxiliaire.

La mémoire auxiliaire constituée par le disque, est employée ainsi :

a) Utilisation en tant que composant de la hiérarchie de mémoire (support de la mémoire virtuelle).

b) Utilisation en tant que support de données partageables (bibliothèque et fichiers externes).

c) Utilisation en tant que sauvegarde pour le va-et-vient.

d) Utilisation en tant que support du mécanisme de recouvrement, l'influence de celui-ci étant observée indirectement.

I.3.1. Caractéristique physique de l'unité de mémoire secondaire.

Deux classes peuvent être considérées :

a) Les disques à tête fixe.

b) Les disques à tête mobile.

La durée moyenne de lecture d'un bloc d'information fait intervenir trois facteurs :

- Un temps de positionnement des têtes de lecture/ écriture pour les disques à tête mobile pour amener celles-ci au dessus de la piste.
- Un temps d'attente du passage du début du bloc sous les têtes (en moyenne une demi-rotation).
- Un temps de transfert du bloc lui même (dépendant de l'unité et de la taille du bloc (256 octets).

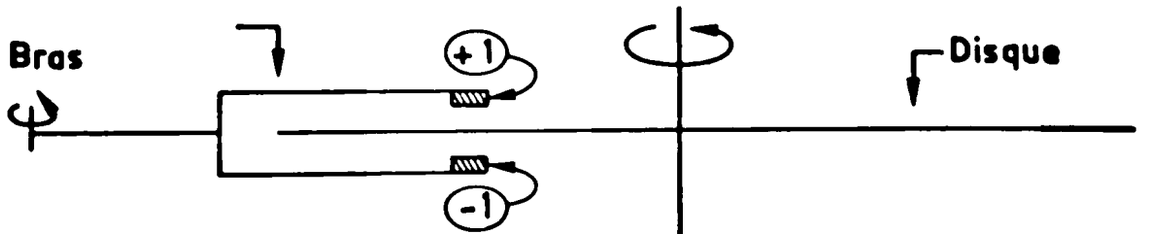
Le tableau rappelle les caractéristiques principales du disque mobile (type IBM 5440) et du disque fixe supportés par le Mitra 15 Mitra 125.

Type du disque	Temps d'accès moyen au secteur	Temps de transfert d'un secteur	temps de déplacement moyen du bras	débit
Disque à tête mobile	12, 5 ms	1, 04 ms	38 ms	312 Ko s
Disque à tête fixe	10 ms	1, 66 ms	0 ms	170 Ko s

Ceci ne fait pas apparaître la caractéristique la plus désavantageuse du disque mobile sous la forme d'un temps d'attente non négligeable (25 ms) lors des occurrences suivantes :

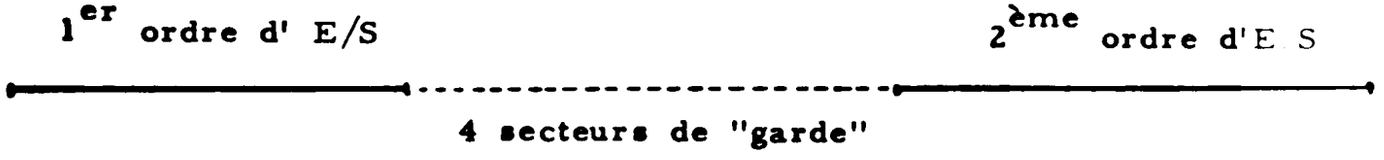
a) Une commutation de tête de lecture en écriture.

b) Une commutation de têtes pour s'adresser à l'une ou l'autre face du disque, selon le schéma suivant :



1.3.2. Les fichiers de sauvegarde : le temps de commutation de tâches

L'influence du temps de commutation de tâches sera abordé plus loin, mais il faut souligner ici son aspect nettement défavorable. Sans préciser la structure des données transférées de la mémoire sur le disque, le volume mis en jeu est en moyenne 22 Koctets. Chaque cylindre offre 12 Koctets (2 fois 24 secteurs de 256 octets), et de plus la préparation d'une entrée-sortie coûtant un temps CPU supérieur à l'intervalle inter-secteur, il est nécessaire de prévoir un saut de secteurs suffisant pour ne pas attendre un délai rotationnel inutile entre deux ordres d'entrée-sortie. Le découpage "linéaire" du fichier de sauvegarde est donc schématisé ainsi :



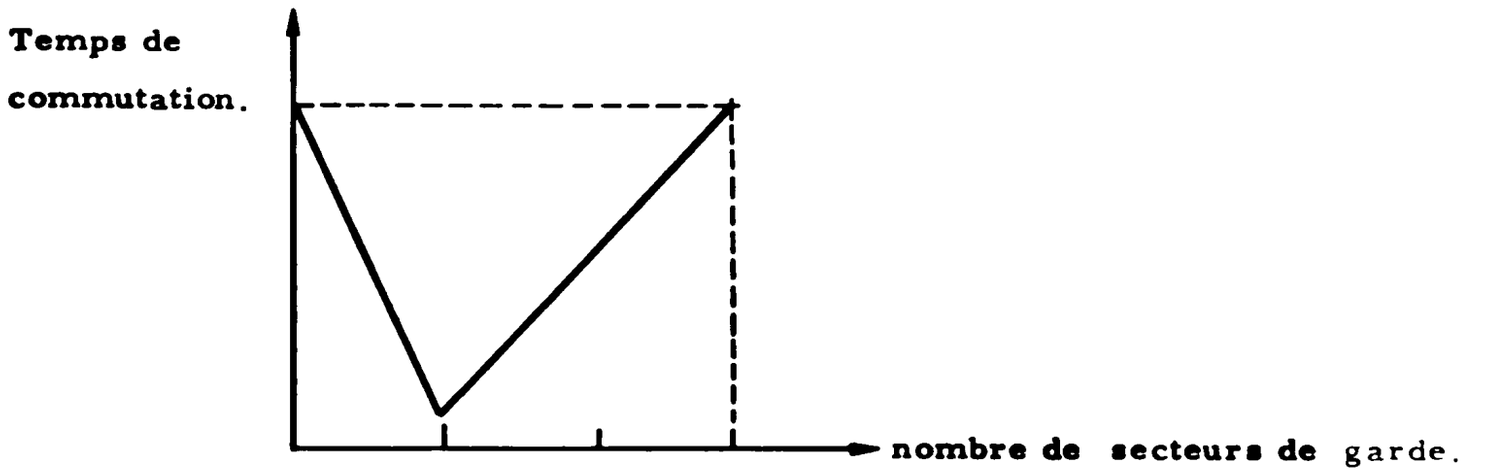
Il faut compter sur le transfert de :

$\frac{23\ K + 1K}{12\ K}$ = 2 cylindres environ variable avec la configuration disponible.

L'évaluation globale d'un échange d'utilisateur est malaisée car nous n'avons ni la maîtrise de l'implantation du fichier, ni le contrôle des commutations de têtes sur le cylindre.

L'étude sur le site a montré un temps voisin de 580 ms. Ce temps pouvant être fluctuant, l'aspect défavorable de la commutation de tâches conduit à étudier de façon précise les critères de l'algorithme de service des tâches, car une erreur d'allocation de la mémoire est très coûteuse en temps d'unité centrale.

Le nombre de secteurs de garde a été déterminé d'après un relevé de mesures dont nous donnons le résultat qualitatif schématisé par la figure suivante :



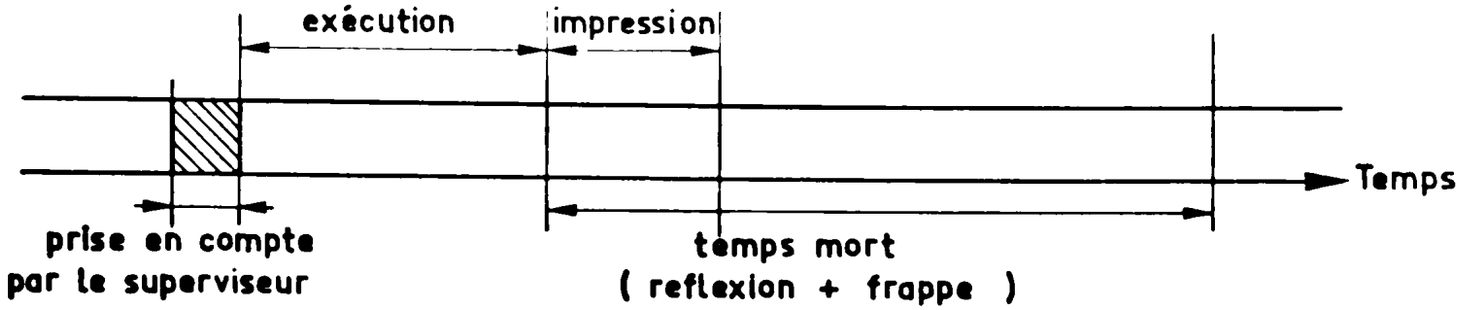
II - EVALUATION DU SYSTEME

II.1. Modèle du temps de réponse *

II.1.1. Quelques données de base.

Etant donné l'absence de mesures concrêtes sur le comportement des utilisateurs pour le système proposé, il est fait usage d'une donnée relative à l'expérience APL sur IBM 360/375. L'intervalle séparant l'arrivée du premier caractère de la réponse de l'émission du caractère fin de ligne du message suivant avoisine la valeur moyenne de 45 secondes.

* APL MITRA 15
Evaluation des performances ; Rapport de Monsieur A. GUILLON
CISI Septembre 1975.

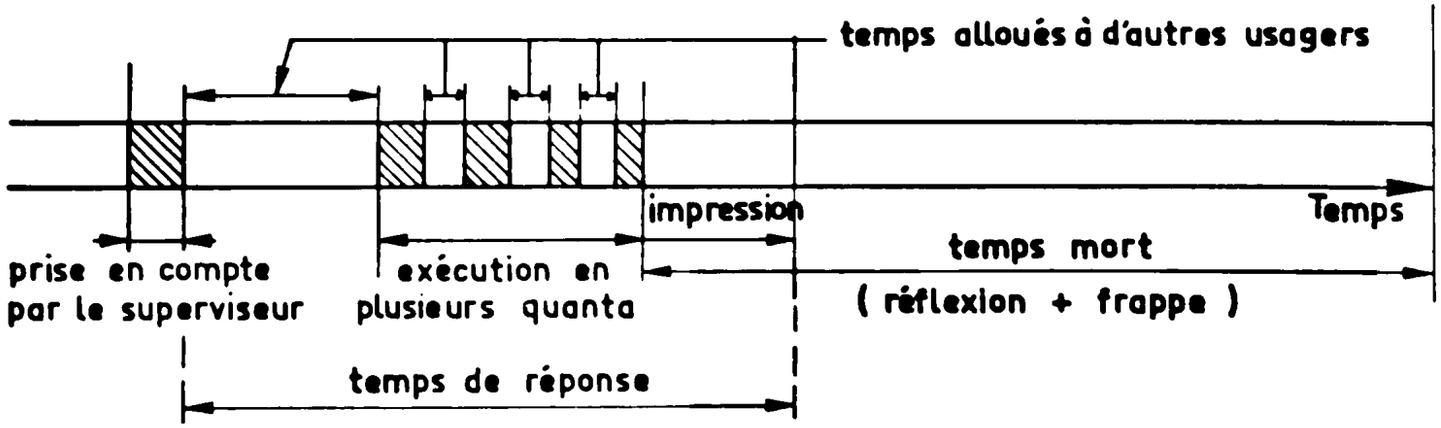


La durée du traitement de l'interaction moyenne comprend le traitement en Unité Centrale et la durée d'échange de deux utilisateurs

a) Nous pourrons évaluer la pénalisation introduite par l'attente de la ressource Unité-Centrale par l'accroissement du nombre de consoles en interaction sur le temps de traitement d'une transaction en mono console. Une base de calcul pourrait se situer vers des transaction de l'ordre de 2 à 3 secondes en moyenne.

b) Dans ces condition une durée d'échange de l'ordre de 600 ms n'aura pas une influence déterminante sur les caractéristiques du système.

c) Nous représentons la situation la plus défavorable où un quantum de temps mal adapté à l'écart-type sur les transactions moyenne impliquerait un traitement en plusieurs tranches. Il faut viser un service satisfait avant l'épuisement du quantum, étant entendu que celui-ci n'est qu'un garde-fou.



II.1.2. Définition du modèle du temps de réponse **

Le modèle tournant est bien adapté à la configuration. En effet les tâches se remettent en queue de file après avoir été servies, avec un délai plus ou moins variable. Soit N = 4 individus circulant dans le système comprenant un traitement séquentiel dont la durée suit une loi exponentielle de moyenne T le rapport Zc entre le temps hors service et le temps en service est :

$$Z_C = \frac{T}{S_0}$$

Qui impose un taux d'activité du service

$$\rho_c = 1 - \frac{Z_C^N / N!}{\sum_0^N Z^I / I!}$$

Le temps de réponse R représentant le temps de traitement ajouté à l'attente a dans la file est donné par :

$$a = S_0 \left(\frac{N}{\rho_c} - Z_C - 1 \right)$$

$$R = S_0 + a = S_0 \left(\frac{N}{\rho_c} - Z_C \right)$$

Avec les paramètres précédemment introduits nous aurrions par exemple :

N = 4

T = 45 secondes

So= 3.08 secondes dont un temps d'échange de 0.58 secondes.

conduisant à : $\begin{cases} a = 0.67 \text{ secondes} \\ R = 3.08 + 0.67 = 3.75 \text{ secondes.} \end{cases}$

** CISI - GIXI.
Séminaire évaluation des performances.
Mme VASSEUR. Mrs. ANTOINE-GUILLON.
5-6 octobre 1976. SACLAY.

A partir de cette évaluation, nous pourron fixer la valeur du quantum de temps satisfaisant les transactions de la durée moyenne évaluée et répondant au critère de temps de réponse exigible par l'usager. Un quantum de temps de 4 secondes serait adapté au profil d'exploitation indiqué dans l'exemple numérique.

II.2. Exploitation des résultats de la modélisation.

La formulation précédente est aisément exploitable en injectant dans un programme les divers paramètres afin d'étudier l'influence de ceux-ci sur le système. Cette méthode permet d'évaluer raisonnablement la configuration, en ignorant les contraintes dues à la pagination et au recouvrement considérées comme des caractéristiques propres de la transaction type en mono-console.

Les diagrammes présentent les temps de réponse qui peuvent être espérés.

a) Diagramme I. (Fig. 3.1.)

Le temps de commutation de tâches et le temps mort étant fixés ; l'influence du nombre des consoles interactives sur le temps de réponse est donné en fonction des temps moyens de transaction.

b) Diagramme II. (Fig. 3.2.)

Complète le diagramme I, en donnant le taux d'occupation du serveur, ici, l'Unité Centrale + disque.

Le taux croît avec la charge du système. Il faut rappeler que le taux d'occupation représente la probabilité pour une tâche éligible en mémoire de trouver le serveur occupé, autrement dit, de voir devant lui une file d'attente plus ou moins importante. Un faible taux d'occupation de l'Unité Centrale est caractéristique d'un langage conversationnel et correspond donc à la situation très

favorable ou les tâches sont rapidement servies et donc bénéficient d'un temps de réponse par ailleurs satisfaisant. Cette conclusion recoupe la constatation qui peut être faite sur des grosse exploitations où le taux d'activité de la ressource Unité Centrale est important au détriment du temps de réponse des travaux passant par exemple en exploitation séquentielle. Ce fait justifie de plus l'introduction d'une tâche supplémentaire dans le système. Tâche non prioritaire (elle est interruptible à tout moment par une tâche APL), elle permet d'accroitre le taux d'activité de l'Unité Centrale, sans pour cela provoquer un phénomène d'écroulement du système.

c) Diagrammes III et IV (Fig. 3.3. et 3.4.).

L'influence des facteurs temps de commutation des tâches et temps morts sur le temps de réponse aux transactions moyennes est examiné. C'est ainsi que le temps de commutation des tâches n'est pas une donnée très critique sur le temps de réponse, alors que le temps mort est une caractéristique du site d'exploitation à ne pas négliger.

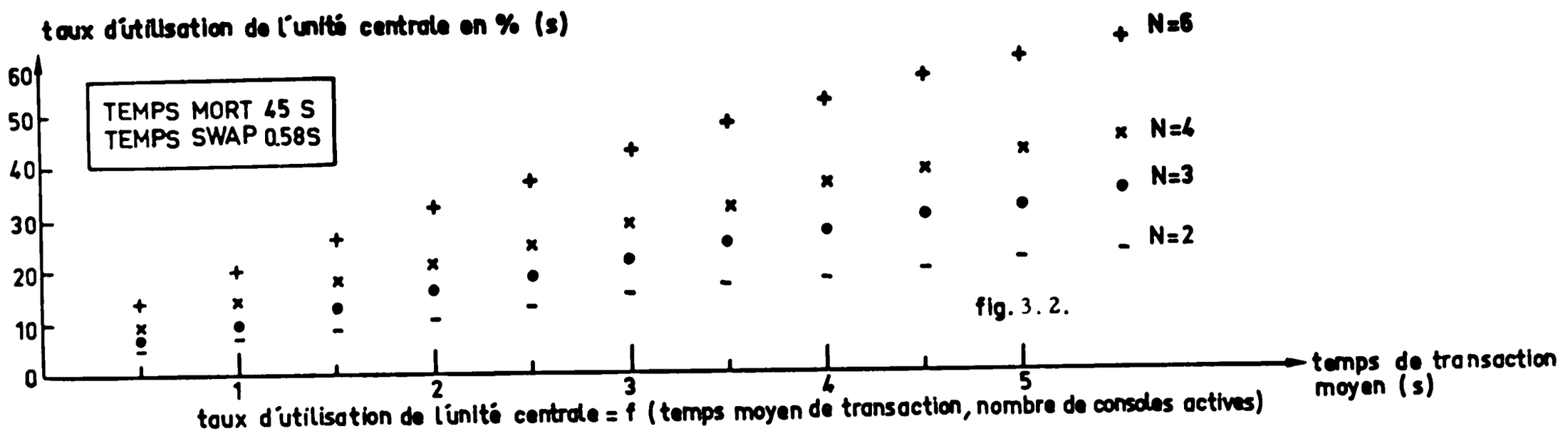


fig. 3.2.

taux d'utilisation de l'unité centrale = f (temps moyen de transaction, nombre de consoles actives)

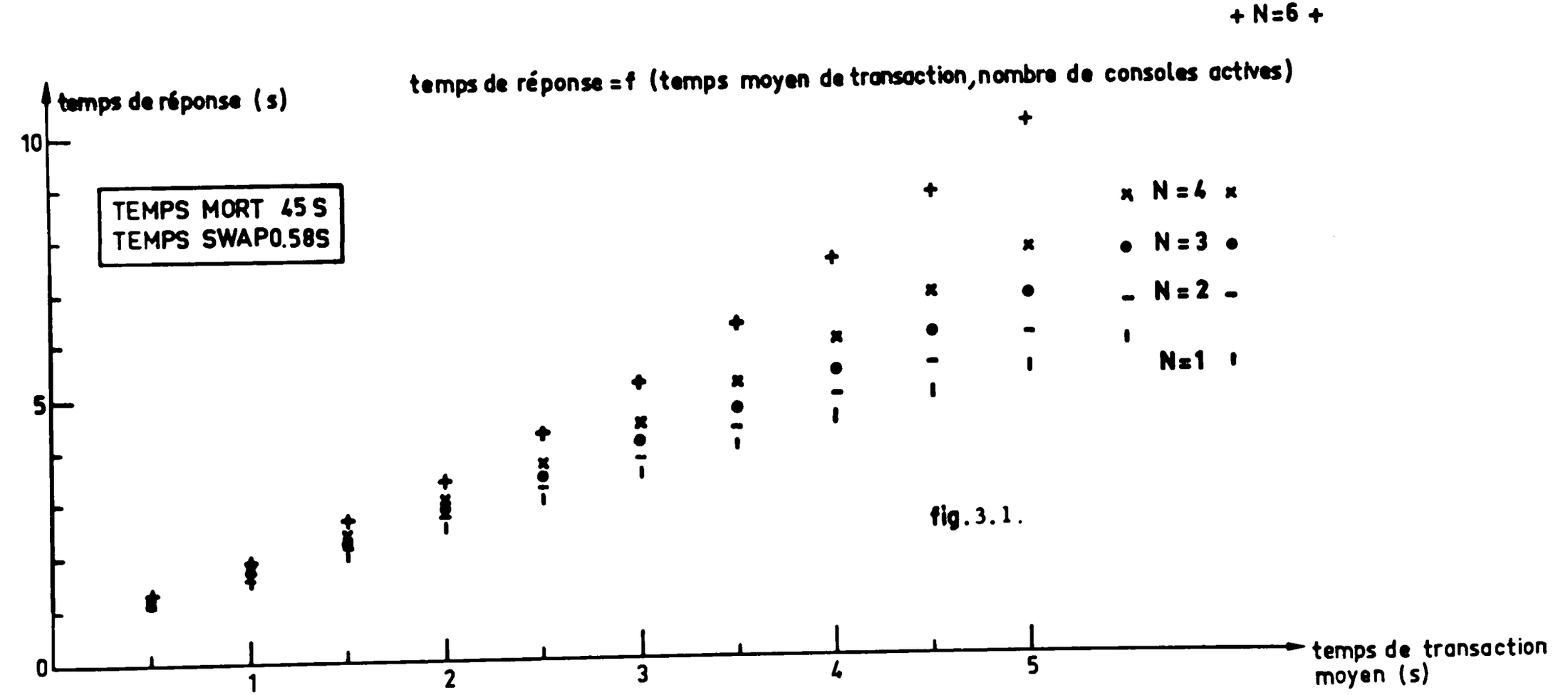


fig. 3.1.

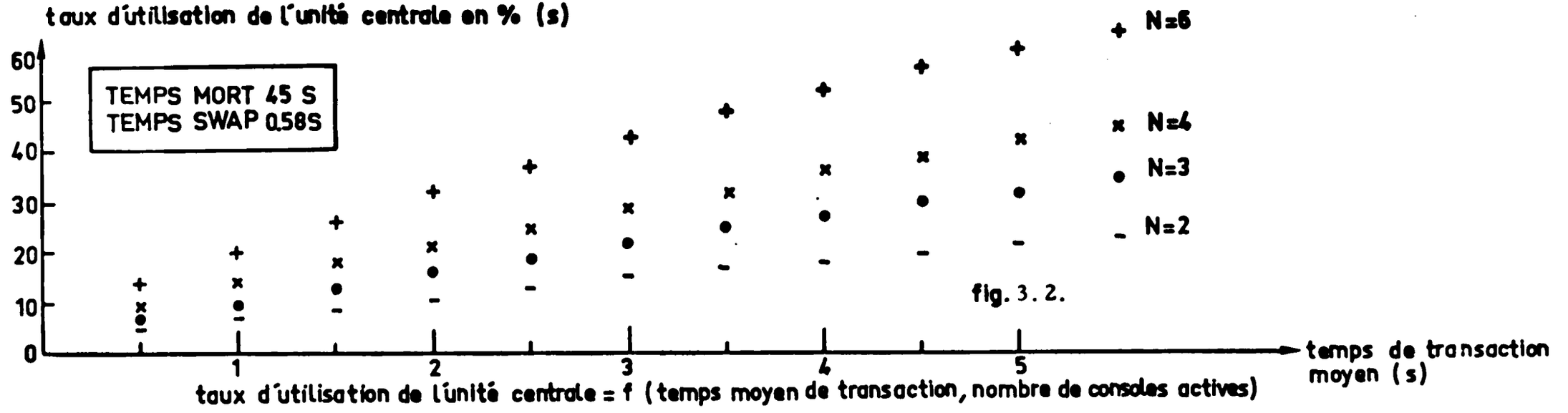


fig. 3.2.

taux d'utilisation de l'unité centrale = f (temps moyen de transaction, nombre de consoles actives)

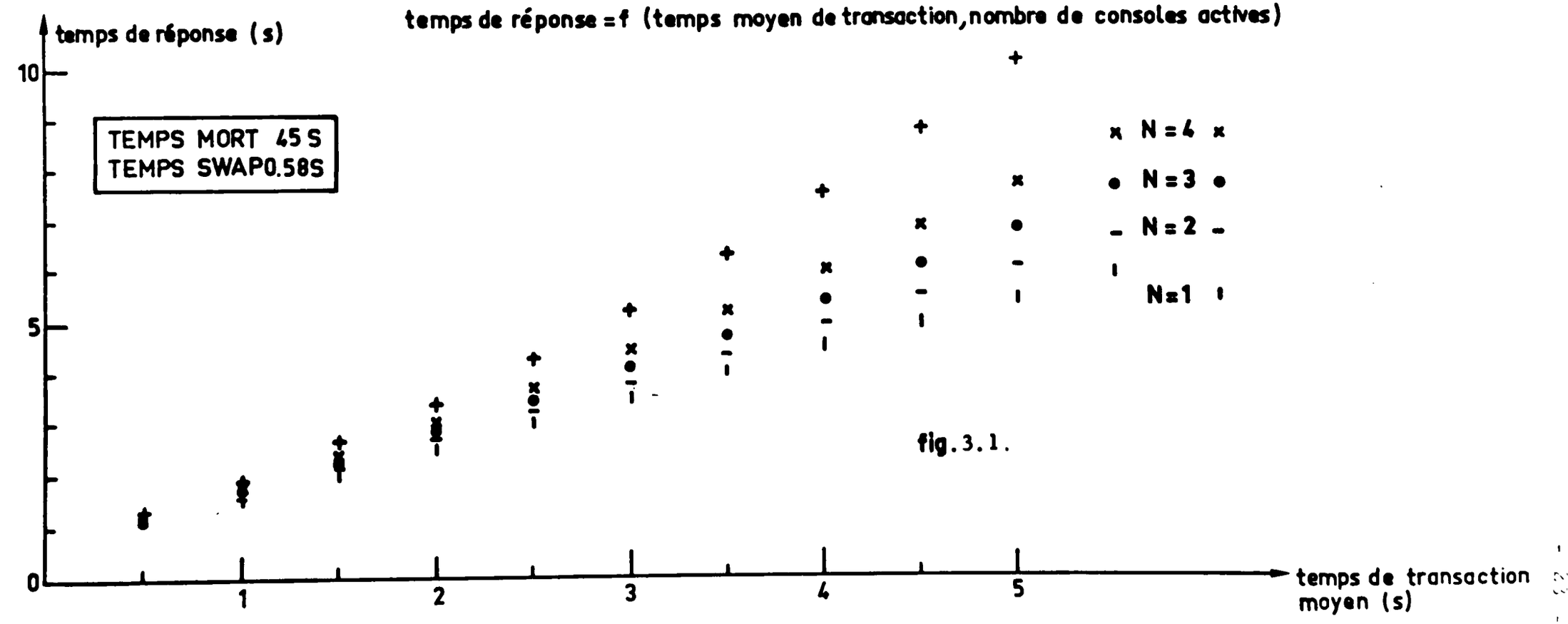
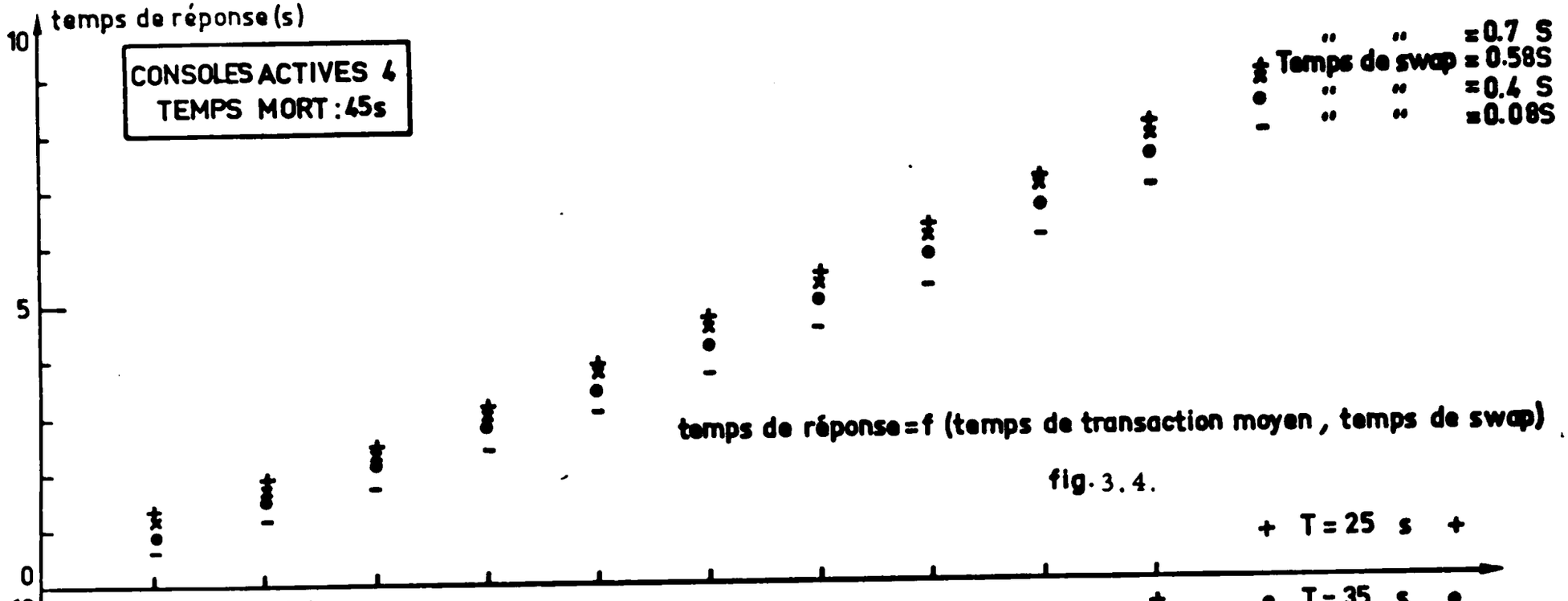
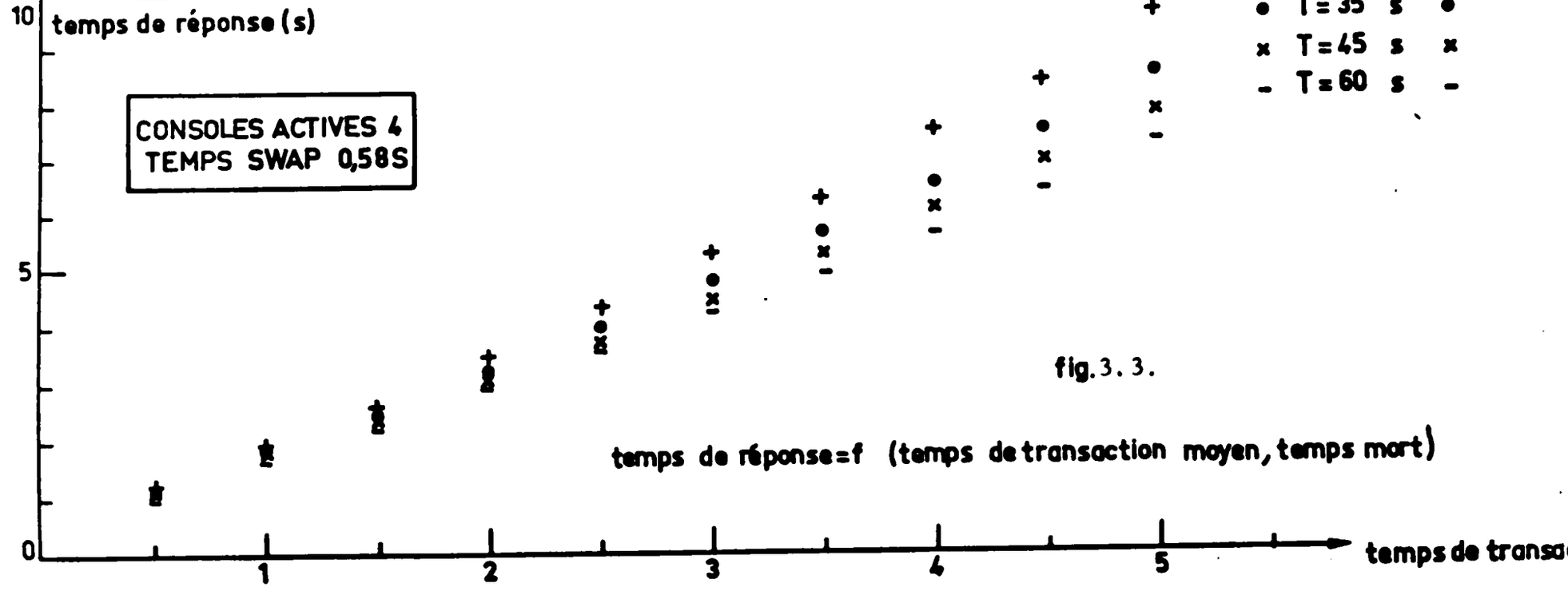


fig. 3.1.



temps de réponse = f (temps de transaction moyen , temps de swap)

fig. 3.4.



temps de réponse = f (temps de transaction moyen, temps mort)

fig. 3.3.

PRESENTATION GENERALE DE L'ORDINATEUR MITRA 15

I - CARACTERISTIQUES PARTICULIERES DU MITRA 15.

I.1. Le système d'interruption. Généralités.

La fontion d'interruption du MITRA 15 est microprogrammée. Ceci permet de réaliser la fonction de multiprogrammation de l'Unité centrale par la possibilité qu'elle a de pouvoir commuter l'Unité de traitement d'une tâche sur une autre plus importante, et d'assurer, ensuite, la reprise de la tâche interrompue.

La fonction d'interruption peut être d'origine interne à l'Unité Centrale (à l'initiative du logiciel), ou bien externe à l'Unité Centrale par la demande d'un usager. Elle nécessite l'exécution des fonctions suivantes :

- Mémorisation de chaque demande d'IT.
- Multiplexage de ces demandes (rang).
- Gestion des priorités d'accès (niveau).
- Commutation de tâches.
- Possibilités de retour.

I.2. Les niveaux d'interruptions.

L'interface périphérique de l'Unité Centrale du Mitra 15 assure le multiplexage matériel de 32 niveaux d'IT, correspondant chacun à un niveau programme. Les 32 niveaux hiérarchiques sont repérés de 0 à 31 dans le sens des priorités croissantes où l'on distingue :

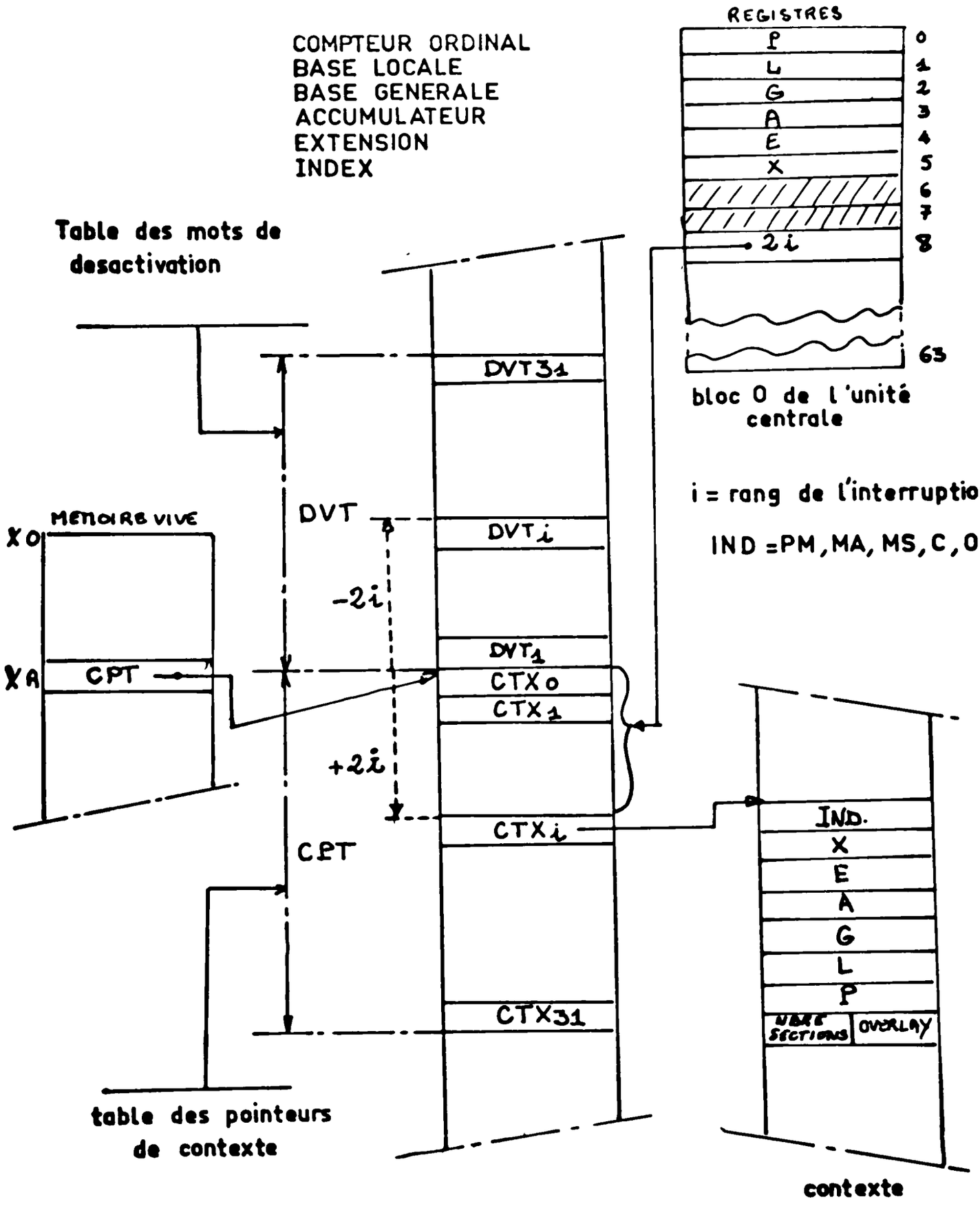
- IT 31 et IT 30 : 2 niveaux internes à l'Unité Centrale relatifs

respectivement à la coupure secteur et au retour secteur.

- IT 29 à IT 01 : 29 niveaux accessibles par le minibus E/S, répartis entre les différents coupleurs. L'affectation des niveaux entre ces différents coupleurs n'est pas rigide et varie avec la configuration du système.

- IT 00 : 1 niveau de fond actif en permanence lorsqu'aucune autre interruption n'est en traitement.

A chacun des 32 niveaux est attachée une adresse mémoire qui contient un pointeur de contexte, désignant le programme spécifique à ce niveau. Les 32 pointeurs de contexte sont rassemblés dans une table CPT dont l'emplacement est indiqué par le contenu du mot d'adresse 10 (en décimal) de la Mémoire Centrale.



IMPLANTATION DES TABLES EN MEMOIRE

Fig. 4.1.

I.3. Contexte programme.

A chaque niveau d'interruption correspond un programme associé de type handler, dévolu à un ou plusieurs périphérique de même nature ou bien un programme utilisateur. Un programme est constitué d'un ensemble d'instructions et de données convenablement rangées en mémoire centrale avant d'être traitées par l'unité centrale.

L'implantation en mémoire et l'état d'avancement d'un programme sont caractérisés par le contenu des registres et indicateurs programme. Ces informations définissent le contexte programme. Dans l'Unité Centrale, les registres programme sont constitués par les six premiers registres rapides du bloc (O).

Les indicateurs accessibles aux programmes et faisant partie de son contexte sont :

- C (Carry) : report ou test opération selon l'instruction qui le positionne.
- O (Overflow) : débordement ou test opération selon la dernière instruction exécutée.
- MS (Indicateur de mode) : à 1 si le programme doit s'exécuter en en mode Maitre ou à 0 dans le cas d'un mode normal.
- MA (Masque d'interruptions) : à 1 pour le masquage des interruptions sinon à 0.
- PR (Protection mémoire).

Le contexte est une table appelée CTX constituée de 8 mots (bien que 7 mots soient parfois suffisants), généralement placée en fin du programme exécutable à l'édition de liens, dont la structure est détaillée en Fig. 4.1.

Cette table CTX permet soit de charger les registres et indicateurs quand on arrive sur ce niveau, soit d'y ranger ces même registres et indicateurs avant de quitter le niveau.

I.3. Contexte programme.

A chaque niveau d'interruption correspond un programme associé de type handler, dévolu à un ou plusieurs périphérique de même nature ou bien un programme utilisateur. Un programme est constitué d'un ensemble d'instructions et de données convenablement rangées en mémoire centrale avant d'être traitées par l'unité centrale.

L'implantation en mémoire et l'état d'avancement d'un programme sont caractérisés par le contenu des registres et indicateurs programme. Ces informations définissent le contexte programme. Dans l'Unité Centrale, les registres programme sont constitués par les six premiers registres rapides du bloc (O).

Les indicateurs accessibles aux programmes et faisant partie de son contexte sont :

- C (Carry) : report ou test opération selon l'instruction qui le positionne.
- O (Overflow) : débordement ou test opération selon la dernière instruction exécutée.
- MS (Indicateur de mode) : à 1 si le programme doit s'exécuter en en mode Maitre ou à 0 dans le cas d'un mode normal.
- MA (Masque d'interruptions) : à 1 pour le masquage des interruptions sinon à 0.
- PR (Protection mémoire).

Le contexte est une table appelée CTX constituée de 8 mots (bien que 7 mots soient parfois suffisants), généralement placée en fin du programme exécutable à l'édition de liens, dont la structure est détaillée en Fig. 4.1.

Cette table CTX permet soit de charger les registres et indicateurs quand on arrive sur ce niveau, soit d'y ranger ces même registres et indicateurs avant de quitter le niveau.

I.4. Prise en compte d'une interruption :

Un programme travaille sous le niveau i , lorsqu'une IT de niveau supérieur j survient, il faut préserver le contexte de la tâche interrompue et le remplacer par le contexte de la tâche de niveau appelant CTX j. La suspension associée aux IT déclenche le microprogramme de changement de contexte. Le microprogramme calcule l'adresse CTX i par :

$$(R8) = 2i \qquad \text{et} \qquad CTXi = (CPT + (R8))$$

et organise les transfert notés I sur la Fig. 4.2

Ensuite le niveau appelant est lu en machine pour remplacer la valeur de R8 par 2j→(R8). Le microprogramme effectue l'opération notée J sur la Fig.4.2. par le calcul d'adresse :

$$CTXj = (CPT + (R8))$$

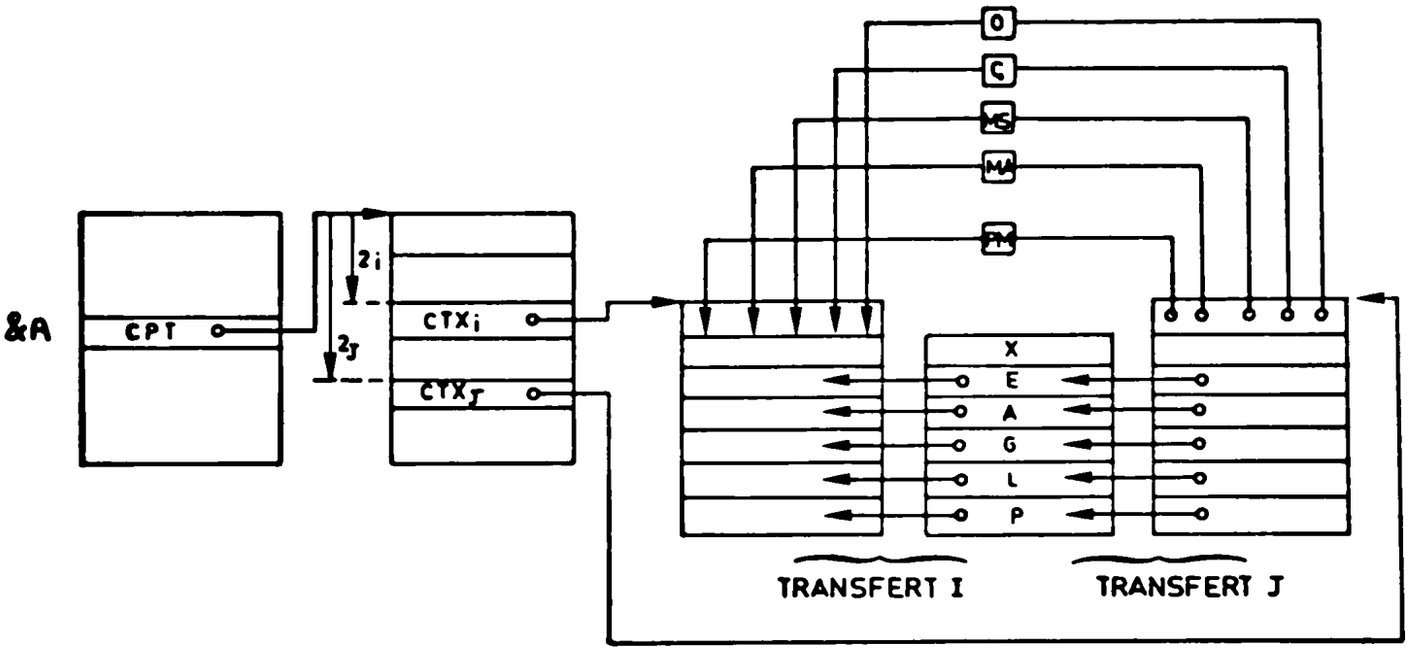


Fig. 4.2.

I.5. Fin de traitement d'un programme associé à un niveau d'interruption.

Une instruction DIT (Désactivation d'IT) termine notre programme de

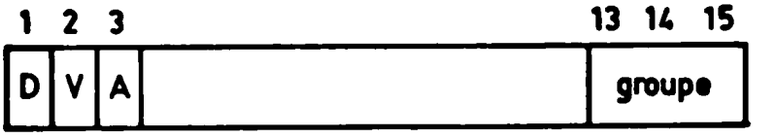
niveau j. La DIT désactive la bascule de déclenchement de l'interruption du coupleur correspondant au niveau j.

Elle effectue un changement de contexte inverse du précédent c'est à dire rangement des registres et indicateurs dans CTXj et remplacement par les valeurs contenues dans CTX i à condition que le niveau i soit toujours le plus prioritaire en attente derrière le niveau j.

La DIT pour effectuer ces opérations fait appel à la table DVT par son microprogramme associé. Cette table DVT est placée immédiatement au-dessus de la table CPT (Fig.4.1.)

L'emplacement de ces mots est directement lié au rang physique de l'interruption du coupleur.

Le mot, CIT, (configuration centrale d'IT) a le format suivant :



- Le bit 0 associé à la CIT a le sens de désactivation.
  D = 0 niveau IT au repos.
  D = 1 en attente ou actif.

- Chaque coupleur de périphérique a un mot (CIT) de désactivation qui lui est associé indépendant du rang physique de l'IT, seul son emplacement dans la table DVT est lié à celui-ci.

Le microprogramme associé à la DIT effectue les opérations suivantes :

On incrémente P tel que P + 2 soit mis dans P.
Puis compte-tenu que R8 contient deux fois le niveau en cours, soit (R8) = 2 j,

- On va chercher le mot CIT associé au niveau j.

- On désactive l'IT avec ce mot et l'on range le contexte j.

- On lit le niveau d'interruption le plus prioritaire en attente derrière j, c'est le niveau i si aucune interruption de niveau intermédiaire n'est apparue entre-temps. On range donc dans R8 deux fois le niveau i et on va chercher le contexte associé au niveau que l'on va mettre dans les registres programme et indicateurs. Connaissant L et P on peut reprendre le programme de niveau i .

On pourra se reporter à la Fig.4.3. pour trouver un exemple de programmation d'une tâche avec un DIT montrant clairement la manière dont la tâche réagit aux interruptions qui suivent le DIT.

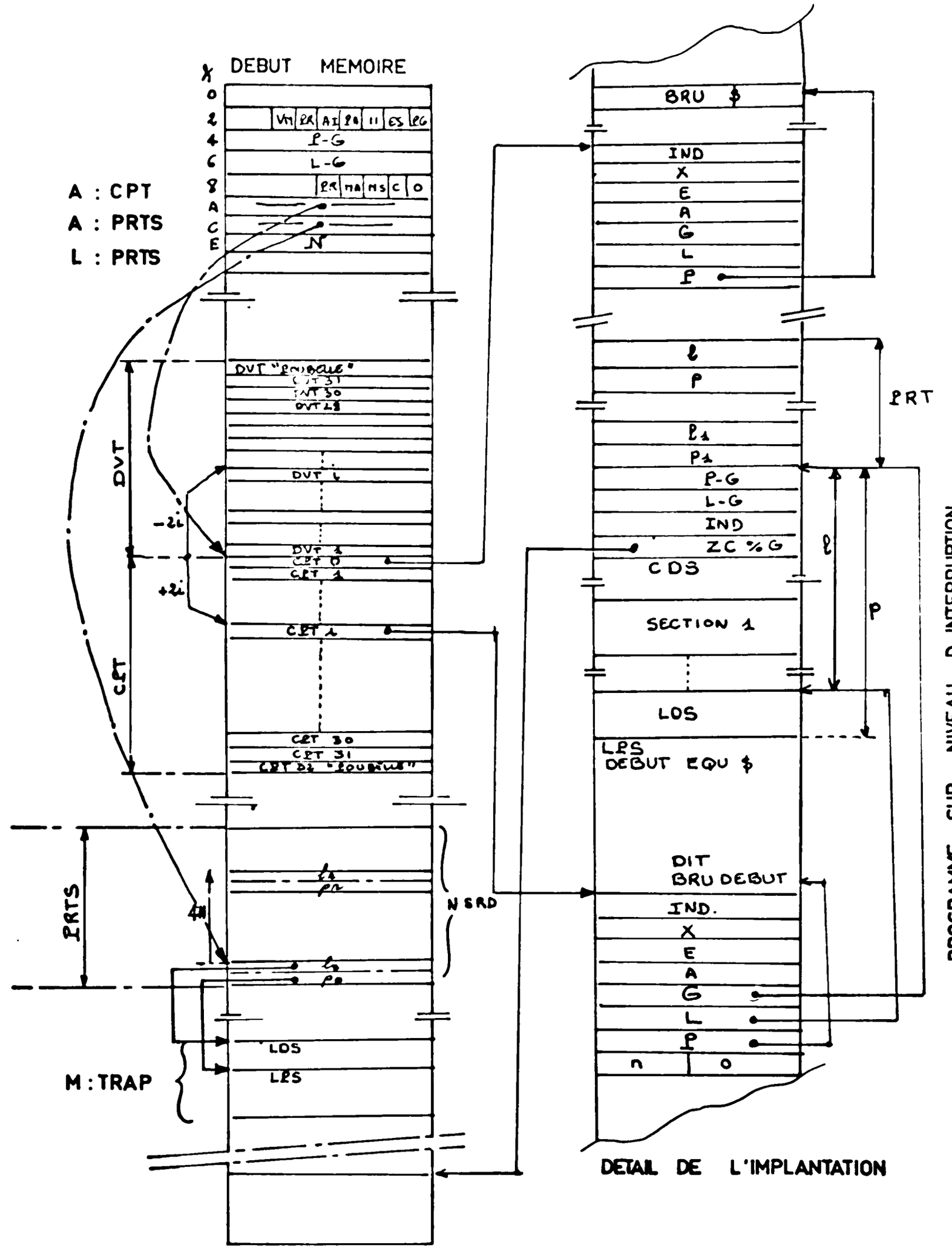

DEBUT MEMOIRE
A : CPT
A : PRTS
L : PRTS
DVT
CPT
PRTS
-2i
+2i
4n
N SRD
M : TRAP
LOS
LPS
BRU $
IND
X
E
A
G
L
P
PRT
P-G
L-G
ZC %G
CDS
SECTION 1
LOS
LPS
DEBUT EQU $
DIT
BRU DEBUT
IND.
PROGRAMME SUR NIVEAU D INTERRUPTION
IMPLANTION DES PROGRAMMES
DETAIL DE L'IMPLANTATION

Fig. 4.3.

REMARQUE.

Sans nous étendre sur l'étude de la logique des interruptions précisons que les instructions dont l'exposé va suivre correspondent à un adressage des niveaux d'IT, pour l'armement, la validation et le déclenchement des interruptions:

- Un niveau d'interruption peut-être armé, validé, ou excité par programme. Cette commande est réalisée par une WD 21.
- Il peut-être désarmé et invalidé par une WD 31 ou encore désactivé par une DIT.

Masque des interruptions :

L'indicateur MA positionné interdit toute prise en compte d'interruptions (instruction STM) sauf celles concernant les coupure et retour secteur.

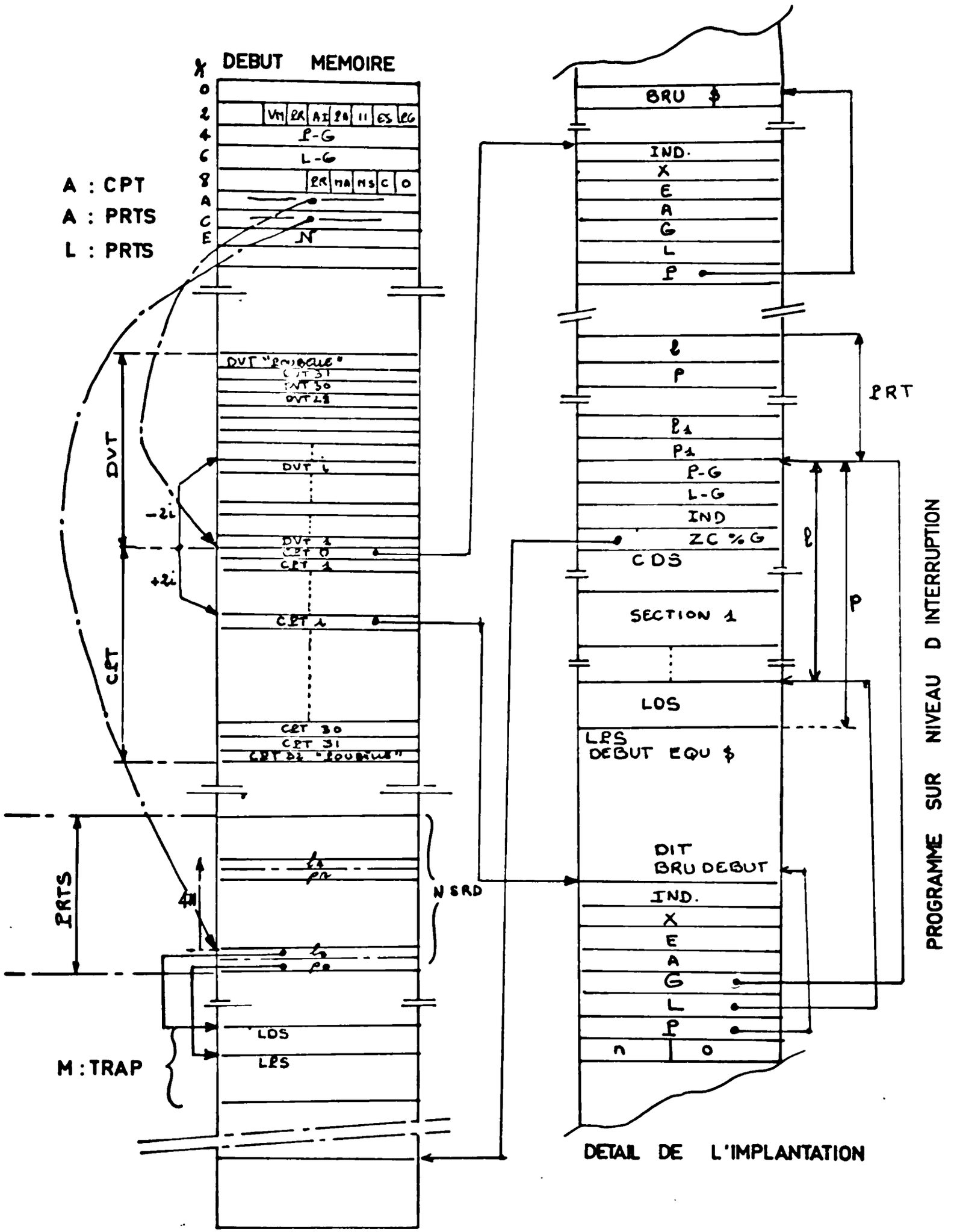
REMARQUE

Pour la clarté de l'exposé nous avons ignoré les regroupements d'IT sur un même niveaux (4 sous-niveaux) dont nous n'avons pas d'utilité dans la réalisation.

II - STRUCTURE D'UN PROGRAMME.

II.1. Définition de la section.

La programmation en langage assembleur Mitra 15 fait largement appel à la modularité. Un programme principal pouvant également être un module (petit élément à interface normalisé), le vocabulaire CII substitue au terme de sous-programme celui de section.



IMPLANTION DES PROGRAMMES EN MEMOIRES

Fig. 4.3.

REMARQUE.

Sans nous étendre sur l'étude de la logique des interruptions précisons que les instructions dont l'exposé va suivre correspondent à un adressage des niveaux d'IT, pour l'armement, la validation et le déclenchement des interruptions;

- Un niveau d'interruption peut-être armé, validé, ou excité par programme. Cette commande est réalisée par une WD 21.
- Il peut-être désarmé et invalidé par une WD 31 ou encore désactivé par une DIT.

Masque des interruptions :

L'indicateur MA positionné interdit toute prise en compte d'interruptions (instruction STM) sauf celles concernant les coupure et retour secteur.

REMARQUE

Pour la clarté de l'exposé nous avons ignoré les regroupements d'IT sur un même niveaux (4 sous-niveaux) dont nous n'avons pas d'utilité dans la réalisation.

II - STRUCTURE D'UN PROGRAMME.

II.1. Définition de la section.

La programmation en langage assembleur Mitra 15 fait largement appel à la modularité. Un programme principal pouvant également être un module (petit élément à interface normalisé), le vocabulaire CII substitue au terme de sous-programme celui de section.

Une section est donc principalement composée d'une "suite d'instructions" appelée segment de programme : LPS.

Ces instructions ont pour objet le traitement des données qui peuvent être propres à une section ou partagées entre plusieurs sections.

Les données propres à une section forment un segment de données locales.

Une section est donc soit la section de données communes (CDS - Common Data Section), soit l'ensemble d'un segment de données locales (LDS -Local Program Segment) et d'un segment de programme exécutable.

De ces définitions on retiendra les points suivants :

- On accède à la CDS à partir de tout le programme, en particulier on accèdera à la CDS à partir d'un LPS en mode général (direct, indirect, indexé), les symboles et étiquettes définis dans la CDS étant communs à tout le programme.

- On accède à un LDS à partir du LPS associé en mode local, (direct, indirect, indexé). Les symboles et étiquettes définis dans un LDS sont locaux à la section mais peuvent être référencés en CDS. Tout autre programme voulant accéder à une donnée locale ne lui appartenant pas doit passer par le mécanisme de référence et définition externes.

- Un segment de programme ne comporte que des éléments non modifiables qui sont les instructions.

II.2. Bases des sections et des segments.

L'étude du contexte programme a montré l'importance de certains registres d'adresses. Leur correspondance avec la structure des programmes est exposée ici.

a) La base générale dite base G

La base G est associée de façon biunivoque au programme ; c'est la base implicite à laquelle toutes les adresses référencées par le programme sont relatives. Elle est ajoutée automatiquement par la micro-machine aux adresses définies dans les instructions.

b) La base locale dite base L

La base L est la base locale implicite ou base des données locales, elle est associée à un segment de données locales.

c) La base programme dite base P

Elle est associée à un segment de programme. A l'initialisation du programme, elle contient l'adresse de début de la section précisée à l'assemblage, puis constitue le compteur ordinal en cours d'exécution de la section.

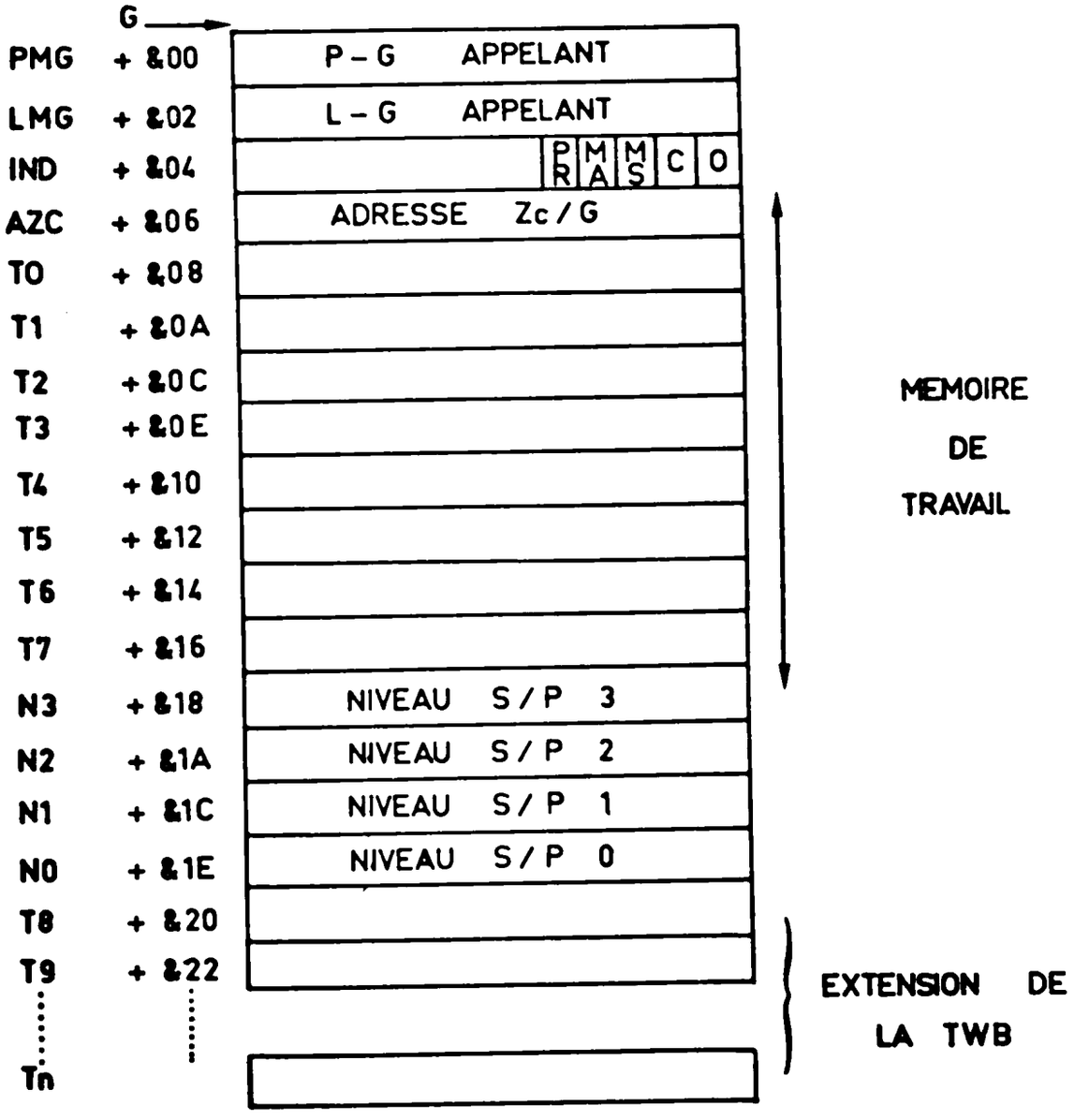
Les valeurs effectives des bases L et P d'une section sont ignorées en général par le programmeur au moment de l'écriture du programme. Elles sont définies automatiquement par l'éditeur de liens en valeur relative à la base générale du programme et stockées dans la table PRT du programme. (Voir plus loin).

La modularité implique l'existence d'instruction spéciales d'appel ou de retour de section dont nous verrons le fonctionnement détaillé plus loin.

Pour faciliter l'écriture des programmes et notamment des sous-

programmes réentrants, l'assembleur reconnaît des "segments fictifs" images de données ultérieures ou de données appartenant à un autre LDS (ou CDS) que celui où la zone future est déclarée : (IDS).

Les segments fictifs jouent le rôle de paramètres formels ; en particulier pour définir des déplacements relatifs au début de ces segments (description de blocs de données dynamiques, définition de valeur d'index, etc...), mais n'engendrent aucun texte objet. Il sera fait un usage intensif de cette notion. La Fig.4.3. reprend les concepts établis dans ce chapitre.



TWB DE LA CDS D'UN PROGRAMME

Fig. 4.4.

II.3. Eléments de communications d'un programme.

a) Bloc de travail (TWB : Task Working Block).

Les premiers mots de la CDS sont obligatoirement réservés au moniteur d'exploitation du Mitra 15, formant le bloc de travail.

Celui-ci pourra y ranger, l'adresse de retour dans la tâche appelante, la base de données locale (L) de l'appelant et les indicateurs programme lors d'un appel superviseur par une tâche.

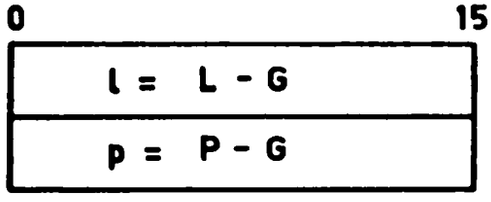
Le moniteur y entretient un pointeur vers la zone de données communes du système (ZC), et enfin, utilise la zone TWB comme mémoire de travail.

C'est ce dispositif qui permet la réentrance des sections moniteur, celles-ci travaillant directement dans le programme appelant.

La longueur de cette zone, habituellement de 16 mots, croît pour des utilisations plus spécialisées (32 mots pour l'utilisation de gestionnaire de fichiers SGF 15, 36 mots pour l'utilisation de lignes asynchrones). La Fig.4.4. donne une description plus détaillée de la TWB, faisant état de la terminologie mnémotechnique en usage dans la programmation du système.

b) Table d'affectation des sections (PRT : Program Relocation Table).

- Une affectation de section est constituée par un double mot qui contient les valeurs initiales de L et P relatives à la base G.



Double mot d'affectation. SRD : Section Relocation Double word.

La PRT est formée de l'ensemble des SRD des sections constituées sur le programme. Elle précède immédiatement l'adresse G de telle sorte que l'adresse du SRD de la section n est égale à G - (4 x) une section étant repérée par un numéro (1, 2, ..., n).

Cette table est constituée à l'édition de liens du programme.

REMARQUE

Le CDS étant accessible de n'importe quelle section, constitue une liaison implicite entre les sections.

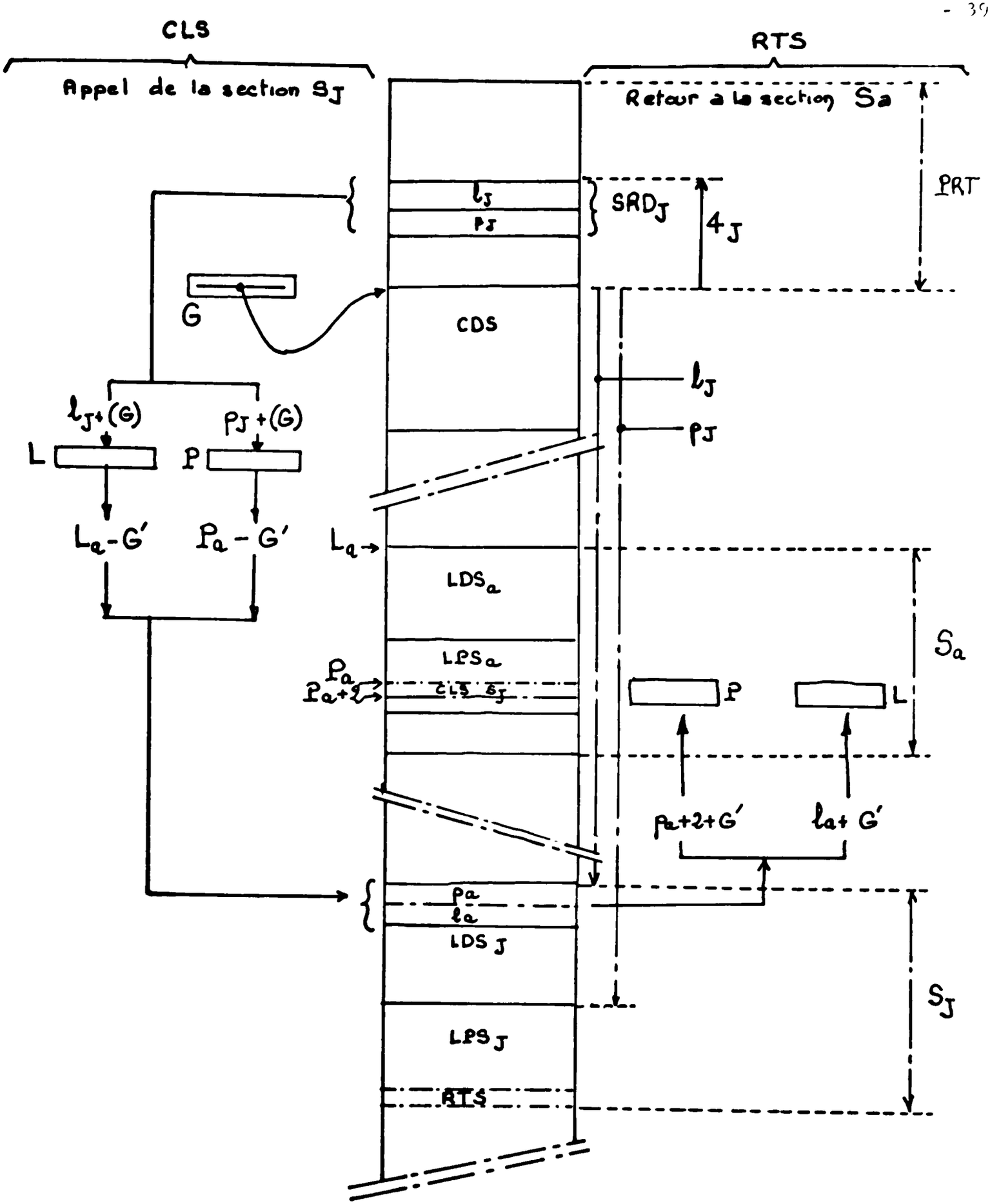

CLS
RTS
Appel de la section S_J
Retour à la section S_a
l_J
p_J
SRD_J
4_J
PRT
G
CDS
l_J
p_J
$l_J+(G)$
$p_J+(G)$
L
P
L_a-G'
P_a-G'
L_a
LDS_a
LPS_a
P_a
P_{a+2}
S_a
P
L
$p_{a+2}+G'$
l_a+G'
p_a
l_a
LDS_J
S_J
LPS_J
RTS

Fig. 4.5.

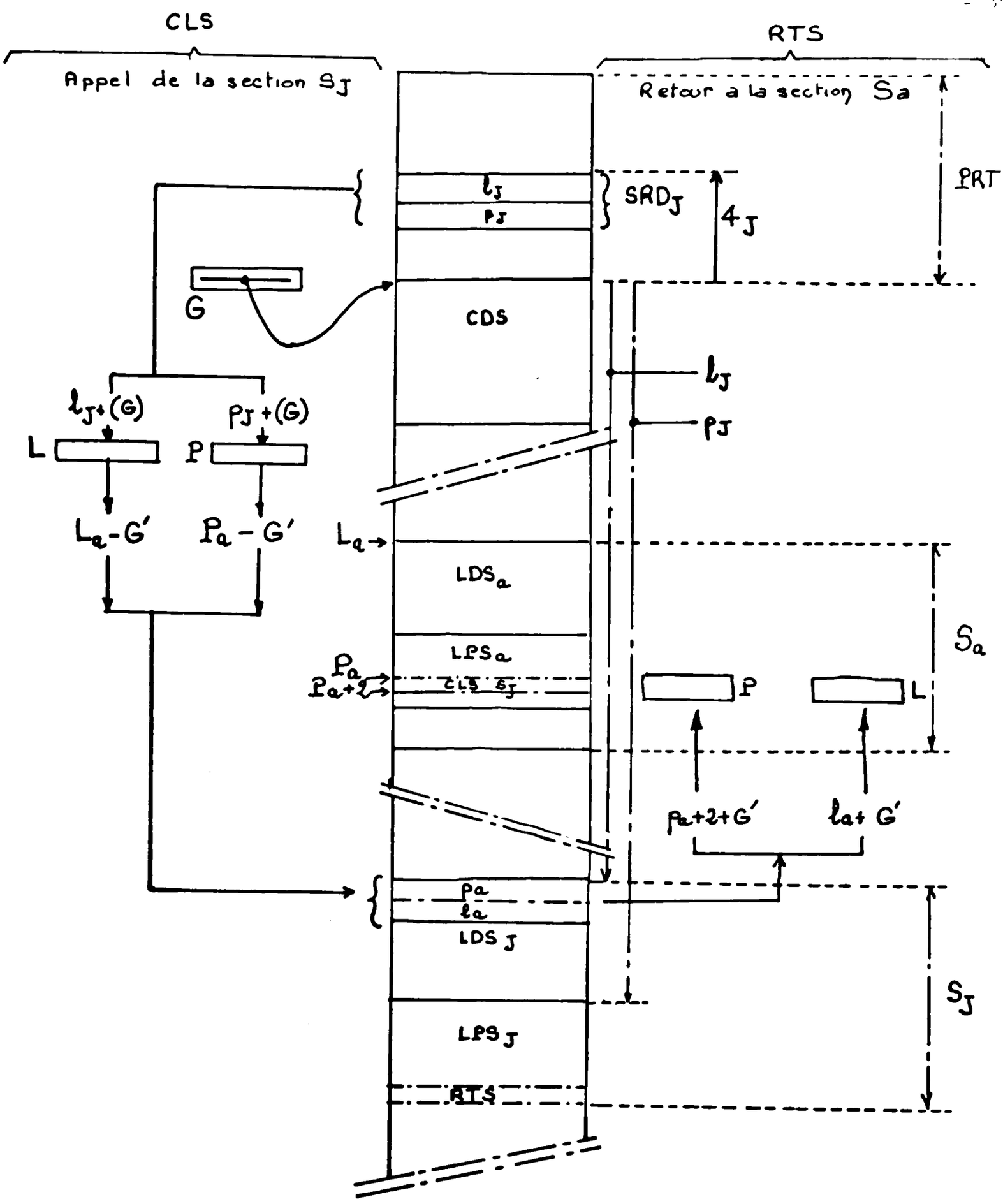

CLS
Appel de la section SJ
RTS
Retour à la section Sa
lJ
PJ
SRDJ
4J
PRT
G
CDS
lJ
PJ
lJ+(G)
PJ+(G)
L
P
La-G'
Pa-G'
La
LDSa
LPSa
Pa
Pa+2
P
L
Sa
Pa+2+G'
la+G'
Pa
la
LDSJ
SJ
LPSJ
RTS

Fig. 4.5.

II.4. Appel de sections.

La section apparaît sous deux formes :

- La section propre à un programme, su'on atteint par le Call Section (CLS).
- La section disponible pour tous les programmes : section superviseur qu'on atteint par le Call Superviseur (CVS). Cette instruction sera particulièrement étudié plus loin.

Le but du CLS est donc d'effectuer un branchement entre une section (appelante) et une autre d'un même programme (appelée) tout en assurant des possibilités de communications permanentes entre la section appelée et la section appelante ainsi qu'un retour simple.

A l'exécution d'une instruction CLS, la micro-machine :

- Range les contenus de P et L dans les deux premiers mots de la zone locale de la section appelée après en avoir soustrait la quantité G. Ces éléments sont nécessaires au retour et le programmeur doit réserver le double mot en tête des données locales.

- Charge P et L à l'aide du SRD correspondant à la section appelée avec les adresses respectives du point d'entrée et du segment de données locales.

Le retour se fera à l'aide de l'instruction RTS.

La fonction est donc :

$(P) - G' \longrightarrow (((G) - 4N) + (G))$

$(L) - G' \longrightarrow (((G) - 4N) + (G) + 2$

] Rangement pour retour

$(G) + ((G) - 4N) \longrightarrow (L)$

$(G) + ((G) - 4N + 2 \longrightarrow (P)$

] Branchement section appelée.

La protection des registres L et P sur le CLS se fait en absolu en mode maître ($G' = 0$) et en relatif par rapport à la base G en mode esclave ($G' = (G)$) .

II.5. Retour de section.

Le RTS situé dans une section appelée par CLS assure le retour à la section appelante en rétablissant dans les registres L et P les valeurs correspondantes rangées par le CLS d'appel, dans les deux premiers mots du LDS de la section appelée tout en assurant le branchement de retour sur l'instruction qui suit le CLS.

La fonction est :

$((L)) + G' + 2 \longrightarrow (P)$ Branchement derrière le CLS

$((L) + 2) + G' \longrightarrow (L)$ Restitution ancienne base L.

Les indicateurs programmes (C, O) sont ceux de l'instruction qui précède le RTS.

Le fonctionnement des instruction CLS, RTS est exposé en Fig. 4.5.

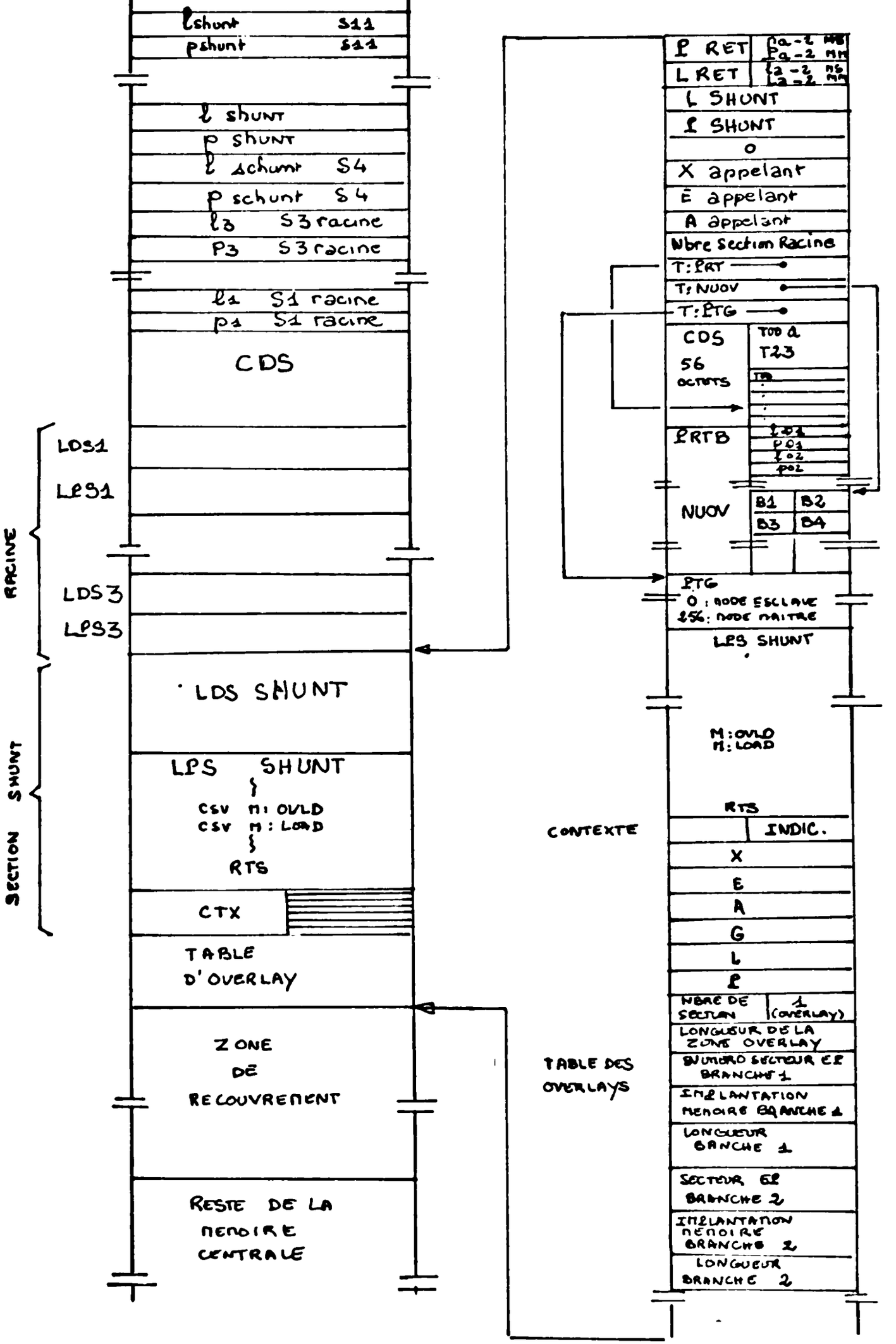
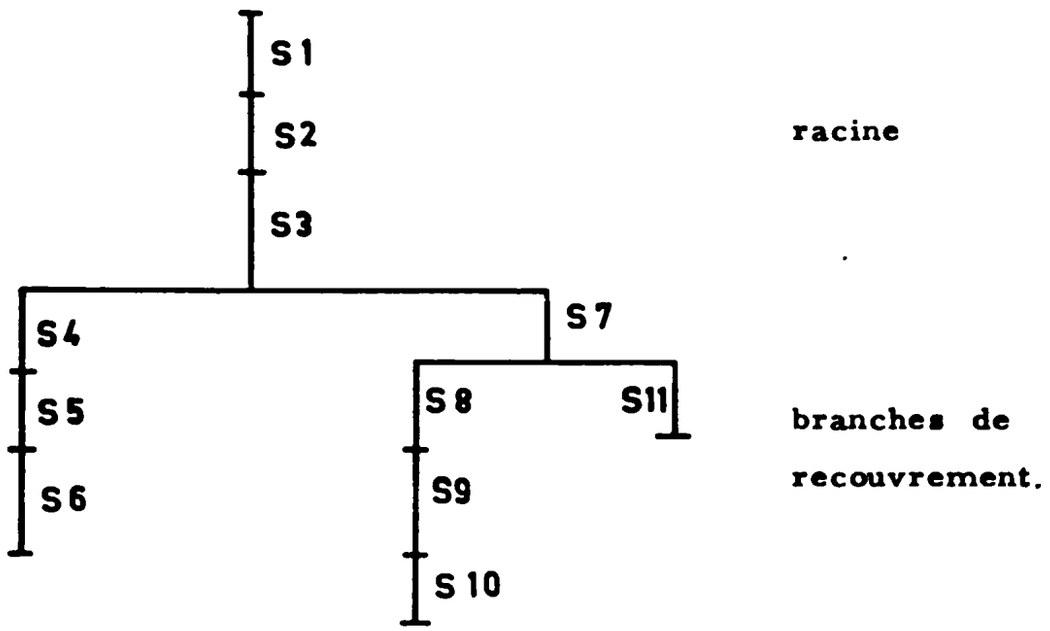


Fig. 4.6. STRUCTURE DE RECOUVREMENT EN MEMOIRE CENTRALE

II.6. La structure de recouvrement.

C'est une extension particulière de la communication entre modules, affectée à la gestion des programmes dont la taille est telle que la technique de recouvrement devienne nécessaire.

C'est l'éditeur de liens qui a la charge de créer la structure finale dont le programmeur lui fournit la description lors de l'édition de liens. Soit par exemple l'arborescence suivante, les sections étant disponibles dans les différentes bibliothèques de programmes sous forme de binaire translatable :



L'éditeur de liens en cours de travail crée un segment de programme associé à une LDS appelé SHUNT. La PRT est fabriquée de telle façon que les SRD des sections en branche pointe en fait sur ce module (SRDsh), à la différence du SRD des sections en racine. L'IMT qui résulte présentera lors de son chargement en mémoire, l'aspect décrit par la Fig.4.6., et la zone disponible derrière la table des "Overlays" servira à la zone de recouvrement dont on précise la longueur maximale.

En cours de traitement, si une section de la racine appelle une section en branche l'instruction CLS provoquera une "déroutement" (par le mécanisme déjà vu) dans le programme SHUNT ; cette procédure est chargée de rendre l'appel à la nouvelle section transparent pour la section appelante. La section SHUNT fait intervenir le module moniteur M : OVLD.

Ce module effectue le travail préliminaire suivant :

- Fait moins 2 sur le mot de sauvegarde de P (1er mot de la LDS de SHUNT).
- Met dans la PRT, la SRD réelle de la section appelée qui se calcule à partir des tables privées de la LDS de SHUNT, et assure partiellement la transformation des SRD des sections appartenant à la même branche.
- Remet tous les autres SRD de la PRT (sauf ceux de la racine et ceux dont les sections ne sont pas écrasées en mémoire par la branche complète) à la valeur SRD sh.

La branche à laquelle appartient la section visée est alors descendue en mémoire. L'instruction RTS de la section Shunt, provoque le retour sur l'instruction qui avait provoqué le déroutement par l'instruction CLS d'appel à la section en branche dans la section appelante. L'aiguillage s'effectuant alors de façon naturelle dans la section appelée, cette fois-ci rendue disponible en mémoire.

L'appel à Shunt est rendu transparent par recopie de la TWB originelle dans une zone de LDS de Shunt. Les registres d'appels sont sauvegardés de même. Le tout est restitué avant l'instruction RTS.

En conclusion de cette étude succincte sur le fonctionnement de la structure de recouvrement, il faut remarquer que le mécanisme n'est pas réentrant du fait que des données évolutives sont disponibles dans la LDS de Shunt.

III - PRINCIPES DE GESTION DU SYSTEME D'EXPLOITATION.

III.1. Structure du système et organisation de la mémoire.

III.1.1. Structure du noyau résident.

Le noyau résident est implanté à partir de l'adresse 0 de la mémoire vive.

Il comporte :

a) Des données système, organisées au sein de segments de données locales appelées dans la terminologie CII "page" ; on y distingue :

- adresses des tables système.
- Zone de sauvegarde sur déroutement.
- Adresses des tables système.
- Adresses de communication entre les modules superviseur.
- Constantes système (pointeurs de contexte, tables d'entrées-sorties, etc...)

b) Des sous-programmes communs (en modules superviseur)

Ce sont des modules accessibles à l'utilisateur par CSV ou bien des modules strictement internes sous forme de segments de programmes associés à un segment de données locales précisé plus haut.

c) Des handlers.

Ce sont les modules de contrôle physique des transferts sur périphériques.

Ils constituent chacun une "tâche immédiate" connectée à l'interruption associée du périphérique. Ils sont disponibles sous forme d'IMT intégrables à la génération.

d) Les tâches immédiates.

Ce sont des modules de programmes liés à un niveau d'interrruption.

On peut distinguer :

- Le programme de traitement de l'interruption pupitre.
- Le programme de traitement de l'interruption horloge.
- Les programmes de traitement des interruptions coupure secteur et retour secteur.
- Les tâches immédiates utilisateur incluses à la génération. C'est une utilisation très spéciale car l'utilisateur peut utiliser la zone foreground pour ses tâches immédiates.

III.1.2. Organisation de la mémoire.

La Zone Moniteur.

A partir de l'adresse zéro jusqu'à la fin du noyau résident y compris les tâches immédiates de type d) ci-dessus. Le moniteur se présente donc comme un programme particulier, écrit en mode maître.

La Zone "Foreground".

C'est une zone s'étendant depuis la fin du noyau résident jusqu'au début de la zone "Background". Cette zone dynamique reçoit les tâches immédiates utilisateur connectées à un niveau d'interruption. Les tâches chargées dans cette zone peuvent être en mode maître ou en mode esclave.

La Zone "Background".

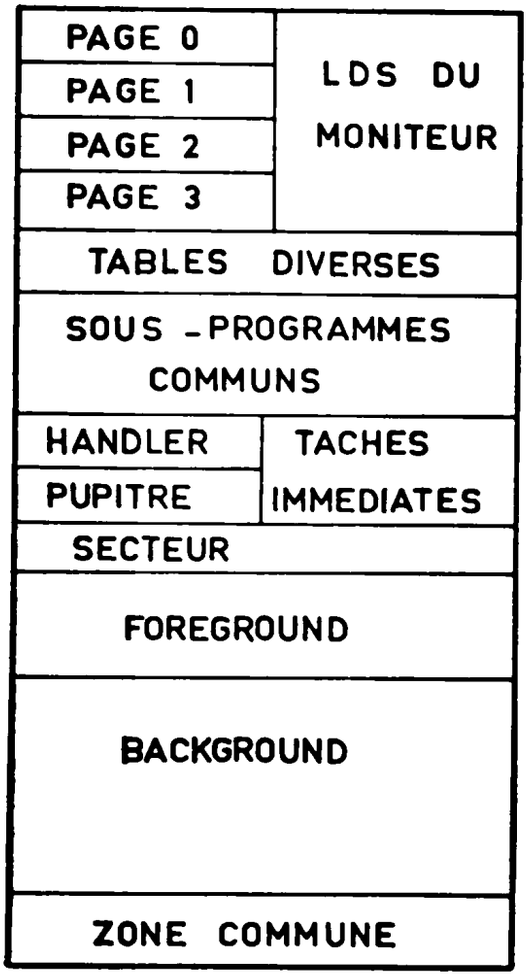
Elle s'étend depuis la fin de la zone "Foreground"

jusqu'au début de la zone commune. Elle est réservée au programme associé au niveau zéro, normalement unique.

La Zone Commune.

Elle s'étend depuis la fin de la zone "Background" jusqu'à la fin de la mémoire, et permet une communication entre des programmes distincts. Son adresse relative à la base G du programme, est disponible dans le quatrième mot de la CDS de chaque programme utilisateur.

Le schéma de la mémoire devient donc :



III.2. Gestion du système d'exploitation.

Une section système, ou module moniteur, reste au niveau de priorité du programme appelant. Il travaille à la fois sur les données de la tâche appelante et sur ses données locales propres. De plus, la base G identifiant une tâche (adresse d'implantation en mémoire), l'appel est associé à la base G. L'instruction CSV (Call Superviseur) fait jouer à G le même rôle qu'avait la base L dans l'instruction CLS.

Une section système peut :

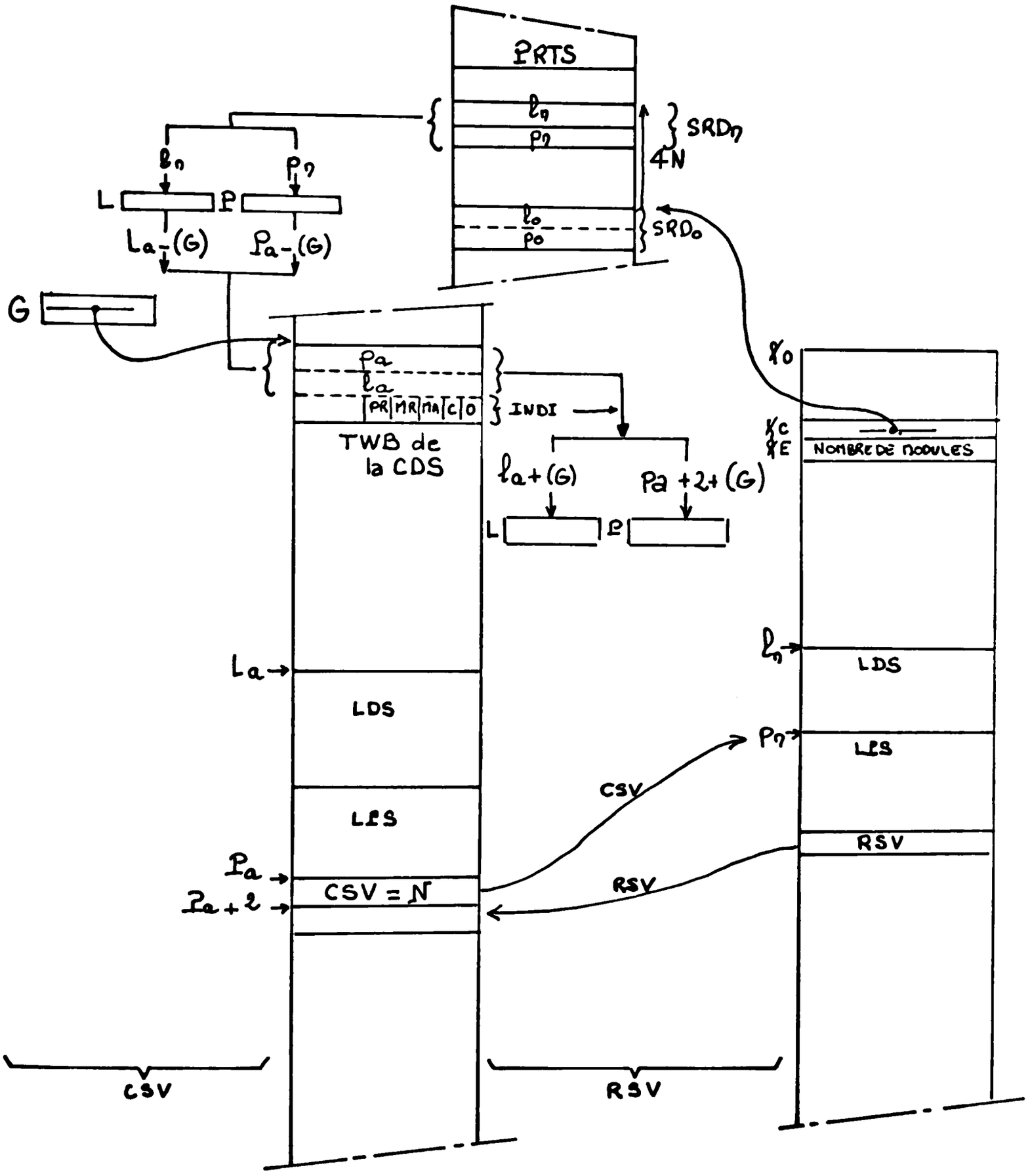
- Accéder à ses données propres en mode DL, IL, ILX, ces données ayant des adresses absolues. La majorité des modules moniteurs a une base nulle, ils travaillent pratiquement tous sur la zone de données formant la LDS du superviseur.

- Accéder aux données de la tâche appelante en mode DG et IGX puisque la base G reste celle de la tâche appelante ; cet adressage permet la réentrance des modules moniteur car le moniteur entretient ses zones de travail dans la TWB du programme appelant.

  Par le biais de la multiprogrammation matérielle, le moniteur est toujours assuré de retrouver l'état complet d'un travail si le traitement actuel est interrompu.

III.2.1. L'appel moniteur CSV.

Chaque section du moniteur est caractérisée par une valeur L et P. Ces valeurs en adresse absolue sont stockées dans la PRTS (PRT particulière du programme superviseur) situé, à la différence de la PRT, à un emplacement quelconque de la mémoire.


PRTS
SRDη
4N
SRDo
L
P
G
INDI
TWB de
la CDS
NOMBRE DE MODULES
LDS
LES
RSV
CSV = N
CSV
RSV

CVS RSV

F. 4.7.

Elle est pointée par l'intermédiaire du mot d'adresse absolue & C. Le nombre de sections moniteur, donc le nombre de SRD (double-mots) est initialisé dans le mot d'adresse absolue &E et sert au contrôle du nombre de sections et du numéro de la section appelée. (si le numéro d'une section appelée dépasse ce nombre maximum il y a déroutement pour violation de mode). Signalons que le module moniteur numéro 0 est réservé au traitement des déroutements.

En valeurs relatives à la base G, les bases P et L au moment de l'appel d'une section moniteur sont rangées avec les indicateurs programme dans les trois premiers mots de la CDS de la tâche appelante. Les indicateurs Mode Maître et Protection mémoire sont forcés à 1. Les registres P et L sont initialisés par le SRDN se trouvant à l'adresse (&C) - 4N ou N est le numéro du module moniteur appelé.

Ces opérations se décrivent ainsi :

(P) - (G) ⟶ ((G))
(L) - (G) ⟶ ((G) + 2) Protection à l'appel.
Indicateurs ⟶ ((G) + 4)

1 ⟶ PR Forçage des indicateurs
1 ⟶ PM

((&C) - 4N) ⟶ (L)
((&C) - 4N + 2) ⟶ (P)

REMARQUE.

Il n'y a pas de point interruptible entre l'instruction CSV et l'instruction suivante qui appartient à la section superviseur appelée.

Ceci permet l'exécution d'une instruction STM évitant l'interruption d'une tâche non réentrante du superviseur.

III.2.2. Le retour moniteur RSV.

Le RSV, situé dans une section moniteur (mode maître) appelée par unCSV assure le retour à la section appelante en rétablissant dans les registres L et P les valeurs correspondantes préservées par le CSV de l'appel dans les deux premiers mot de la CDS de l'appelant et rétablit les indicateurs d'appel.

Sa fonction peut se décrire ainsi :

((G) + 4) ⟶ Indicateurs

((G) + ((G) + 2) ⟶ (L)

(G) + 2 + ((G)) ⟶ (P)

Restitue l'ancienne base effectue le branchement derrière le CSV par progression du compteur ordinal

La Fig.4.7. reprend ces notions.

III.2.3. L'appel interne.

Afin de permettre à un module moniteur d'en appeler un autre deux appels CSV ne peuvent se succéder, car le second appel écraserait alors les trois premiers mots de la CDS préservés par le premier CSV.

Les pointeurs d'entrée dans les modules moniteur sont initialisés dans les LDS du superviseur, pour permettre l'appel interne.

L'appel interne s'effectuera donc par branchement indirect avec rangement de retour dans la TWB de l'appelant au moyen de l'instruction SPA (Store Program Adress), rangement de l'adresse programme.

Ceci permet l'exécution d'une instruction STM évitant l'interruption d'une tâche non réentrante du superviseur.

III.2.2. Le retour moniteur RSV.

Le RSV, situé dans une section moniteur (mode maître) appelée par unCSV assure le retour à la section appelante en rétablissant dans les registres L et P les valeurs correspondantes préservées par le CSV de l'appel dans les deux premiers mot de la CDS de l'appelant et rétablit les indicateurs d'appel.

Sa fonction peut se décrire ainsi :

((G) + 4) ⟶ Indicateurs

((G) + ((G) + 2) ⟶ (L)

(G) + 2 + ((G)) ⟶ (P)

Restitue l'ancienne base effectue le branchement derrière le CSV par progression du compteur ordinal

La Fig.4.7. reprend ces notions.

III.2.3. L'appel interne.

Afin de permettre à un module moniteur d'en appeler un autre deux appels CSV ne peuvent se succéder, car le second appel écraserait alors les trois premiers mots de la CDS préservés par le premier CSV.

Les pointeurs d'entrée dans les modules moniteur sont initialisés dans les LDS du superviseur, pour permettre l'appel interne.

L'appel interne s'effectuera donc par branchement indirect avec rangement de retour dans la TWB de l'appelant au moyen de l'instruction SPA (Store Program Adress), rangement de l'adresse programme.

Cette instruction effectue la fonction :

(P) + 4 - G' ⟶ y

ou y est la fonction d'adressage et

G' = 0 en mode maître et

G' = (G) en mode esclave.

Le contenu du registre P augmenté de 4 constitue alors l'adresse de retour du sous-programme, car l'instruction est normalement suivie par un branchement au dit sous-programme.

Les 4 mots baptisés N0, N1, N2, N3 dans le TWB permettent aux modules du moniteur une profondeur d'appel interne de 4 niveaux successifs.

Les modules moniteur sont de la forme suivante :

```
M : MOVE      LPS  PAGEO
    SPA   #N0                 entrée sur CSV.
    BRU   $+2
    RSV
R : MOVE      EQU  $          entrée par appel interne
    .
    .
    .
    BRU @#N0                  retour à l'appelant.
```

Et on trouvera dans la LDS du superviseur :

```
P : MOVE  DATA   R : MOVE : pointeur pour appel
                            interne.
```

Les appels de ce module seront de la forme :

CSV M : MOVE	pour l'appel par une tâche
SPA #N0	pour l'appel par un module moniteur.
BRU @P : MOVE	

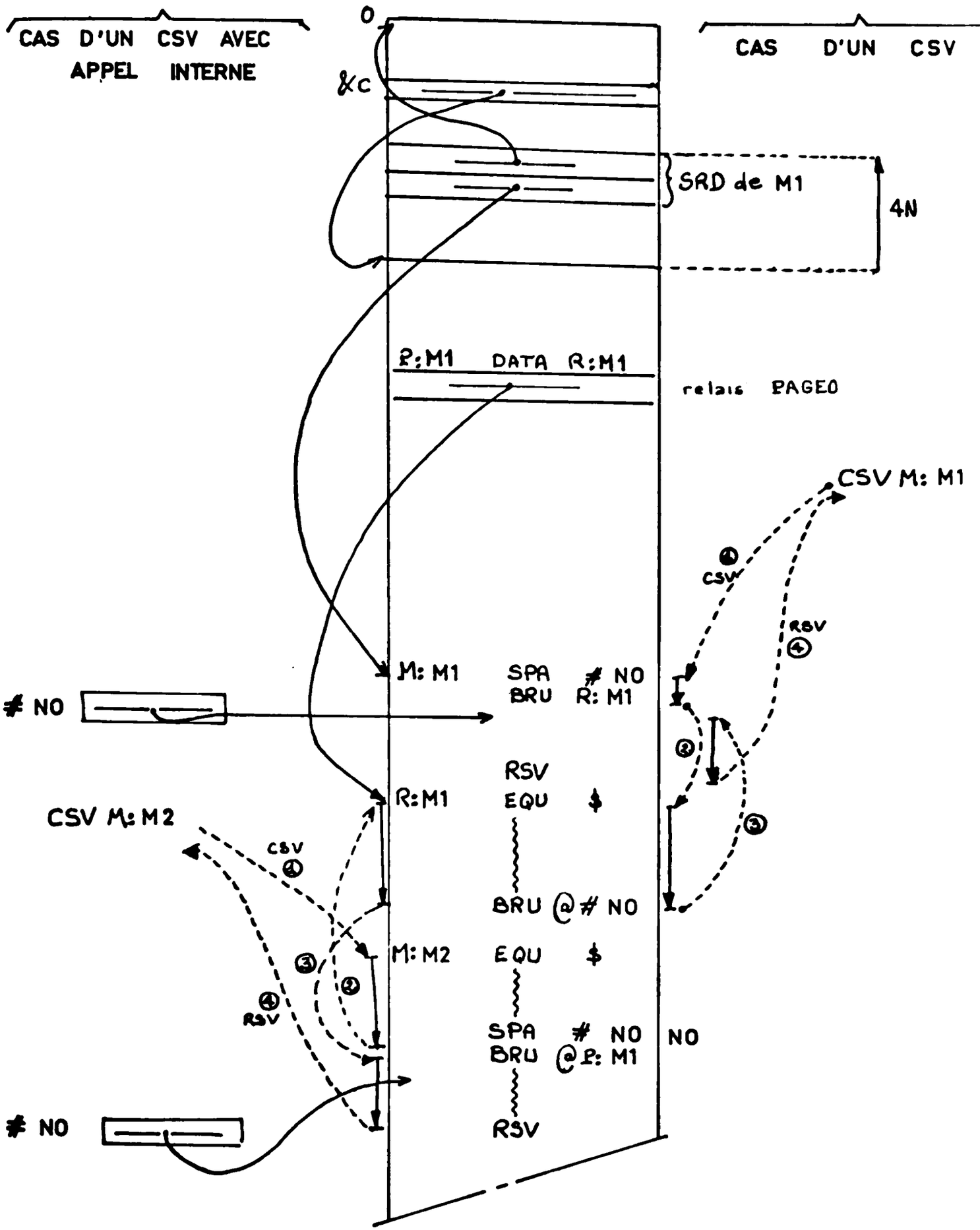
Le retour à l'appelant se fait par un branchement indirect sur N0 (BRU @ # N0),

Ce mot contient alors :

- L'adresse de RSV si l'appel a été fait par CSV.
- L'adresse de l'instruction qui suit le branchement indirect d'entrée dans le module en cas d'appel interne.

La LDS notée PAGE0 correspondant à une base L nulle, les modules moniteurs appelant le module M : Move doivent avoir la base L nulle au moment du branchement par BRU P : MOVE pour l'appel interne.

La Fig.4.8. montre une configuration d'appel simple, les pointillés représentant le "chemin suivi".



CAS D UN CVS AVEC APPEL INTERNE

Fig. 4.8.

REMARQUE :

On peut toutefois remarquer dans certains modules des appels internes réalisés par CSV. Il suffit alors pour régler le problème de protection des informations de l'appel CSV d'origine, de protéger les trois premiers mots de la CDS au sein de la CDS par recopie et au retour de restituer ces mots à leur place. Ceci ne peut s'entendre que s'il est acquis que le module moniteur ainsi appelé n'écrase pas les mots de travail de la CDS où sont sauvegardées les informations.

La structure générale en mémoire du moniteur est représentée en Fig. 4.3 . . On peut y remarquer notamment le module traitant des déroutements nécessitant :

- Une table de 4 mots réservés à partir de l'adresse 2 pour permettre à la micro-machine de ranger le mot d'état déroutement, P-G, L-G et les indicateurs au moment du déroutement.
- Le pointeur en &C désignant le SRD associé au module de traitement des déroutements afin d'initialiser les registres L et P.

Ces informations sont donc disposées dans la LDS PAGE0 du système débutant à L = 0.

III.3. Conclusion.

Dans cette étude nous avons dégagé les principaux points permettant de comprendre le mécanisme de travail du moniteur d'exploitation des Mitra 15.

La notion de réentrance toujours présente, que nous avons justifiée, est

en fait insuffisante dans les cas où, le mécanisme de multiprogrammation jouant, des modules moniteurs sont appelés à travailler sur des données non réentrantes, notamment certaines tables situées dans l'une des LDS du superviseur. Le conflit d'accès est alors réglé par le jeu de ressources système dont nous n'étudierons pas le mécanisme.

Enfin, cette étude nous permet d'aborder la section concernant le traitement des entrées/sorties.

IV- TRAITEMENT DES ENTREES/SORTIES.

IV.1 Généralités

Une entrée/sortie peut se décomposer en cinq phases.

a) Phase 1 : Réservation éventuelle d'un tampon et préparation des données nécessaires à l'accomplissement de la transaction.
b) Phase 2 : Demande de transfert.
c) Phase 3 : Initialisation du transfert.
d) Phase 4 : Déroulement du transfert.
e) Phase 5 : Fin de transfert et contrôle de validité.

Les Phases 2 et 3 sont assurées par le moniteur après l'appel au module spécialisé M : IO. Le retour au programme appelant est fait après cette initialisation.

Les deux dernières phases (4 et 5) sont assurées par le système d'une façon totalement indépendante du programme appelant, en simultanéité apparente avec le programme en cours d'occupation de l'unité Centrale, le programme appelant pouvant suivre l'évolution du transfert s'il le désire.

Si lors de la demande de transfert, le périphérique est occupé, la demande est mise en file d'attente et le contrôle est rendu au programme appelant.

Pour l'utilisateur tout se passe comme si la phase d'initialisation 3 avait eu lieu : il peut considérer que pour lui, la phase 4 commence, et se mettre éventuellement en attente de la fin du transfert.

Dans ce cas, la phase 4 comprend à la fois l'attente de libération logique du périphérique, et l'initialisation et le déroulements effectifs du transfert.

L'organisation générale du traitement des Entrées/sorties est complétée par les paragraphes suivants.

IV.2. Module de controle d'Entrée/sortie : "Handler".

Dans ce qui suit, nous appellerons coupleur l'ensemble :

Coupleur physique = micro-programme de couplage.

L'initialisation effective du transfert de vers le périphérique, le contrôle de cette initialisation, le suivi éventuel du transfert et la prise en compte de la fin de transfert dépendent du type de périphérique, c'est à dire du coupleur. Ces traitements sont réalisés par un module de contrôle spécifique du coupleur, c'est ce module de contrôle spécialisé qu'on appellera "Handler".

Il se décompose en deux éléments :

- handler 1 ou "handler d'initialisation".
- handler 2 ou "handler de contrôle du transfert.

Handler 1 : initialisation d'un transfert.

Il effectue :

a) S'il y a lieu, une analise de l'état initial du coupleur et/ou du périphérique en ce qui concerne les conditions particulières au type de périphérique considéré, avant de commander le transfert. En cas d'anomalie, le transfert est considéré comme terminé et les conditions d'erreurs sont transmises au superviseur d'Entrée/Sortie.

b) L'envoi d'une commande de transfert de bloc ou une lecture/écriture de donnée, suivant le fonctionnement du coupleur et de handler 2.

c) La vérification de la bonne prise en compte de cette initialisation par le coupleur pour les condition d'anomalies qui ne se traduisent pas par une interruption. En cas d'anomalie on est ramené au cas "a".

d) Retour au programme appelant.

Le handler 1 travaille toujours au niveau de priorité du programme appelant.

Le programme appelant est généralement le superviseur d'entrée/sortie, mais ce peut être le handler 2 en cas de relance automatique des transferts.

Handler 2 : contrôle du transfert.

C'est un programme immédiat, en mode maître, déclenché par

Les interruptions du coupleur qui effectue :

- La prise en compte sur interruption des phases du transfert.
- La relance de handler 1.
- Le contrôle de la fin du transfert.

Il travaille toujours au niveau de priorité de l'interruption de couplage. En fin de transfert (anormal ou non), le handler 2 fait un appel au superviseur d'entrée/sortie pour lui signaler la fin de transfert et les conditions de validité du transfert, avant de désactiver le niveau d'IT associé.

REMARQUE.

Le handler 2 pourra traiter en simultanéité apparente plusieurs coupleurs de même type. En effet les IT correspondantes seront regroupées sur un même niveau avant d'arriver à la machine, de sorte que lorsqu'une IT occupera le handler 2, les autres resteront en attente de celui-ci.

IV.3. Les transferts.

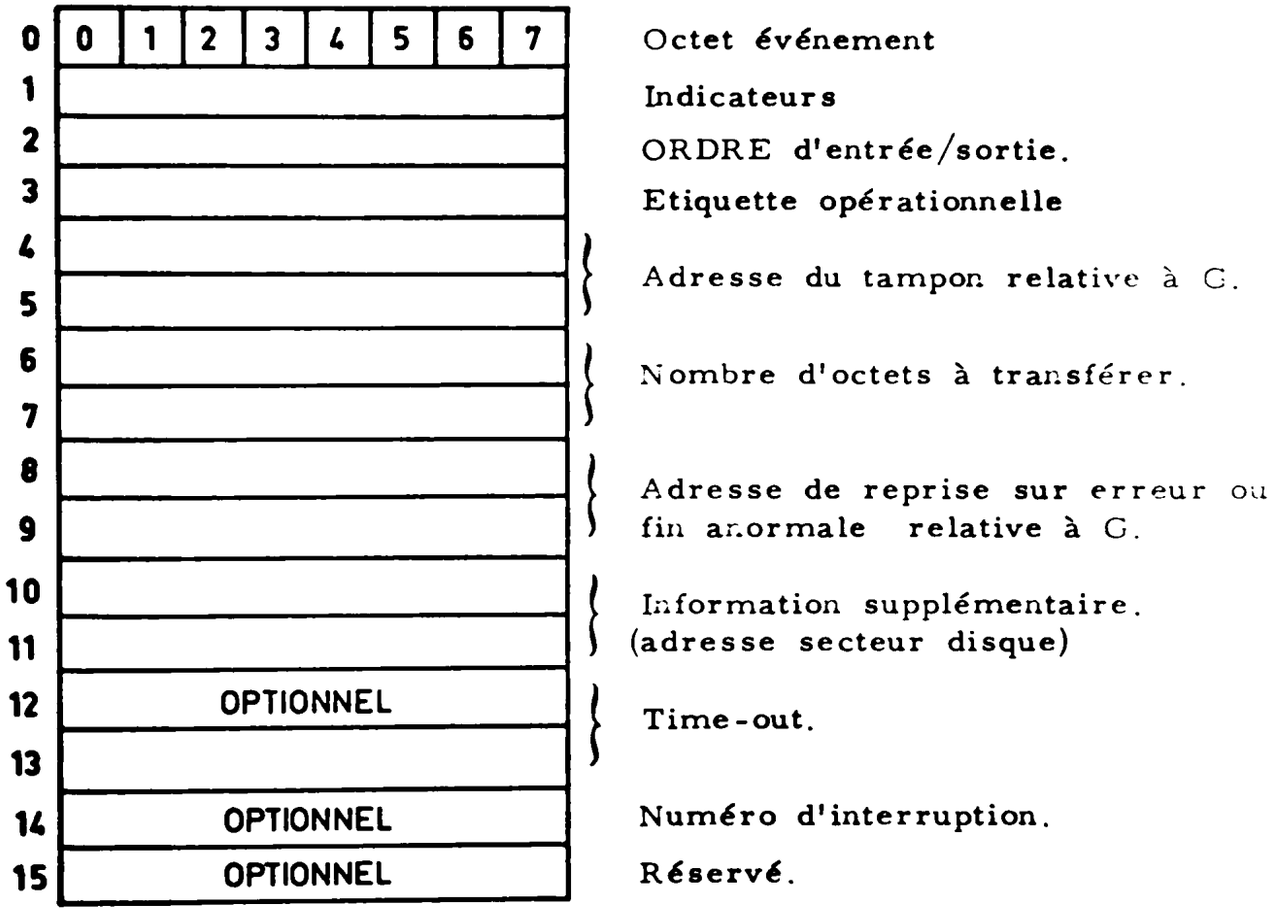
Les demandes de transferts.

a) Appels d'entrée/sortie CSVM : IO

Au Call superviseur (CSV) pour demande de transfert est associé un bloc de commande CB. Ce Bloc contient les paramètres définis par l'utilisateur. En fin de transfert, il comporte également des informations d'état rendues par le système. Au moment de l'appel le registre A contient l'adresse du CB relative à G. Le moniteur associe à la demande de transfert un évènement dynamique défini par

l'adresse du CB puis rend le contrôle au programme appelant après avoir initialisé ou mis le transfert en file d'attente.

b) Description des paramètres du bloc de commande (control bloc).



Signalons le rôle prépondérant de l'octet 0 : octet évènement. Le bit 0 de l'octet est mis à 1 à la prise en compte de l'appel et remis à 0 en fin de transfert.

Bit (0) { = 0 E/S terminée
= 1 E/S en cours

Bit (1) { = 1 erreur ou fin anormale
= 0 erreur logique (format d'appel incorect).

Bit (2) { = 1 erreur physique (signalée par exemple par coupleurs).
= 0 contrôle en fin de transfert.

Bit (3) { = 1 contrôle en initialisation du transfert.

Bit 4, 5, 6, 7, code d'erreur ou fin anormale.

Nous ne nous étendrons pas sur les multiples combinaisons qui sont offertes à l'utilisateur dans l'utilisation du bloc de commande et qui intéressent aussi l'utilisation de chaque handler spécifique.

c) Contrôle des transferts.

Il y a deux modes de contrôle de fin de transfert :

- Mode standard : l'état du coupleur est entièrement analysé par le moniteur qui renvoie à l'utilisateur un code d'anomalies normalisé, facile à interpréter. si l'érreur est irrésupérable, le contrôle n'est pas

rendu à l'utilisateur mais, après édition de l'identificateur du programme et du message d'erreur correspondant, la tâche est tuée.

- Mode utilisateur : le moniteur n'analyse pas l'état du coupleur(qui est à la charge de l'utilisateur) et rend le contrôle au programme utilisateur.

Le contrôle du transfert est effectué par un Call superviseur CSV M : WAIT le registre A devant contenir l'adresse à G du bloc de commande de l'entrée/sortie à contrôler. Sa fonction est de mettre en attente de fin d'E/S le programme appelant. Si le transfert est terminé au moment de l'appel de m: WAIT, celui-ci rend le contrôle à l'utilisateur. Si le transfert n'est pas terminé, le module 2 :

- range le niveau de la tâche appelante dans le CB d'entrée sortie.
- Désactive ce niveau s'il est non nul, effectue une boucle d'attente active s'il est nul.

M : WAIT désactive (par un DIT), provisoirement, le niveau d'un programme en attente d'entrée/sortie permettant ainsi la prise en compte sans blocage de programmes moins prioritaires. Ce programme sera repris au moment de la fin de transfert par réactivation de son niveau d' IT par le Handler du coupleur concerné par l'entrée/sortie. Un exemple d'organisation d'entrée/sortie est donné par le schéma suivant : Fig.4.9.

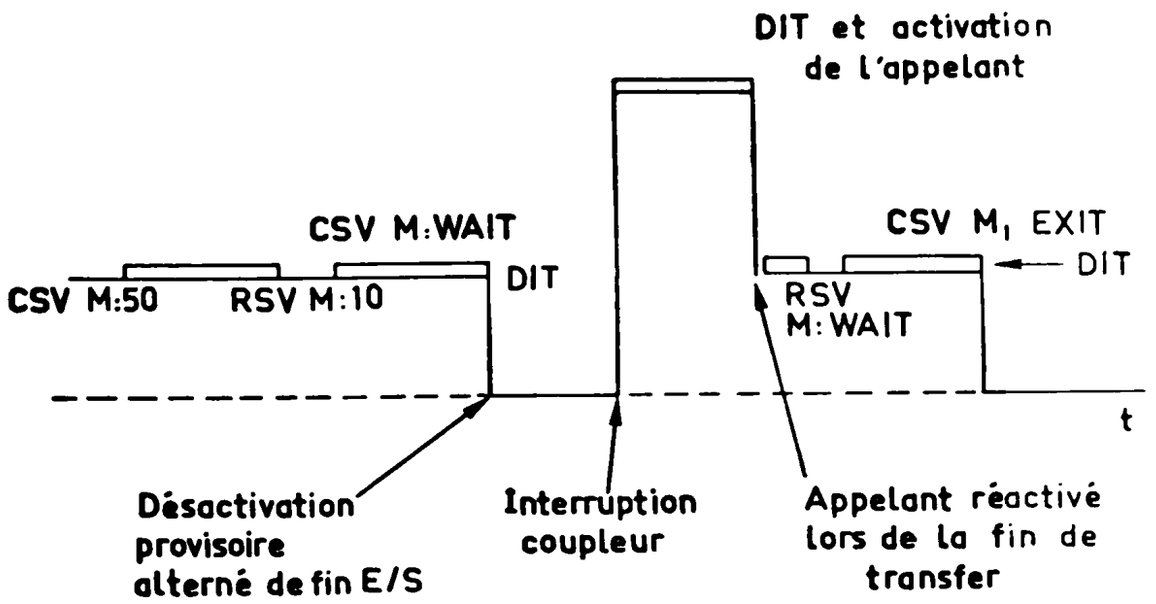


Fig. 4.9.

IV.4. Les étiquettes opérationnelles.

IV.4.1. Généralités.

Le système des étiquettes opérationnelles a été introduit pour permettre aux programmes de traiter un environnement logique. Une entrée/sortie sera demandée sur une étiquette opérationnelle qui représente une fonction d'Entrée-sortie. La correspondance enre étiquette opérationnelle et périphérique sera gérée par le moniteur par le biais des assignations.

Ce système permet de ne pas faire dépendre la programmation des entrées/sorties des périphériques, et donc, de rendre chaque programme entièrement compatible avec toute configuration.

IV.4.2. Définition.

Une étiquette opérationnelle est un numéro. Ce numéro est en général un mnémonique réparti en deux classes :

- Les étiquettes opérationnelles standard de la forme M:XX dont la fonction est définie.
- Les étiquettes opérationnelles utilisateur de la forme U : FX, dont les fonctions peuvent être définies complètement par l'utilisateur.

L'assignation des étiquettes opérationnelles standard est fixée à la génération mais peut être modifiée par commande moniteur (exepté pour les assignations téléscriptrice M : OC et disque système M : SY).

Pour les étiquettes opérationnelles utilisateur, c'est l'utilisateur qui précise les assignations qu'il désire.

- 5 -

PRINCIPES DU SYSTEME EN TEMPS PARTAGE MITRA 15

I - NECESSITE D'UNE GESTION CENTRALISEE DU NIVEAU DE FOND.

I.1. Généralités.

Le moniteur standard MTRD-E est orienté vers le traitement en temps réel de tâches et dans cette optique ne propose pas les outils necessaires à la multiprogrammation en dehors des possibilités offertes par le jeu du matériel (interruption prioritaire). La récupération des temps-morts occasionnés par les entrées-sorties n'est conçue qu'entre deux tâches travaillant à des niveaux de priorités différents en provoquant la désactivation de la tâche la plus prioritaire, par appel au module M : WAIT.

D'autre part, deux tâches ne peuvent se partager d'autres modules que les modules moniteurs.

L'introduction d'un processeur disposé sur un niveau d'interruption assure la commutation des utilisateurs APL sur le niveau de priorité le plus bas, le niveau "0". Celui-ci est, rappelons le, constamment actif lorsque tous les autres niveaux de priorité sont au repos. Ce processeur constitue le complément logique du moniteur Standard à condition d'établir les communications nécessaires entre les logiciels. L'obligation de gérer le niveau "0" peut s'illustrer à la faveur de l'étude de la sortie d'un programme en fonctionnement normal.

I.2. Sortie d'un programme sous système d'exploitation MTRD-E.

Un programme peut se dérouler au niveau "0" ou sur un niveau d'interruption.

a) Sur niveau d'interruption, la sortie d'un programme consiste à provoquer un DIT sur un contexte de désactivation disponible dans le système.

b) Au niveau "0" plusieurs cas se présentent :
   - Le "Batch" est en cours d'activité, il faut le reprendre.
   - Le "Batch" n'est pas actif, il faut exécuter le BRU $\not{S}$ du contexte d'attente (attente active).
   - C'est la fin d'un vidage au niveau "0", il faut reprendre le programme de rang "0".

La section M : EXIT termine normalement tout programme qui effectue ces opérations très spécialisées, de même en ce qui concerne M : ABRT (abandon d'un programme) qui se branche dans M : EXIT par un des trois points de reprise prévu. Ceci provoque des effets secondaires non négligeables :

a) L'abandon de toutes les entrées-sorties en cours sur le niveau du programme.
b) La fermeture de toutes les étiquettes opérationnelles détenues par la tâche.

De ces diverses remarques découlent les idées suivantes :

- Un enchaînement de travaux à un même niveau nécessite la prise en compte de la gestion des entrées-sorties et de la sortie d'un programme.
- La fiabilité du système en temps partagé nécessite le maintien de l'indépendance des usagers. La gestion de l'abandon doit être centralisée pour éviter qu'un usager n'entraîne l'arrêt des autres tâches par son éventuel déroutement.

II - GESTION DES ENTREES-SORTIES DANS LE SYSTEME EN TEMPS PARTAGE

II.1. La notion d'attente est révisée.

Nous pouvons distinguer deux classes d'entrée-sortie.

- Les entrées-sorties sur le terminal conversationnel lorsqu'il s'agit des usagers APL.
- Les entrées-sorties sur l'équipement du calculateur concernant tous les types de tâche en cours.

La prise en compte à un niveau d'interruption des entrées-sorties interdit l'attente traditionnelle effectuée par le module moniteur M : WAIT car celui-ci est essentiellement bloquant vis à vis du programme puisque le test de fin de transfert achevé provoque seul la sortie de ce module.

Nous prendrons comme hypothèse de travail que EXEMCO, superviseur gérant le temps partagé, est capable d'assurer plusieurs travaux simultanément notamment l'entrée clavier de chaque utilisateur. Après avoir initialisé un transfert d'entrée-sortie, EXEMCO est capable d'utiliser le temps mort qui en résulte pour traiter d'autres séquences indépendantes de programme. Il y a lieu de respecter les spécifications afin de permettre que la fin de transfert provoque la réanimation de la tâche du niveau superviseur ((R8)/2 donnant ce niveau lors de la prise de controle de l'Unité Centrale par le programme, est disposé dans les bits 1 à 7 de l'octet évènement du bloc de commande).

II.2. Les entrées-sorties sur les terminaux conversationnels.

Les récupération des temps morts spécifiques à cette classe d'entrée-sortie s'impose de façon naturelle à condition d'offrir à l'usager la possibilité

de frapper ses instructions APL alors que sa propre tâche ne contrôle plus l'Unité Centrale. Dans l'hypothèse précedemment définie, le processeur d'entrée/sortie disposé à un niveau d'interruption permet de prendre immédiatement en compte tout caractère ou ordre émanant des terminaux sans perturber le déroulement du programme APL par le jeu de la multiprogrammation matérielle.

Dans sa version monoconsole, le processeur APL 15 procède lui-même à ses entrées-sorties au sein de programmes spécialisés de façon classique (lancement de l'Entrée/Sortie, puis attente de la fin de transfert par CSV M : WAIT). Les blocs de contrôle sont résidents en CDS du processeur APL 15 et les tampons disponibles dynamiquement au sein de l'espace de travail en mémoire centrale, pour recevoir en entrée les caractères acquis qui sont transcodés par le processeur APL en code interne APL 15 et en sortie le message en code ASCII.

De cette conception résulte de nombreuses facilités offertes à l'opérateur pour l'écriture de ses instructions (contrôle des caractères, contrôle de la validité d'association de caractères, corrections, gestion de la ligne, gestion de l'écran ou de la page du terminal, modes de transaction graphique possible, etc...). De plus le nombre de caractères permis dans la commande en entrée est pratiquement illimité.

La solution proposée, afin d'adapter le programme APL 15 à son environnement en temps-partagé, exige d'apporter certaines modifications à ce processeur et la réduction du nombre de caractère admissible dans une commande. La gestion simultanée des terminaux est réalisée en récupérant les caractères dans des tampons résidents en mémoire, de taille limitée (mais ajustable à la génération du multi-console), au sein du processeur multi-console d'entrée-sortie. Les blocs de contrôle y sont

aussi implantés afin de piloter les terminaux. Nous verrons plus loin comment ce processeur peut travailler de façon parfaitement réentrante, afin de gérer les transactions en cours sur les terminaux connectés, avec les facilités précédemment esquissées. Dans son environnement en temps-partagé la fonction d'entrée/sortie du processeur APL 15 se réduira à définir les modalités de la transaction, à déposer une demande d'entrée-sortie au processeur multi-console et d'attendre la fin de la transaction.

Ce procédé permet ainsi de préparer les tâches à perdre le contrôle de l'unité centrale au profit d'une tâche prête en attente de cette ressource. La décision de changement de tâche est prise par le planificateur de travaux sur une entrée d'instruction, la sortie sur terminal type visualisation ne constituant pas un critère suffisant, étant donné sa rapidité.

II.3. Les entrées/sorties sur l'équipement externe du calculateur.

Le processeur APL 15 ainsi que la tâche supplémentaire offrent la possibilité d'adresser des périphériques autres que le terminal conversationnel pour la tâche APL. Les périphériques adressables peuvent être de vitesses très diverses et constituer, éventuellement un critère de commutation de tâche lorsqu'il s'agit de périphériques relativement lents, tel une imprimante, un lecteur de cartes, un lecteur de ruban, etc... Le vidage de la tâche est possible si la prise en compte de la transaction s'effectue dans des tampons non soumis au va-et-vient. Les entrées-sorties sont lancées par le jeu des étiquettes opérationnelles après l'assignation de celles-ci aux périphériques. Ceci permet de règlementer l'accès aux périphériques car, ils sont considérés comme non partageables par les tâches présentes dans le système. L'occupation de la ressource physique ne provoque aucune mise en attente de la tâche qui tenterait d'y accéder entre temps.

Le système gère donc des tables décrivant l'environnement physique dont bénéficient les usagers. L'ensemble des mesures prises prépare la tâche à pouvoir être reléguée sur le support secondaire.

Les échanges peuvent concerner des périphériques plus rapides que ceux précédemment évoqués tels les disques, ou les bandes qui font appel à une gestion de fichiers. Les périphériques sont partageables entre les diverses tâches. Les transactions menées sur ces supports sont relativement courtes et ne peuvent être prises comme critère de commutation de tâche notamment lorsqu'il s'agit du disque. Il faut donc introduire la notion consistant à préparer la tâche à ne pas perdre le contrôle de l'unité centrale pendant le déroulement de l'entrée-sortie. La tâche étant gelée en mémoire centrale, la transaction peut s'effectuer à partir du tampon défini par cette tâche.

Les requêtes d'entrée-sortie sont présentées par l'intermédiaire des modules spécialisés du moniteur standard. La prise du contrôle de la transaction par le système multi-console est assurée par des modifications apportées dans ces modules. Les modalités de la transaction sont donc imposées explicitement par les spécifications d'exploitation standardisées de ces modules.

III - LA GESTION DES PROGRAMMES DU NIVEAU "0".

III.1. Généralités.

Nous avons évoqué la nécessité d'une gestion décentralisée du programme présent au niveau "0".

Celui-ci est en mémoire sous forme d'un IMT (code exécutable), dont nous avons pu voir la structure au chapitre précédent. S'il s'agit du code processeur APL, le partage doit se faire entre tous les utilisateurs connectés. Cette réentrance ne peut être assurée que par ce même gestionnaire

à condition de sauvegarder et de rétablir les données suffisantes pour reprendre l'exécution d'une tâche interrompue. Ce problème ne se pose pas pour une tâche de type différent de l'APL car son IMT n'est pas partageable.

III.2. Les données du programme APL.

a) La PRT : elle comporte une partie évolutive décrivant le passage en mémoire des branches de recouvrement.

b) La CDS : comporte dans son intégralité toutes les informations nécessaires au déroulement d'une tâche APL pour le compte d'un usager.

c) Les SRD : L'ensemble des doubles-mot en tête de chaque segment de données locales de la racine, décrit l'évolution du programme APL par le mécanisme d'appel des sous-programmes.

d) La zone de recouvrement débute dès le début du segment de données locales du module "SHUNT", et comporte le contexte propre à l'IMT.

e) L'espace de travail : s'étend de la fin de la zone de recouvrement à la zone commune.

L'examen de la disposition en mémoire des données montre que les informations désignées aux points a et b, d et e, sont respectivement continues. Les informations du point c sont réparties en mémoire selon l'agencement propre aux segments de programme de la racine. L'agencement étant décrit dans la PRT permet la localisation des SRD et la recopie de ceux-ci dans la position située en amont de l'implantation du programme APL. Ce dernier artifice contribue à rendre continue les informations a, b, c.

En d'autres termes, l'échange d'utilisateur comportera deux ordres d'entrée-sortie sur le support de sauvegarde et une opération similaire à un déplacement. Le schéma 5.1. précise le volume et la disposition des informations concernées par la réentrance du programme APL. Le rôle de la zone privée sera établi plus loin.

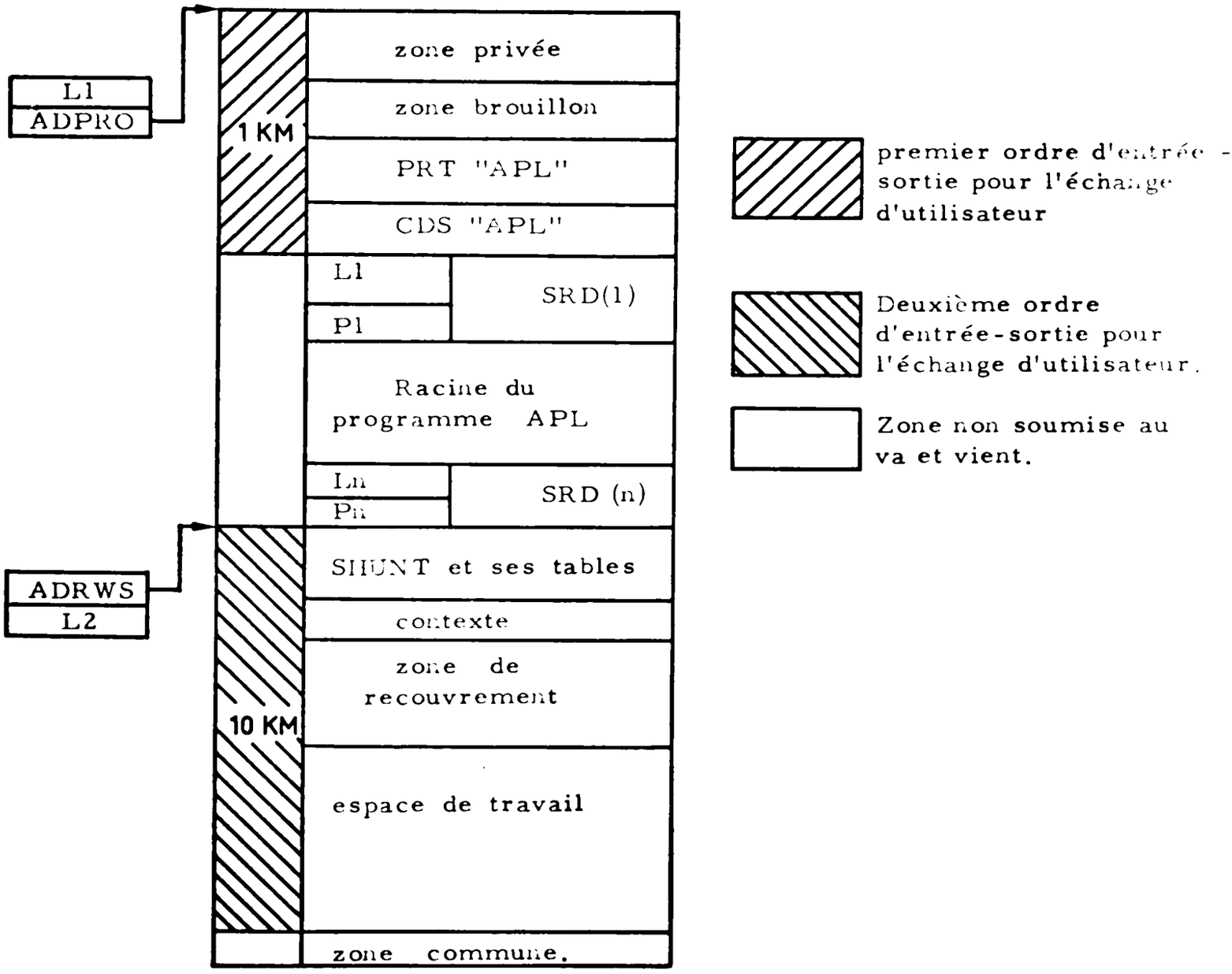


Fig. 5.1.

III.3. Gestion de la tâche de type non APL.

L'IMT de cette tâche ne pourra bénificier que d'une étendue de mémoire comprise entre le début de la zone de recouvrement et la zone commune, afin de ne pas détruire la racine de l'APL. Cette tâche ne requièrera qu'un seul ordre d'entrée-sortie sur son fichier de sauvegarde.

Toutes les informations nécessaires à la reprise de la tâche après interruption sont donc disponibles automatiquement.

III.4. Le rôle du contexte.

Le multiplexage du programme de type APL revient à imposer, outre les données, un contexte pour chaque usager. Son implantation est fixée par l'édition de liens de l'IMT APL et, est la même pour tous les usagers connectés, elle est, en revanche, dynamique pour une tâche de type différent.

III.4.1. L'usager APL.

A la position mémoire fixée nous trouvons :

a) Un contexte de démarrage imposé lors de la première apparition de la tâche dans le séquencement, établi à l'initialisation du système multiconsole.

b) Un contexte courant permettant la reprise du programme APL après une interruption par le niveau supérieur ou un sommeil sur disque.

A chaque passage de l'usager en mémoire il suffit de faire la connexion du contexte entrant ; c'est à dire remplir dans la table CPT la case pointeur correspondant au niveau de fond du pointeur en adresse absolue du contexte. Le niveau d'interruption se désactivant, le niveau de fond est activé automatiquement.

III.4.2. L'usager non APL

Lorsque le système multiconsole introduit un usager supplémentaire de type différent de l'APL, l'emplacement du contexte de cette tâche n'est connu que lorsque l'IMT correspondant a été chargé de la bibliothèque des programmes exécutables. La procédure de démarrage et de reprise est ici unique et s'accompagne de l'opération de connexion exposée plus haut.

IV - ORGANISATION DU PLANIFICATEUR DE TRAVAUX.

IV.1. Généralités.

IV.1.1. La notion d'événement.

Le processeur permettant la gestion des entrées-sorties des usagers, et le séquencement des tâches en mémoire, reçoit des excitations provenant des différents niveaux actifs.

- Des ordres du niveau de fond pour les entrées-sorties de type clavier et les entrées-sorties sur les périphériques gérés par le système.
- Des fins de transfert des différents coupleurs gérant les périphériques dont le résultat est de faire basculer à 0 le bit événement des premiers octets du bloc de contrôle concerné par l'entrée-sortie, résident en mémoire.
- De l'horloge destinée à quantifier le temps dont le résultat est de faire basculer à 0 le bit événement du premier octet des blocs de contrôle concernés par les délais résident en mémoire (notamment CBQT gérant le quantum de temps).

Il nous suffit d'aligner le déclenchement de la prise en compte des ordres du niveau "0" sur les spécifications des blocs de contrôle pour permettre une gestion homogène des événements significatifs du système. L'arrivée d'un événement peut donner, d'un point de vue logique, l'occasion de modifier la situation actuelle du système.

En d'autres termes, il est nécessaire d'organiser celui-ci de manière à être dans un état d'attente des dits événements.

La conception d'une file d'attente où le souvenir des événements en cours est conservé, correspond à ce critère. Une scrutation systématique de cette file permet de retrouver le ou les événements ayant causé l'excitation du superviseur.

IV.1.2. Retour sur la notion d'attente des événements : le rôle d'un contexte réduit pour la réentrance des entrées - sorties.

Le superviseur doit être capable de reprendre les séquences interrompues par l'attente. C'est à dire de rétablir le compteur ordinal et les conditions initiales en conséquence : de même, pour permettre les transactions sur les différents terminaux, il est nécessaire de disposer pour chaque clavier connecté, d'une zone où toutes les informations concernant l'écran et la ligne en traitement sont conservés, ainsi que les blocs de contrôle pilotant les entrées-sorties sur les périphériques et les tampons associés obligatoirement résidents. A chaque utilisateur, connecté ou non, est donc associée une zone de données, accessible simultanément par le superviseur en temps-partagé et par le processeur APL.

Pour pouvoir travailler de façon directe sur ces données il suffit de déplacer la base locale pour chaque utilisateur. Les instructions du programme superviseur sont du type à référence locale pour

pouvoir le rendre réentrant.

L'examen des conditions d'entrées-sorties montre que le contexte minimum de reprise d'une transaction peut se contenter de l'indication des deux bases suivantes : la base locale et le compteur ordinal. C'est ce que nous appellerons dorénavant le contexte réduit.

IV.1.3. La notion de boite aux lettres.

Nous pouvons préciser le mode de communication artificiellement implanté entre le niveau de fond contrôlant la mémoire et le système multiconsole. La notion d'événement, calquée sur celle adoptée par le système d'exploitation standard, s'applique à un mot dont les bits de plus haut poids sont significatif vis à vis du système multi-console. Il faut deux boites aux lettres ainsi définies :

- Une boite aux lettres réservée aux communications entre le système multi-console et le processeur APL, pour les entrées-sorties sur le terminal conversationnel de la tâche. Le nombre de boites aux lettres est ajusté au nombre de tâche de type APL.
- Une boite aux lettres réservée aux communications entre le système multiconsole et le système d'exploitation pour la gestion décentralisée du niveau "0". Elle est unique et son accès étant règlementé, il n'y a aucune ambiguïté sur l'appelant.

IV.1.4. Conclusions sur la notion d'événement et la structure du planificateur de travaux.

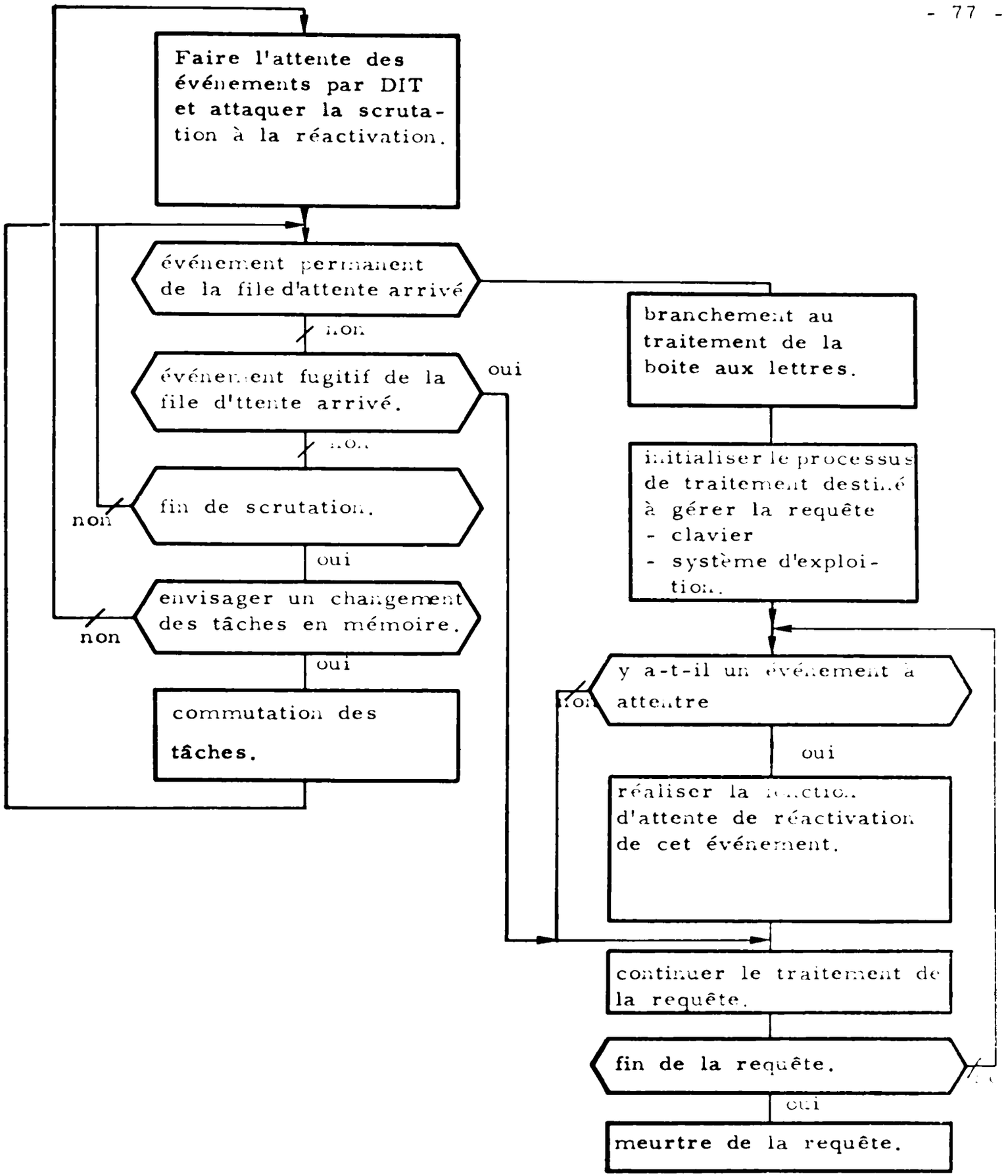


Fig. 5.2. La gestion des événements dans EXEMCO.

Le processeur en temps partagé EXEMCO est toujours en position d'attente de l'activation d'événements chaînés entre eux dans la file d'attente. Des événements y sont chaînés en permanence, associés à des contextes réduits, fixés une fois pour toutes qui permettent d'exécuter des traitements types.

Les boites aux lettres appartiennent à cette catégorie et constituent les processus initiateurs des requêtes des usagers : les requêtes peuvent avoir pour but de relancer des événements "fugitifs" jusqu'à la fin du traitement prévu. La tâche qui en découle évolue par la mise en attente de la réactivation de l'événement et se termine sans réinitialisation (puisqu'elle est permanente). Le schéma 5.2. montre l'articulation du système autour de la fonction d'attente réalisée par le DIT.

IV.2. Les modifications des modules d'exploitation.

L'action sur la boite aux lettres ci-dessus évoquée, nécessite l'adjonction d'une "verrue" dans plusieurs modules du système d'exploitation. Le travail demande la connaissance du mode d'appel des modules moniteurs, l'emploi de TWB d'après les principes exposés au chapitre précédent.

IV.2.1. Principes de la modification.

Un module moniteur est modifié dans le but de transmettre un certain nombre d'informations au superviseur multiconsole au moment du passage dans ce module du programme de fond, ceci de manière transparente à l'usager du module. La mise en place est facilitée par centralisation, au sein d'un module moniteur intégré au système d'exploitation lors de la constitution du noyau moniteur des actions suivantes :

- Transmission des données.
- Excitation du niveau superviseur.

L'appel à ce module spécialisé est réalisé par CSV dans chaque module modifié. Ces modules étant eux même appelés par CSV d'un usager, il faut dans cette éventualité réaliser la protection des mots de la TWB assurant le retour dans le programme utilisateur appelant. Cette protection doit d'effectuer dans cette même TWB à des emplacements qui ne sont pas concernés par l'utilisation du module modifié ou par les appelants internes (appel par SPA). Ces conditions satisfaites, nous pouvons présenter les organigrammes des modifications. Le rôle de la barrière est de permettre l'utilisation du système d'exploitation hors gestion en temps-partagé. C'est une instruction de saut du code de la modification pour assurer la standardisation, une instruction de branchement dans ce code pour déclencher le processus du temps-partagé. Chaque modification est donc adaptée au module dans lequel elle est implantée.

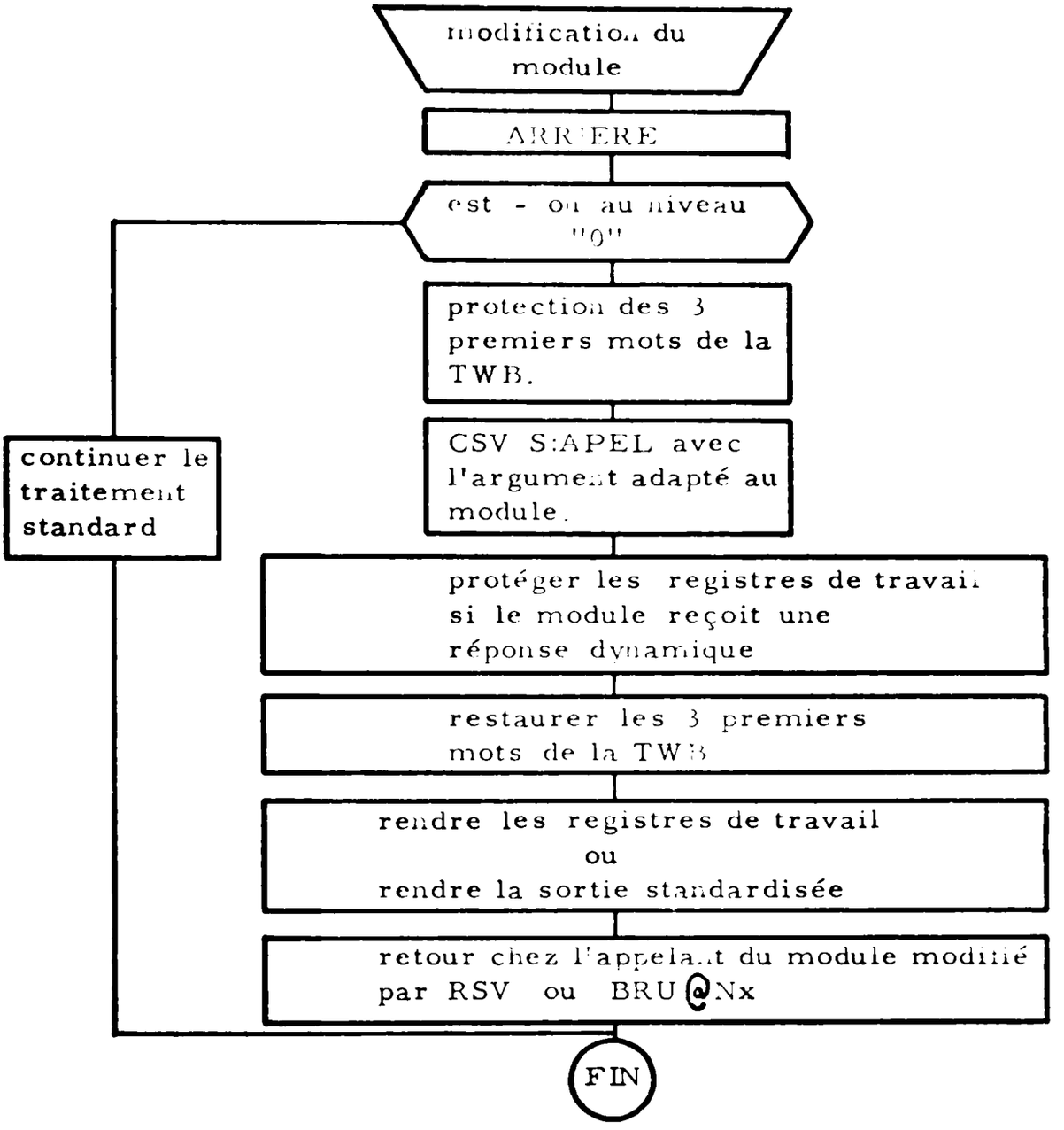


Fig. 5.3. Principe de la modification d'un module moniteur.

Le module S : APEL n'est pas réentrant. Les informations du niveau "0" adaptées à chaque module sont stockées dans sa LDS pour être restituées sur demande du superviseur multiconsole, selon le schéma suivant.

Le passage dans ce module se fait sous masque pour ne pas voir la tâche enlevée de la mémoire durant l'opération, le masque étant enlevé après l'excitation par une instruction WD21 par utilisation du CIT du niveau superviseur.

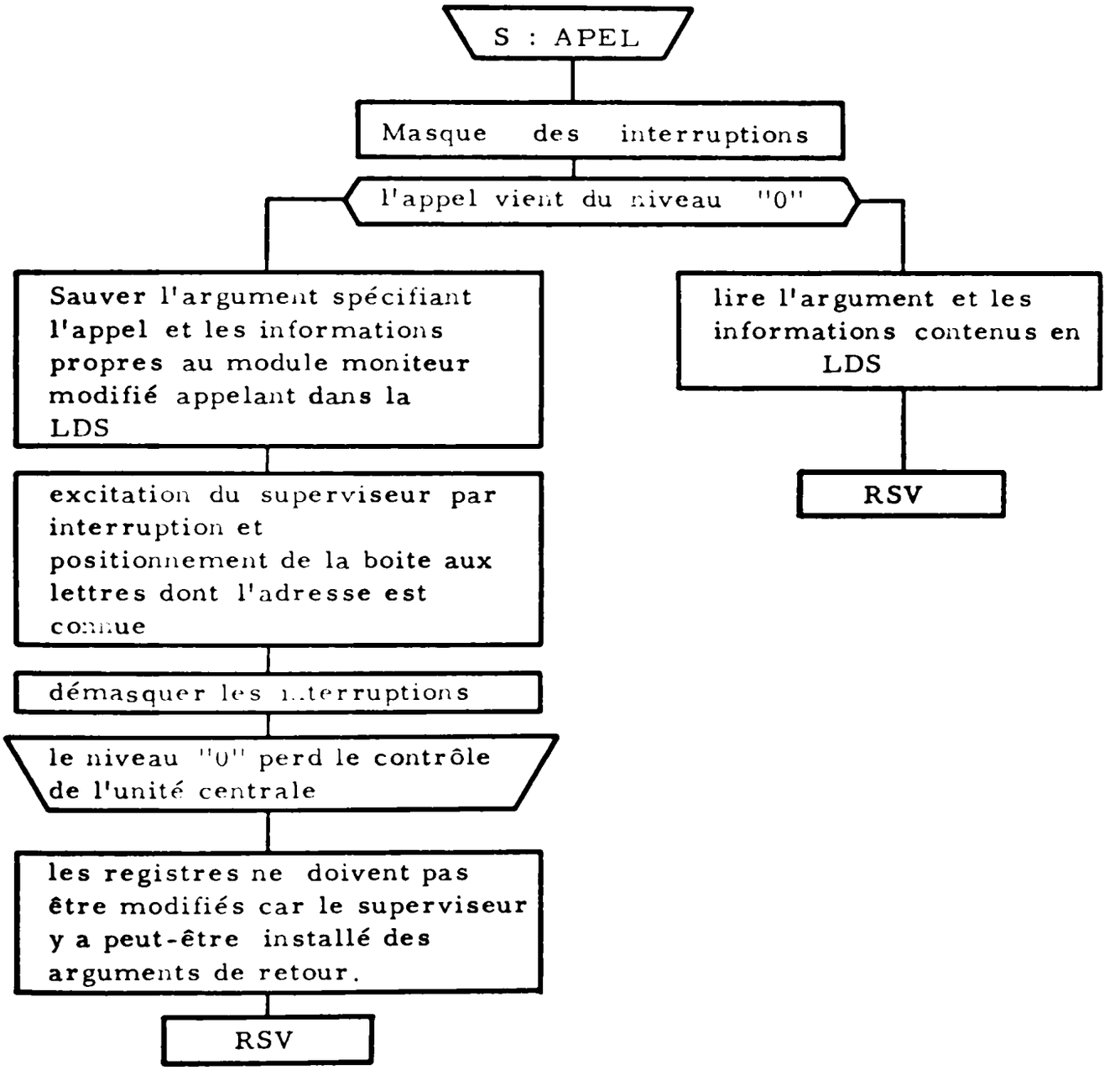


Fig. 5.4 Schéma d'organisation du module S :APL non réentrant servant à la liaison entre temps-partagé et système d'exploitation.

Le système est valable car il n'y a aucune ambiguïté sur le possesseur de la mémoire centrale. Nous pouvons voir que la restitution d'arguments est possible et dépend des spécifications du module utilisé. Certains modules se contentant de restituer l'argument d'appel, d'autres présentant des registres chargés d'une réponse à la transaction demandée. Ceci est obtenu par écriture directe dans le contexte de la tâche en mémoire lorsque le superviseur multiconsole a terminé la prise en charge de la transaction.

IV.2.2. Les modules moniteur modifiés.

Chaque module est repéré par un code transmis en même temps que les informations utiles à l'exploitation (en général une adresse de table relative à la base générale du niveau "0"). Les modules modifiés sont les modules touchant aux entrées-sorties, aux programme de fin de travail et d'abandon, en conformité avec les notions introduites. Les modules de gestion de fichiers sont concernés par ces modifications car les notions de "background" et "foreground" interviennent au niveau des protections.

En d'autres termes l'introduction du temps partagé a nécessité l'abandon de certaines modalités liées à l'utilisation en temps-réel du système d'exploitation. Les modules modifiés sont :

M : IO	module d'entrée-sortie.
M : WAIT	module d'attente.
M : ASGN	assignation d'une étiquette opérationnelle à un fichier ou un périphérique.
M : OPEN	ouverture d'une étiquette opérationnelle.
M : CLOS	fermeture d'un fichier.
M : EXIT	fin de programme.
M : ABRT	abandon du programme.
M : KILL	meurtre d'une tâche.

IV.2.3. Les autres modules spécialisés.

Outre le module moniteur S : APEL, le module spécialisé S : MCO est destiné à manipuler les instructions de mise en jeu des modifications selon l'argument d'appel. Intégré à la constitution du noyau moniteur il fonctionne selon l'organigramme suivant : Fig. 5.4.

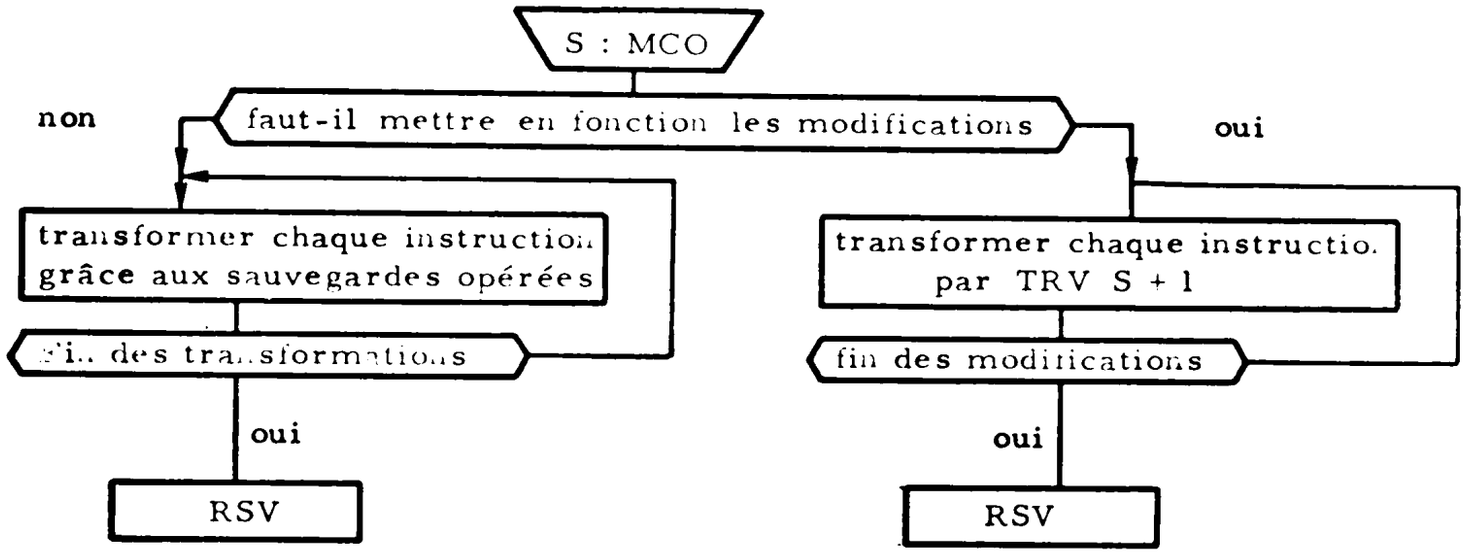


Fig. 5.4. Organigramme simplifié de S : MCO.

Le dernier module non standard S : ACTI est utilisé par le processeur APL et réalise l'excitation du niveau d'interruption superviseur du multiconsole par utilisation directe du CIT du niveau superviseur avec l'instruction WD 21.

Ceci permet une certaine souplesse d'installation du système sur des sites de configurations très particulières car le niveau d'interruption constitue un paramètre facilement modifiable.

IV.3. Les principes de la commutation des tâches.

IV.3.1. La nécessité de l'arrêt du niveau "0".

Lorsque le planificateur de travaux décide de changer le processeur de la mémoire centrale, il place le niveau "0" sur le contexte d'attente.

Ceci permet de réaliser le va et vient de façon à ce que les informations transférées n'évoluent pas pendant le temps de l'opération ; les ordres d'entrées-sorties passant par le mécanisme de file d'attente pour ne pas bloquer l'entrée des caractères durant les temps morts.

Le planificateur de travaux peut aussi faire fonctionner le système en temps-partagé par le jeu d'une table des pointeurs des différents contextes des tâches évoluant dans le séquencement.

IV.3.2. Les critères d'ordonnancement.

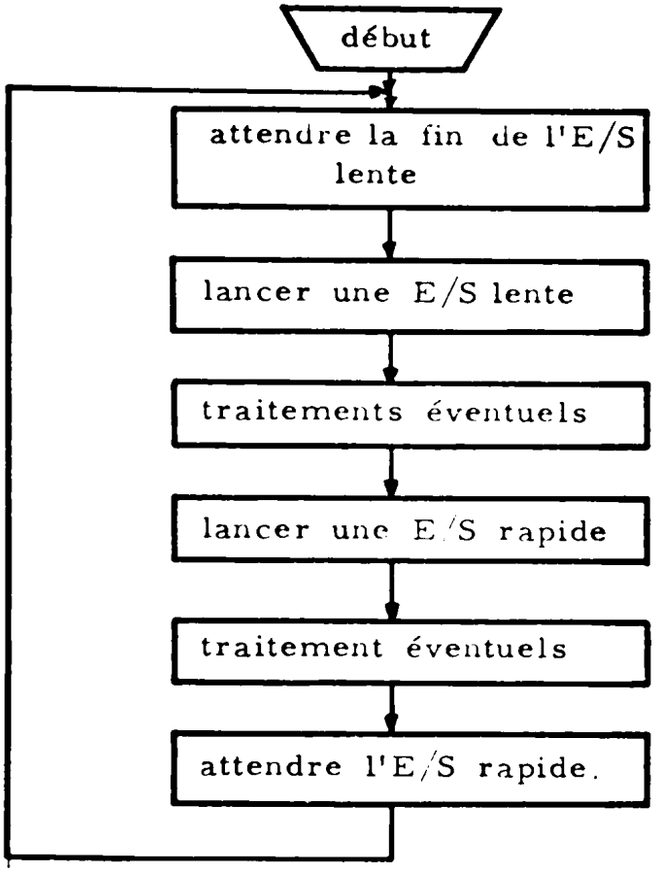
Nous désignons ainsi la politique adoptée pour introduire les changements de contrôle de l'unité Centrale. Nous avons répertorié jusqu'à présent les motifs essentiels suivants :

- Transaction de durée indéterminée (primitive bloquante).
- Entrée-sortie sur terminal conversationnel.
- Entrée-sortie sur terminal de type lent.
- Interruption de l'horloge (quantum de temps).

Il est possible au niveau des entrée-sorties de dégager une notion générale de demande de commutation.

En effet, il peut être souhaitable d'écrire un programme optimisé lorsque plusieurs périphériques de vitesses différentes sont

concernés. Le rythme peut être aligné sur la récupération des temps morts du périphérique le plus lent de la façon suivante :



En d'autre termes la simultanéité des traitements d'entrée-sortie et de l'unité centrale doit être respectée. Le critère de commutation sera donc la demande explicite d'attente de fin de transaction lente soit par module M : WAIT, soit implicitement à l'entrée clavier où l'usager fait une boucle active sur l'indicateur de transaction.

IV.4. Organisation de la sauvegarde.

La section INIMCO est destinée à initialiser le système de sauvegarde et, de manière générale, le système en temps-partagé. Cette partie de code est donc inutilisée par la suite mais il est récupérable pour constituer les

zones privées et brouillon, le contexte du système multiconsole étant déplacé pour la circonstance. A l'initialisation du système, l'appel à INIMCO provoque le chargement de l'IMT processeur APL 15 dont le nom est indiqué au processeur de lancement. Ceci permet de définir les constantes utiles du système dans un premier temps.

Nom	Accès	Fonction
GPR APL	Local	Valeur absolue de la base C du processeur APL
CTX APL	Local	Valeur absolue du pointeur de contexte
ZINTSP	général	Déplacements en CDS de l'APL. Pointeur de la zone utile de l'usager.
ZPERIP		Indique le type du périphérique connecté.
ZTESDA		Compteur d'appui de l'abandon
ZCBDAB		Indicateur de l'événement abandon
ZCBDSQ		Fournit une paire d'étiquettes opérationnelles de travail.
ZCACT		Numéro de consoles actives.
ZINTRA		Pointeur pour utiliser la table de transcodage. dans une LDS résidente de l'IMT APL.
ZREFTE		Pour la référence de temps.
A : FORE	général	Implantation du catalogue de sémaphores réservés à la tâche.

Pour les caractéristiques des fichiers de sauvegarde destinées aux tâches APL, les caractéristiques de la tâche supplémentaire étant définies dynamiquement.

ADPRO	Pointeur sélectif à C de la première zone à sauvegarder
LNGCDS	Longueur de la zone privée + zone brouillon + PRT + CDS
SWDPRT	Secteur d'implantation sur le fichier de sauvegarde.
LNGSWP	Longueur de SHUNT + "OVERLAY" + Espace de travail.
SWPWS	Secteur d'implantation sur le fichier de sauvegarde.

Les blocs de contrôle des entrées-sorties sur fichiers de sauvegarde sont initialisés grâce à ces données.

Le fichier de sauvegarde est défini a priori et la table des noms est décrite en T :TEXT. Le système en temps-partagé définit autant d'étiquettes opérationnelles que d'usagers, y compris la tâche supplémentaire, pour effectuer les opérations de va-et-vient sur les fichiers de sauvegarde par la table PUF.

Alors que le fichier réservé à la tâche supplémentaire ne subit aucun traitement, les fichiers dévolus aux tâches de type APL sont initialisés avec une CDS fraiche servant à chaque première entrée de la tâche en mémoire.

Ceci permet de contrôler l'environnement du système puisqu'il y a aussi une répartition des étiquettes opérationnelles qui peuvent être utilisées. En effet il faut obligatoirement réserver autant de couples d'étiquettes opérationnelles débutant par une étiquette paire, qu'il y a d'usagers APL, et déterminer un ensemble d'étiquettes libre où les tâches APL pourront prélever les étiquettes destinées aux fonctions d'entrée-sortie sur fichier externe.

ORGANISATION DES ENTRES-SORTIES

I - ORGANISATION DE LA FILE D'ATTENTE DES EVENEMENTS ET DE LA SCRUTATION.

I.1. La file d'attente des événements et des contextes réduits.

La file d'attente des événements est une liste à simple pointeur organisée en cellule élémentaire de deux mots, et décrite dans le sens de la plus ancienne demande à la plus récente demande. Lorsqu'un événement est arrivé, la cellule correspondante est ôtée de la file d'attente pour être mise dans une file regroupant les cellules vides. Les éléments libres de la file sont chaînés entre eux ; l'empilement des cases libérées se fait par la fin de la file et deux pointeurs permettent de trouver le premier et le dernier élément. La file d'attente des événements a la particularité de comporter en tête de liste des cases permanentes, c'est à dire qui ne sont jamais sorties de la file lorsque l'événement que la case désignée est accompli.

La file comporte 3 + N cases réservées, N étant le nombre d'utilisateur APL permis par le système. Les trois événements sont dans l'ordre :

- Le pointeur à la boite aux lettres de demande d'entrée-sortie clavier de la tâche contrôlant actuellement l'unité-centrale.
- Le pointeur à la boite aux lettre de demande de transaction venant des modifications du moniteur d'exploitation (adresse fixe en CDS du superviseur).
- Le pointeur au bloc de contrôle destiné au découpage du temps.

Les N événements sont les pointeurs aux blocs de contrôle du caractère de reconnaissance de chaque ligne connectable au système. La console étant connectée, ce bloc de contrôle est en permanence à l'affût d'un caractère

d'abandon.

Parallèlement à cette file d'attente, une liste de cases de deux mots permet la sauvegarde du contexte réduit, servant à la reprise des codes de traitement de l'événement occupant la case de même index de la file d'attente.

La file d'attente des événements est décrite par la Fig. 6.1.

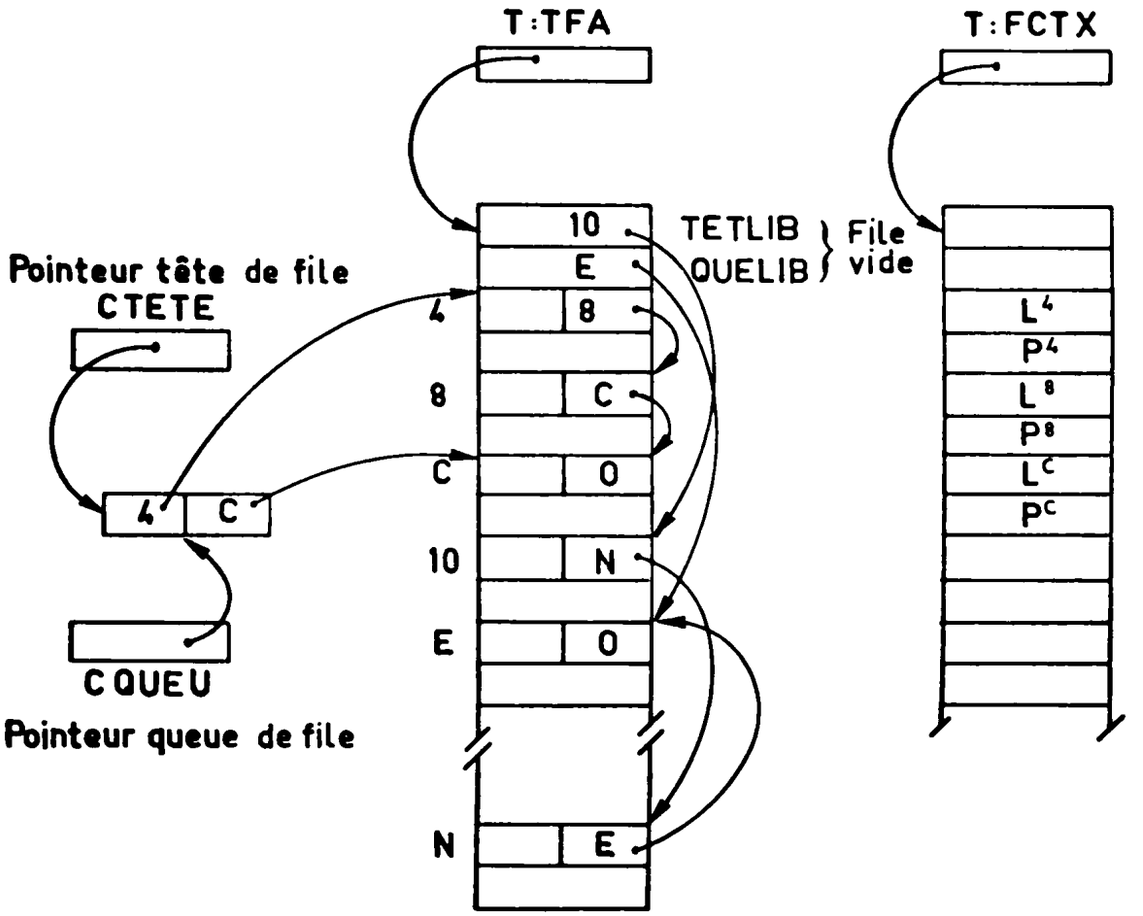
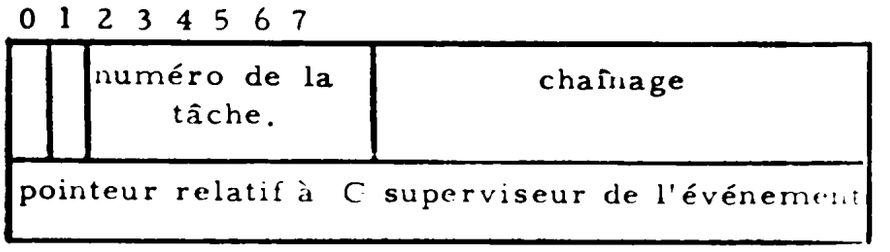


Fig. 6.1. Implantation de la file d'attente des événements en cours .

a) Description d'une case de la file d'attente.

La case élémentaire comporte les informations suivantes :

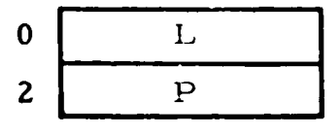
description de l'événement



Le chaînage est l'index dans la file d'attente de la case suivant la case actuelle, nul, si la case est la dernière de la file. Le codage sur un octet entraîne une limitation à 63 événements possibles. Un test de saturation de file est néammoins prévu pour répondre à toute éventualité. L'octet 0 est codé selon le principe suivant :

- Les bits 0 - 1 indiquent les caractéristiques de l'événement mis en attente.
- Bit 0 = 1 événement résident : il ne sera pas enlevé de la file après traitement.
- Bit 0 - 1 = 0 événement banalisé : c'est à dire entrée-sortie sur un périphérique de type terminal conversationnel ou partageable ou toute information de structure bloc de commande (délai)...
- Bit 1 = 1 événement lent : caractérise les entrées-sorties sur un périphérique non partageable, nécessite un traitement particulier comme il sera vu plus loin.
- Bit 2.7. codage du numéro de tâche ayant produit l'événement parmi 63 tâche. Le deuxième mot contient le pointeur relatif à G du superviseur multiconsole de l'événement en attente.

b) Description d'une case de la file des contextes réduits.



Le premier mot contient l'adresse absolue de la zone clavier de la tâche qui est équivalent à une base locale;par abus de langage nous désignerons par zone clavier la zone réduite dévolue à la tâche non APL du système.

Le deuxième mot contient le compteur ordinal de reprise de la séquence de traitement à l'arrivée de l'événement associé.

La file est composée d'une façon analogue à la file d'attente des événements ; ainsi les contextes réduits des événements permanents sont fixés selon le schéma suivant où nous prenons comme hypothèse de travail qu'il n'y a pas d'ambiguïté sur l'usager en mémoire de zone usager ZC1.

La Fig. 6.2. détaille l'implantation du dispositif en mémoire centrale.

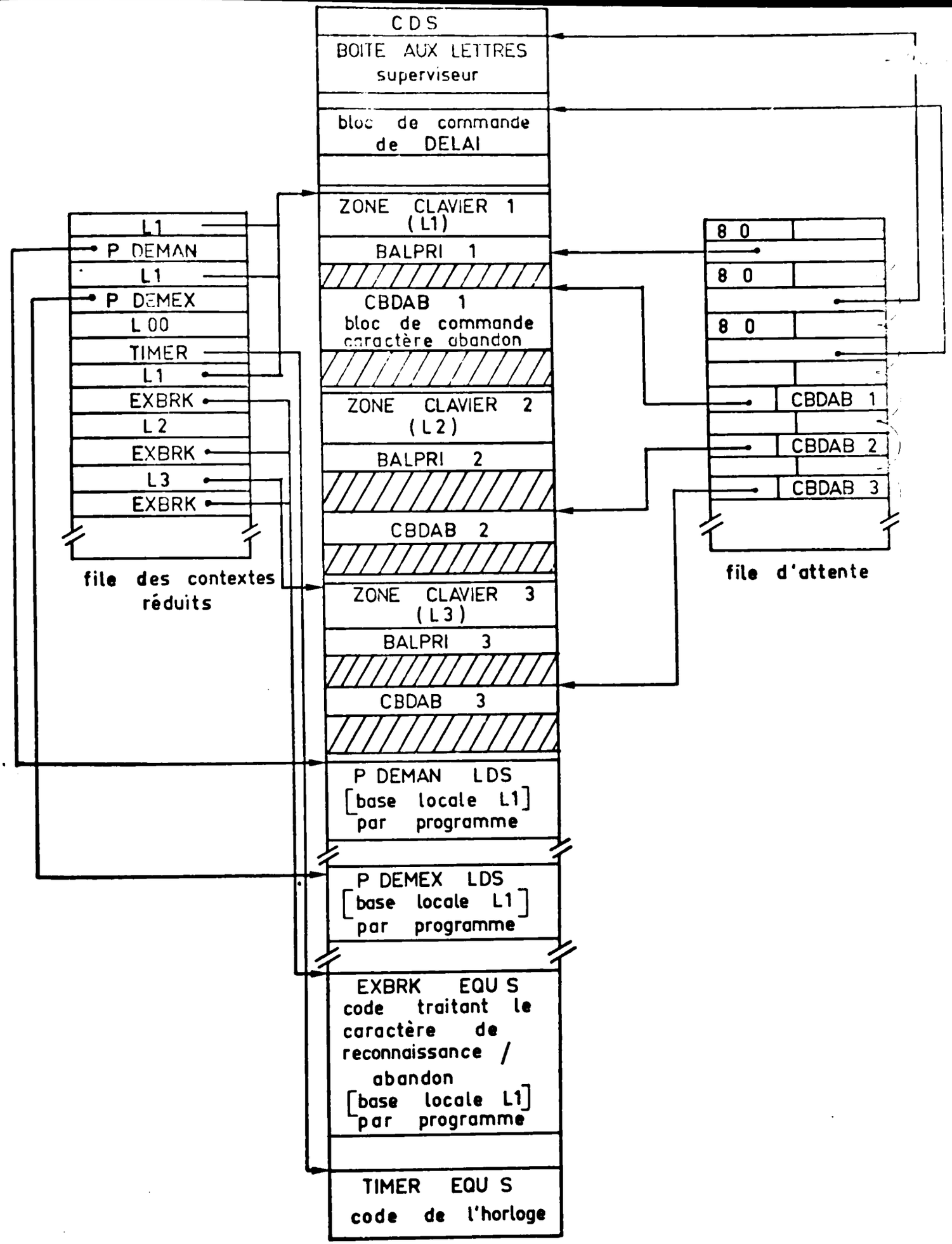


Fig. 6.2. La file d'attente et la file des contextes réduits pour les événements permanents. Détail de l'implantation.

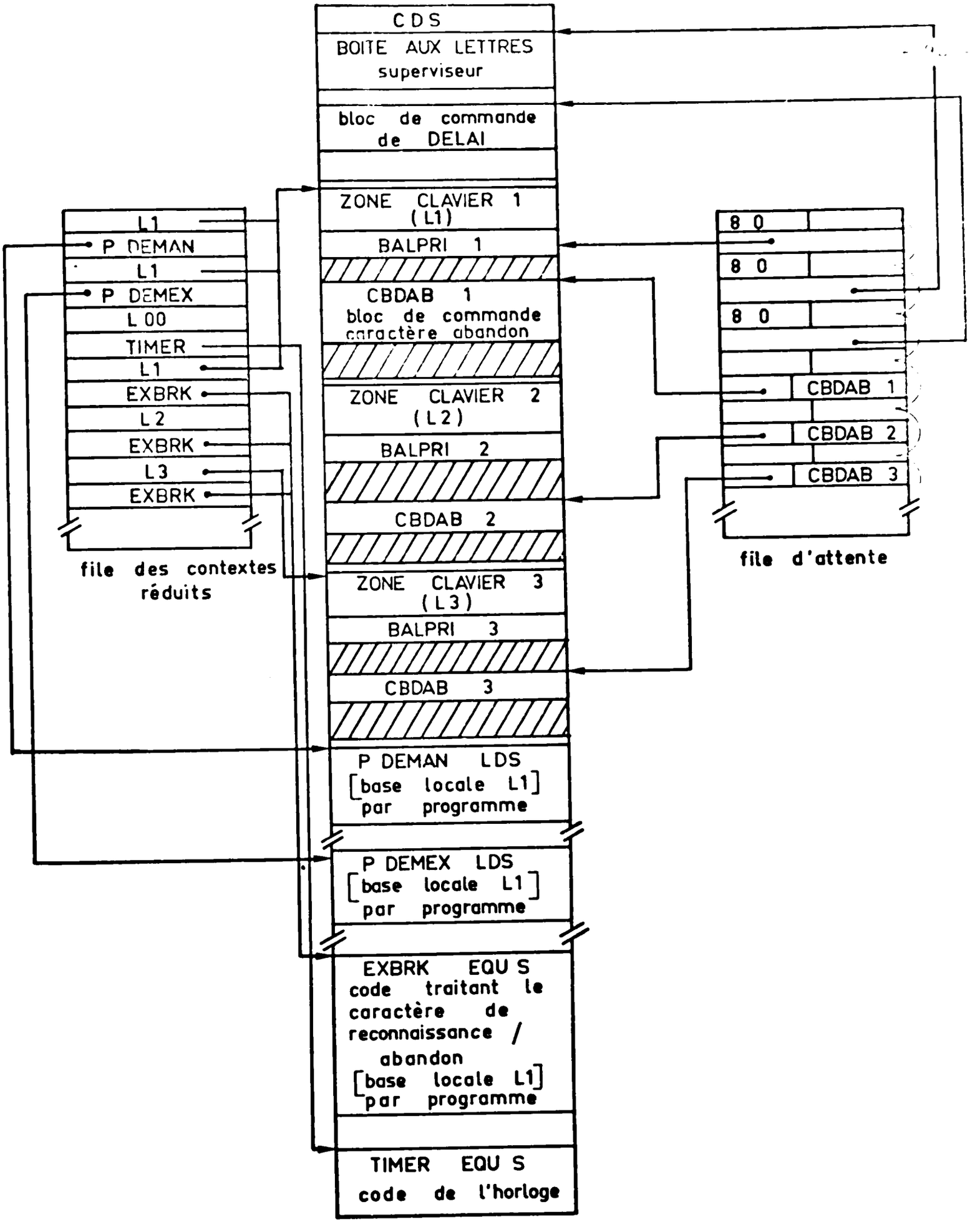


Fig. 6.2. La file d'attente et la file des contextes réduits pour les événements permanents. Détail de l'implantation.

I.2. Principe de la scrutation.

Le système en temps partagé, supposé au repos après l'instruction DIT, est à l'affut de toute excitation provenant de l'environnement logique ou physique.

Les tables étant disposées dans la LDS du superviseur, l'attaque du code de la boucle de scrutation se fait avec la base L positionnée correctement, la valeur de celle-ci par ailleurs est disponible en accès général direct dans le mot désigné par LORDON.

L'examen commence par le numéro de tête de la file et se termine par détection du chaînage nul, il consiste à tester le bit 0 des mots pointés par le deuxième mot de chaque case. Si un événement est détecté, il est :

- Enlevé de la file d'attente pour un événement banalisé ou pour un événement lent lorsque ceci est permis.
- Laissé en file d'attente pour les autres types.

Dans tous les cas, les informations suivantes recopiées dans la CDS en accès direct permettent de s'aiguiller correctement dans le traitement de l'évènement.

Origine de l'information	position	fonction de l'information
file d'attente	NUMAC	numéro "filtré" de la tâche ayant lancé l'événement
file des contextes réduits	L TACHE	valeur de la base locale de la tâche qui va être relancée.
	P TACHE	valeur du compteur ordinal de la tâche qui va être relancée.

L'aiguillage dans le traitement est réalisé par changement de la base L puis par un branchement inconditionnel au contenu de P TACHE.

Les informations intéressant la scrutation sont rassemblées dans ce tableau.

position de l'information	Origine de l'information	fonction de l'information.
PORDON	Programme	adresse de retour dans la scrutation après le traitement
LORDON	Programme	valeur de la base locale des données de la boucle de scrutation.
COURAN	File d'attente	index de la case en examen dans la file d'attente
CHAIN	File d'attente	index de la case suivante dans le chaînage
AVANT	File d'attente	index de la case précédente à la case courante.

Le schéma 6.3. montre le principe du déchaînage et l'introduction du numéro de la case dans la file libre. Remarquons que NUMAC contient le numéro de tâche pour le compte de laquelle le traitement va être effectué, une fois les caractéristiques de l'événement enlevées.

Si le traitement est consécutif à une demande faite par une boite aux lettres (il ne peut y avoir identification à ce niveau) NUMAC est rempli à partir de V : CUR contenant le numéro de la tâche contrôlant la mémoire centrale car seule cette tâche a le pouvoir de positionner les boites aux lettres.

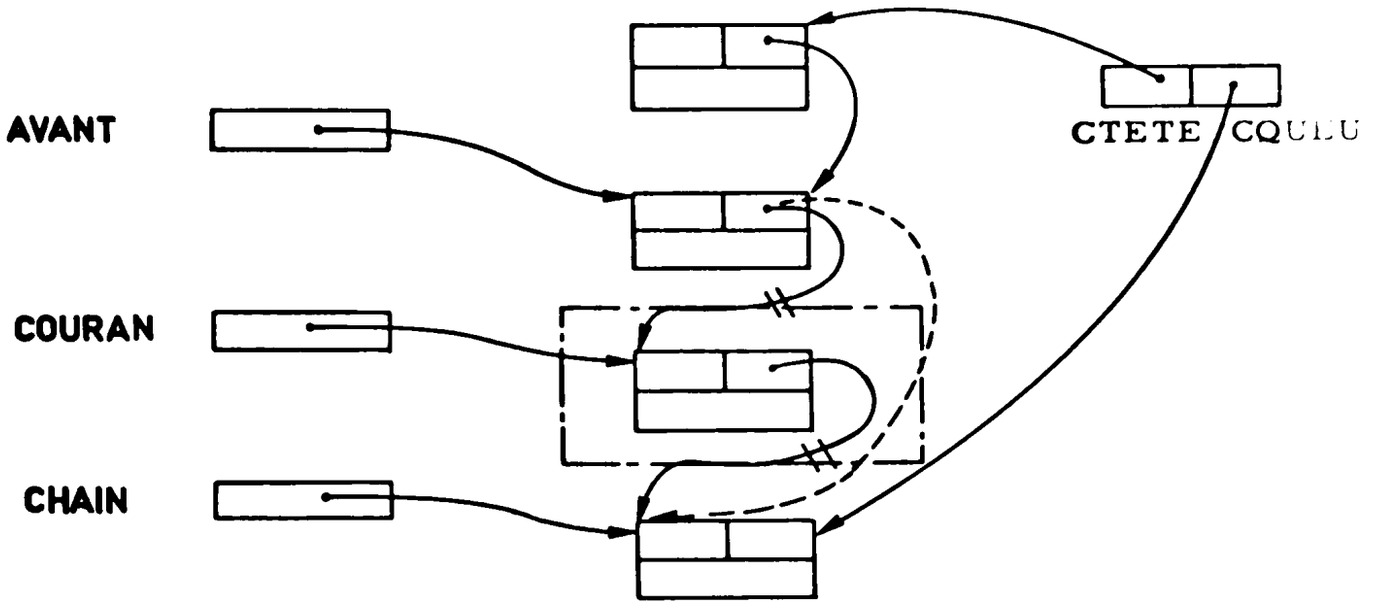


Fig. 6.3.a. Déchaînage de la case en file d'attente.

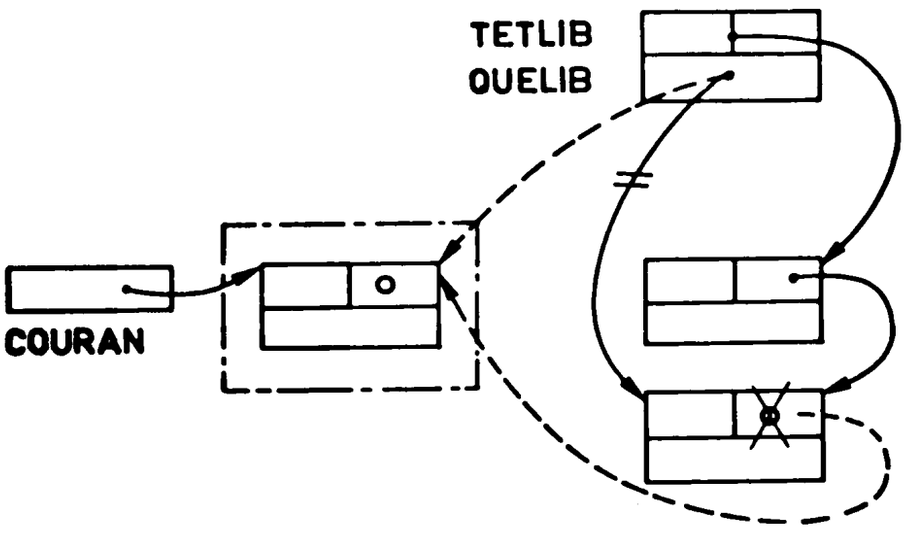


Fig. 6.3.b. Mise en file des cases vides.

I.3. La mise en file d'attente.

Au cours d'un traitement pour le compte d'une tâche au niveau superviseur, il peut y avoir lancement d'une E/S et mise en file d'attente simple ; l'événement qui en résulte est pointé par ADCB, relativement à la base C du superviseur.

La section Q : IOWA lance et contrôle une entrée sortie et Q :WAIT est chargé de faire la mise en attente d'un événement, l'appel à ces sections permet de récupérer les valeurs de la base L et la valeur du compteur ordinal au moment du CLS.

L'incrémentation de cette dernière valeur donne la valeur du compteur ordinal de reprise de traitement. L'opération s'effectue en deux temps abstraction faite d'une éventuelle saturation de la file d'attente.

- La récupération d'une case dans la file des cases libres.
- L'insertion dans la queue de file d'attente de cette case.

La case est constituée à partir de l'information donnée par NUMAC qui désigne le numéro de la tâche pour laquelle le superviseur est en train d'effectuer les travaux de mise en file d'attente.

Ces opérations sont schématisées en Fig. 6.4.

Ce numéro peut être différent du numéro de la tâche actuellement en mémoire centrale.

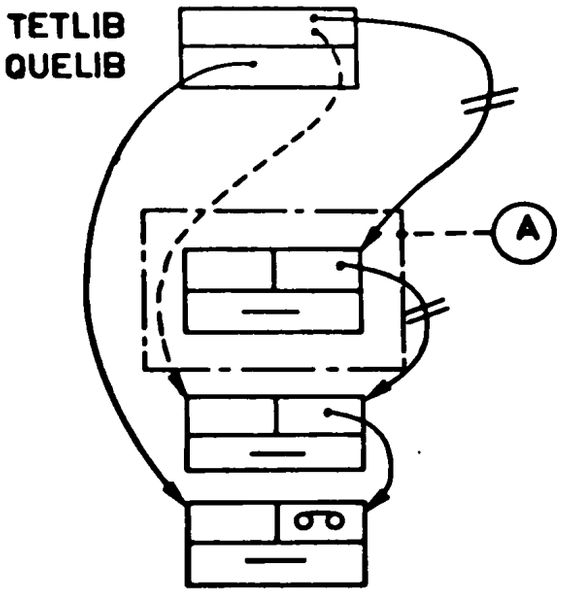


Fig. 6.4.a. Prélèvement d'une case libre en début de la file vide

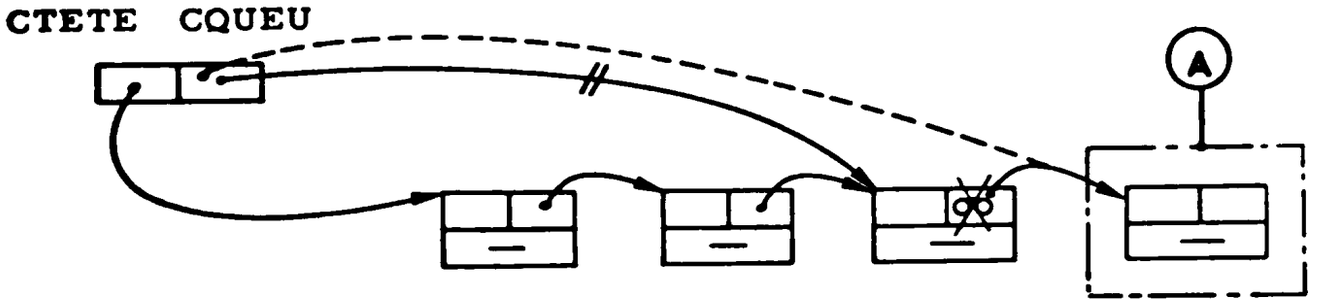


Fig. 6.4.b. Insertion par la queue dans la file d'attente des événements de la case A.

La mise en file d'attente se termine toujour par le retour dans la boucle de scrutation.

I.4. Le retour en scrutation après une mise en file d'attente.

Le superviseur ayant traité une transaction démarrée par une tâche doit retourner faire le traitement éventuel d'une transaction qui relève d'une autre tâche. En d'autres termes, il faut effectuer une sorte de multiprogrammation au niveau du superviseur, commandée par le mécanisme de scrutation. Le retour s'effectue par appel à la séquence P : END2.

- La base L est chargée par le contenu du mot LORDON.
- Le compteur ordinal est chargé par branchement inconditionnel au contenu du mot PORDON.

I.5. La fin de la scrutation.

La scrutation détecte l'occurrence de la fin de la file chaînant les évènements en attente (CHAIN est nul). Le superviseur est alors susceptible de la désactiver dans l'attente d'une nouvelle activation. Néammoins, le fait qu'il ait traité différentes transactions pour le compte des diverses tâches présentes dans le système peut avoir établi les conditions d'une commutation de tâche. Le test de l'indicateur d'appel du processus de changement de tâche (P : CALL), positionné lors d'un précédent traitement de transaction, permet le branchement dans le segment de programme SWAP. Celui-ci travaillant sur ses propres données locales, l'appel est effectué par CLS, mais le retour dans la boucle de scrutation pour exécuter la désactivation ne passe pas par l'instruction RTS mais par les chemins aménagés dans le superviseur. Une commutation de tâche pouvant avoir lieu, les éventuelles entrées-sorties sur le support secondaire sont traitées comme une requête normale, le numéro de tâche étant positionné à "0".

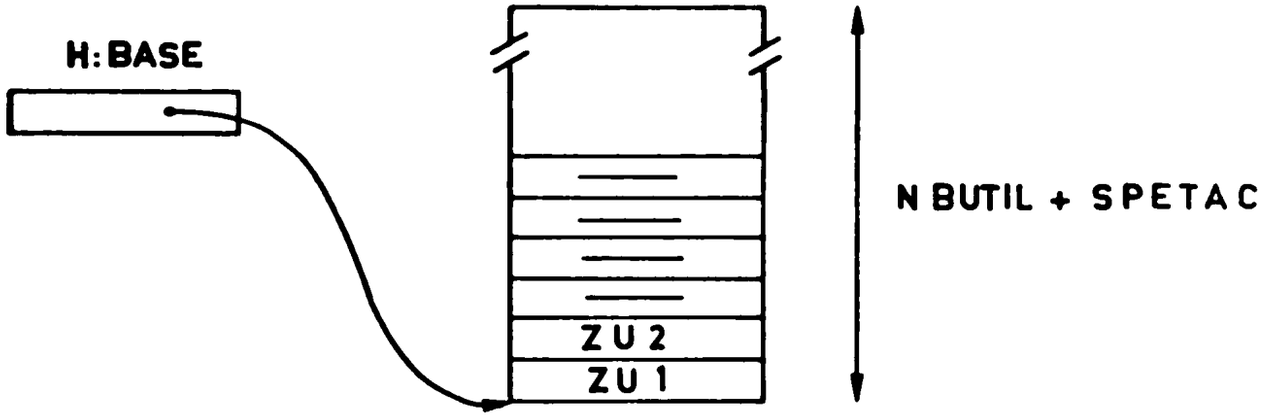
Avant de tomber en désactivation, le superviseur effectue sous masque d'interruption une boucle de test des événements de la file d'attente pour détecter les éventuelles fins de transactions qui auraient pu émaner des

"handlers" de plus haute priorité que le superviseur. L'instruction DIT termine cette séquence si le test n'est pas concluant. En revanche la détection d'une arrivée d'événement lance le processus de scrutation.

II - TRAITEMENT DES ENTREES-SORTIES SUR LES TERMINAUX CONVERSATIONNELS.

II.1. La table des zones de travail

Une table accessible à toutes les procédures du superviseur, permet de déterminer grâce au numéro de la tâche concernée, la position de sa zone utile selon le schéma suivant, en remarquant que la tâche numéro "0" n'existe pas.



Le nombre de pointeurs est égal au nombre de consoles connectables plus une tâche non APL permise.

II.2. Description d'une zone utile.

Une zone comprend quatre aires consécutives : la première servant de modèle commun à la tâche supplémentaire. Chaque zone usager de type APL est calquée sur ce modèle. Le concept de réentrance introduit sur la

LDS implique des contraintes similaires à celles rencontrées lors de la réentrance des modules moniteur appelés par CSV. Le mot BRCHPC a un rôle équivalent à celui de l'un des mots N0, N1, ...etc... de la TWB du programme. L'appel d'une section est réalisé dans le système en temps-partagé de la façon suivante :

```
SPA    BRCHPG
BRU    @#SORVIS
```

Etant entendu qu'au cours du traitement (lequel est la partie de code pointée en accès général par SORVIS) le retour après appel se fasse par

```
BRU    @BRCHPG
```

Les appels par CLS sont permis sur des modules travaillant en fait sur leur propre segment de données locales. Ce mode de communication impose d'entretenir en accès général un certain nombre de pointeurs faisant référence à des étiquettes dont chacune correspond de façon logique à une section.

position	M numérique	fonction
0	FSWAP	compteur des E/S destinées au vidage
2	FNSWAP	compteur des E/S destinées à ne pas être vidé
4	FILEAT	description de la file des événements lents arrivés
6	CEXIT	compteur d'E/S en cours après l'abandon de la tâche et souvenir d'arrêt derrière un sémaphore.
8	BALPRI	boite aux lettres réservées à la tâche
A	CB:DLY	bloc de contrôle du temps à l'usage de "tektronix".
C		
E		
10	CB:REC	bloc de contrôle de réception de caractère
12		
14		
16		
18	CB:DAB	bloc de contrôle de réception caractère reconnaissance/
1A		abandon/
1C		
1E		
20	CB:EMV	bloc de contrôle d'émission des messages
22		
24		
26		
28	AD BUF	pointeur en absolu du tampon d'entrée-sortie.
2A	T:DAB	compteur des appuis d'abandon
2C	BRCKPC	sauvegarde des adresses de retour en séquence traitement
2E	SUPPER	type du périphérique (0, 0).
30	:COMPT	contrôle de tampon (DIABLO/HYPRINT), numéro de page
32		
34		gestion de la ligne et de l'écran éventuel
36		
38		
3A		
3C	SUPLAR	largeur de ligne physique maximum
3E	SUPSW,	pour aiquillage d'action, en fin de page "TEKTRO"
	SUPTYP	attente après effaçage "TEKTRO
40	:LIMIT	
42	:INDK	indicateur argument puis réponse.
44	:SPOT	
46	:SCAN	
48	:MAXI	
4A	:SWITC	demande d'entrée clavier
4E		
50	TOPOU	temps de sommeil
52		
54		
58		
5A		
5C		
5R	:BRK,OCTEK	tampons récepteur caractère d'abandon, caractère d'entrée
60	ONE:C	tampon du caractère "ECHO"
62		tampon de réception de l'emplacement
64		du spot en "TEKTRONIX"
66	:BUFFR	tampon général des entrées sortie dont la longueur est un paramètre du système en temps-partagé.

II. 3. La communication entre le processeur APL et le système multiconsole.

Le programme APL accède à cette zone par indirection relative au pointeur qui est fourni dans INTSPV de la CDS de chaque usager APL sur la tête du bloc ainsi constitué. Les sections de l'IMT APL chargées des entrées-sorties sont donc constituées grâce à cette description virtuelle disponible d'un point de vue programmation dans leur zone de données.

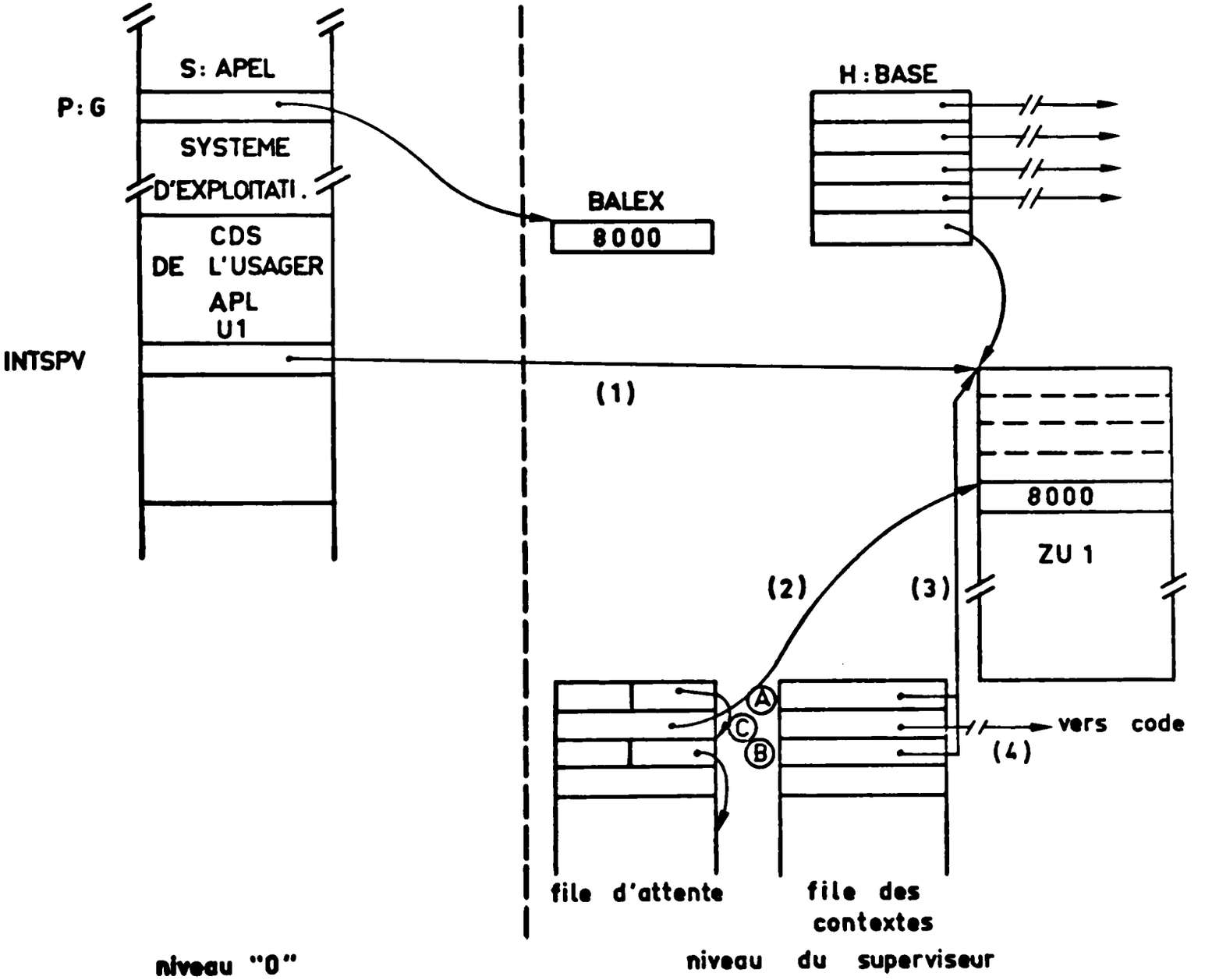


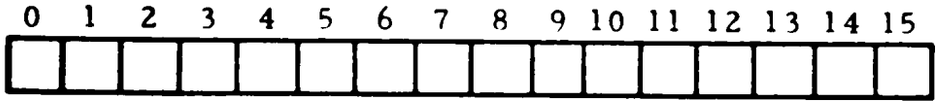
Fig. 6.5

Schéma décrivant les liaisons existant entre

- Le système multiconsole et la tâche APL
- Le système multiconsole et le système d'exploitation.

Lorsqu'une tâche APL prend le contrôle de l'unité centrale en régime stabilisé, la liaison (1) existant, le planificateur des travaux met en place les informations nécessaires pour que la boucle de scrutation soit prête à répondre aux sollications de la tâche et ce, sans ambiguité, puisqu'il n'y a qu'un seul utilisateur de l'unité centrale à un moment donné.

Le pointeur de la boite aux lettres de la zone usager de la tâche permet de détecter en cours de scrutation des demandes d'entrées-sorties sur le terminal selon la convention suivante, la boite aux lettres étant codée sur un mot. (La position du mot dans la zone est indexée par BALPRI).



Le bit (0) concerne la gestion des niveaux superviseur, celui-ci étant à l'affût du basculement du bit à 0.

Le bit (1) concerne la gestion de l'entrée-sortie, le processeur APL étant à l'affût du basculement du bit à 0 lorsqu'il à demandé une transaction.

bit 0	bit 1	Signification
0	0	demande de transaction de la part du niveau 0
0	1	
1	0	niveau superviseur en position d'attente ou transaction demandée terminée.
1	1	la transaction demandée n'est pas terminée.

Configurations possibles de la boite aux lettres terminal.

La Fig. 6.5. visualise les liaisons existantes entre le système multi-console et la tâche APL.

II. 4. Le traitement d'une transaction d'entrée-sortie par le processeur APL.

Ce traitement, vis à vis du niveau "0", s'effectue en deux temps :

- Un remplissage de la zone utilisateur de manière telle que le superviseur puisse gérer l'ensemble de l'entrée-sortie, de façon autonome: si une sortie sur terminal est demandée il y a recopie du message dans le tampon disponible, puis la mise à 0 de la boite aux lettres accompagnée de l'activation du niveau superviseur.

- L'attente active sur le basculement du bit (1) à 0 de la boite aux lettres et récupération du tampon si une entrée de ligne de caractère avait été commandée ainsi que les retours d'arguments pour gérer correctement cette ligne.

Analysons la séquence vis à vis du superviseur multi-console.

a) La boucle de scrutation détecte le basculement à "0" du bit (0) de la boite aux lettres sur laquelle pointe la case résidente affectée aux transactions sur terminaux.

b) Le contexte réduit est installé ; il y a branchement dans le code traitant la demande. Le code transaction en cours est positionné dans la boite aux lettres.

L'analyse préliminaire des arguments permet la détermination de la transaction selon les conventions suivantes :

positionnement de : INDK	conditions supplémentaires	fonction
nul lecture de "spot" ou sorties	:LIMIT = 0 :LIMIT = 0	lecture de "spot" en "tektronix" émission d'un message de :LIMIT octets
négatif entrée par lot de caractères	:SWITC 0 :SWITC = 0 :SWITC 0	spécification de révision de fonctions entrée du type espace pris en compte entrée du type espace ne comptant pas
= PECAR = PEGRF = PEXIT	" :SWITC "	lecture d'un octect au clavier lecture de :SWITC caractères graphiques demande de fin de session.
extension au traitement des sémaphores et à la commande de tâche		

c) Préparer la tâche APL à perdre le contrôle de l'unité centrale si le critère de commutation de tâche est rempli, par l'appel à la séquence spécialisée M2SWAP. Toute requête entraînant un séquencement passe par ce mécanisme.

d) Effectuer la transaction demandée qui peut être très discontinue, c'est à dire faire appel plusieurs fois au mécanisme de la file d'attente.

e) Terminer la transaction par la séquence spécialisée TERMES.

REMARQUE :

Pour ne pas alourdir l'exposé, nous avons volontairement omis de détailler une partie non négligeable de la réalisation concernant

a) La programmation effective des sections modifiées du processeur APL.

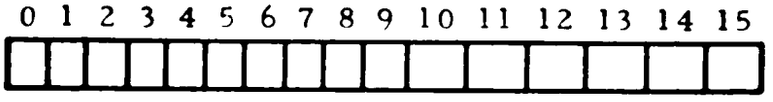
b) La programmation effective des entrées-sorties sur terminal conversationnel au niveau du superviseur.

Nous avons cherché plutôt à montrer l'outil et la manière de s'en servir que l'application qui en est faite.

III - ORGANISATION DES TRANSACTIONS PASSANT PAR LES MODIFICATIONS DU MONITEUR MTRD-E.

III.1. Principe.

Le principe est valable pour une tâche de type non APL. On se rapportera au schéma de la liaison superviseur et système d'exploitation. La boite aux lettres unique est positionnée par la tâche contrôlant l'unité centrale par la mise à 0 du bit (0) du mot la constituant, désigné par BALEX en accès général direct dans la CDS du programme multi-console.



Bit (0) = 0 demande de transaction.

Bit (0) = 1 niveau superviseur libre.

Nous pouvons contater dans cette optique que le niveau superviseur est toujours libre vis à vis des communications passant par le système d'exploitation. Dès que la demande est détectée, via la boucle de scrutation, la séquence de programme PDEMEX est déroulée.

a) Remise à 1 du bit (0) de la boite aux lettres

b) Restitution des argements d'appel stockés dans le module non réentrant du système d'exploitation et aiguillage sur la fonction demandée.

code le la transaction	argument principal	fonction	retour *
1) M:ASGN	adresse d'une table	assignation d'une étiquette opérationnelle à un fichier ou un périphérique	oui par registre
2) M:OPEN	étiquette opérationnelle	ouverture d'une étiquette opérationnelle argument secondaire : adresse dans certains cas.	oui par registre
3) M:CLOS	étiquette opérationnelle	fermeture d'une étiquette opérationnelle	oui par registre
4) M:IO	adresse du bloc contrôle	lancement d'une entrée-sortie	non
5) M:WAIT	adresse de bloc de contrôle	contrôle d'une fin de transfert	non
6) M:EXIT M:KILL M:ABRT		fin d'un programme abandon d'un programme suicide d'une tâche	non

* Retour d'argument du superviseur.

Nous allons répertorier les codes 1-2-3- caractérisant les opérations SGF 15 - reportées au niveau superviseur par la conception même du système d'exploitation. Signalons que les transactions exposées ci-dessous n'ont aucune répercution sur le séquencement des tâches.

III.2. L'assignation d'une étiquette opérationnelle.

Le superviseur travaille sur la table d'assignation dont l'adresse sera correctement positionnée vis-à-vis de la base G.

O.L	MODE
Nom EBCDIC du PERIPHERIQUE	
0	N°périph.
n°UEM	n°compte
NOM du FICHIER	

BIT	0 1	2	3 4 5 6 7
0		OL back	
1		OL Fore	

OCTET MODE

ADCB, contient le pointeur relatif à G superviseur de la table de données.

représentation de la table de données de l'assignation.

Le bit 2 de l'octet mode est forcé à 1 car l'assignation se fait au niveau d'interruption. Cette méthode permet d'utiliser la table de données sans recopie au niveau superviseur.

Le contrôle du nom EBCDIC du périphérique permet de déterminer si son type entraîne ou non son partage par les tâches APL.

III.2.1. La table des périphériques non partageables.

Il nous faut préciser la notion de périphérique non partageable : C'est en fait le tampon du périphérique qui n'est pas partageable. L'usager ayant le loisir de s'attribuer plusieurs étiquettes opérationnelles pour adresser son périphérique.

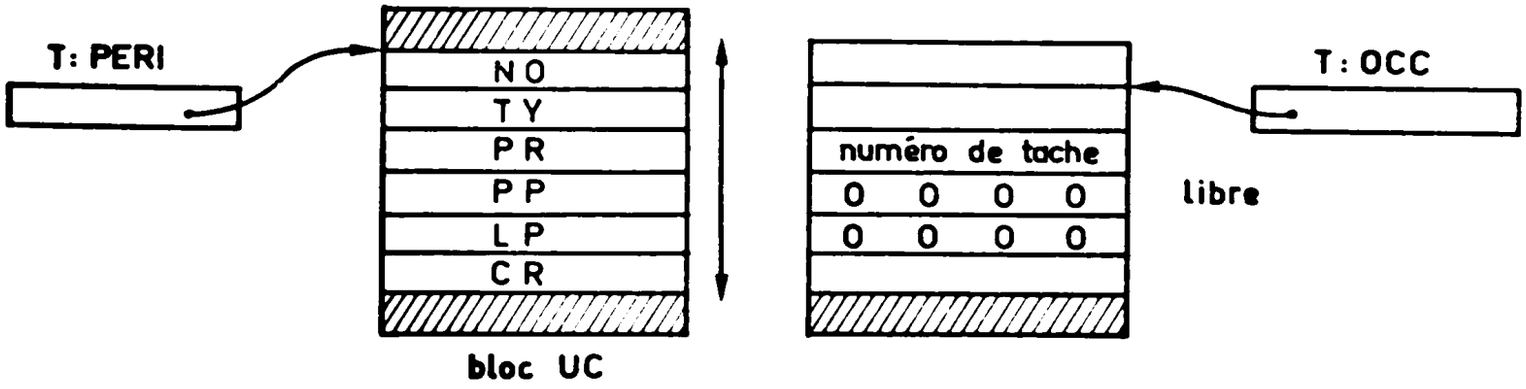


table des périphériques et table d'occupation des périphériques.

Le but de cette table est de mémoriser la configuration des périphériques non partageables connectés à l'unité centrale. Elle peut être étendue par blocs correspondant aux diverses unités de traitement présentés au système d'exploitation. Tout périphérique qui n'est pas consigné dans le bloc requis par l'assignation est considéré comme partageable.

III.2.2. Assignation d'un périphérique partageable.

L'action d'assignation est effectuée avec le niveau 0 connecté sur contexte d'attente car elle commande une entrée-sortie interne au module moniteur d'assignation pour lire les catalogues de fichiers. La réponse à l'assignation est transmise au niveau "0" par écriture dans le contexte de la tâche possédant la mémoire centrale.

Accumulateur	index
négatif (octet gauche = & 81)	3 ; erreur transfert disque 2 ; assignation interdite, nom inconnu 1 ; erreur table de données.
positif ou nul	0 ; assignation correcte.

Retour d'assignation.

III.2.3. Assignation d'un périphérique non partageable.

Parallèlement à la table des périphériques non partageables, la table d'occupation des périphériques permet de déceler si l'usager à le droit d'accéder à ce périphérique.

Si celui-ci est occupé pour une autre tâche, l'assignation est refusée par message standard. Sinon le numéro de la tâche est

consigné dans cette table et le résultat de l'assignation est rendu de la même façon qu'au §III.2.2. Si le périphérique appartient déjà à la tâche, "l'empilement" des assignations est permis. Ceci permet, par exemple, aucours d'une programmation en APL d'utiliser le périphérique en modes alphanumérique et binaire.

III.3. L'ouverture et la fermeture d'une étiquette opérationnelle.

L'opération d'ouverture est effectuée au niveau superviseur pour lever la protection des fichiers inhérente à l'emploi du processeur UGF 15 . En effet, le niveau d'interruption qui effectue une entrée-sortie sur un fichier doit être le même que le niveau d'interruption qui a fait l'ouverture du fichier : ceci explique la "montée" de l'ouverture au niveau superviseur pour permettre de lancer les entrées-sorties. La clôture du fichier et de son étiquette opérationnelle est reprise au niveau superviseur afin d'aligner les spécifications du système sur les autres opérations.

Les comptes rendu d'appel moniteur sont intégralement répercutés au niveau "0" par l'écriture directe dans le contexte. Les opérations sont effectuées au niveau "0" sur contexte d'attente pour éviter le déroulement du programme de fond lors des entrées-sorties sur disque internes aux modules (contrôlées par M:WAIT).

Accumulateur		index
négatif	1	OPEN interdit
		Déjà clos, erreur n°OL, OL non assigné.
	2	erreur transfert disque.
positif	0	OPEN correct
		clos correct.

IV - ORGANISATION DES ENTREES-SORTIES PASSANT PAR LES MODIFICATIONS DU MONITEUR MTRD-E

La communication au niveau superviseur de l'adresse du bloc de contrôle dirigeant l'entrée-sortie du niveau "0" permet de contrôler l'affectation de l'étiquette opérationnelle. Le principe relève du §III.1.

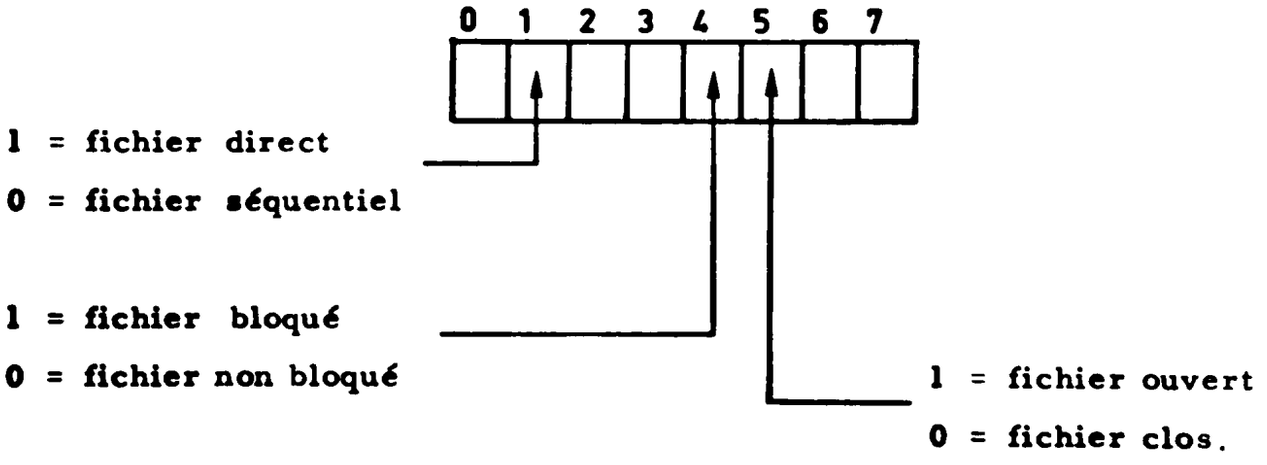
IV.1. Les entrées-sorties sur périphérique partageable.

Le système multi-console gèle la tâche en mémoire par incrémentation d'une unité de l'indicateur gel : FNSWAP. Cette action permet, moyennant quelques précautions, de lancer l'opération d'entrée-sortie sur le bloc de contrôle et le tampon prévus par le niveau "0".

Il faut en effet, passer des adresses relatives à G du programme niveau de fon, aux adresses relatives à la base générale du superviseur.

Nous contrôlons le statut du fichier assigné à l'étiquette opérationnelle afin de permettre la gestion correcte du bloc de contrôle par la consultation de la table OLFI du système d'exploitation.

La récupération de l'octet MODE de la case relative à l'étiquette opérationnelle en examen permet l'extraction des renseignements suivants :



a) Le fichier est du type accès direct : l'entrée-sortie passe par le mécanisme de la file d'attente piloté par Q : IOWA , la fin de transaction permet de décrémenter l'indicateur de gel.

b) Le fichier est de type séquentiel : il faut notamment détecter l'ouverture d'un fichier séquentiel bloqué qui nécessite la modification de l'adresse contenue dans un mot information supplémentaire du bloc de contrôle. Ceci n'est pas la seule particularité de cette classe de fichier pour laquelle l'entrée-sortie ne passe pas par le mécanisme d'attente du système multi-console car le module moniteur fait ses attentes lui même (nous placerons le niveau de fond sur contexte d'attente pendant l'opération).

D'une manière générale, la fin de transaction est propice pour organiser le séquencement des tâches.

IV.2. Les entrées-sorties sur périphérique non partageable.

IV.2.1. Schéma général de l'organisation des tables (Fig. 6.6.).

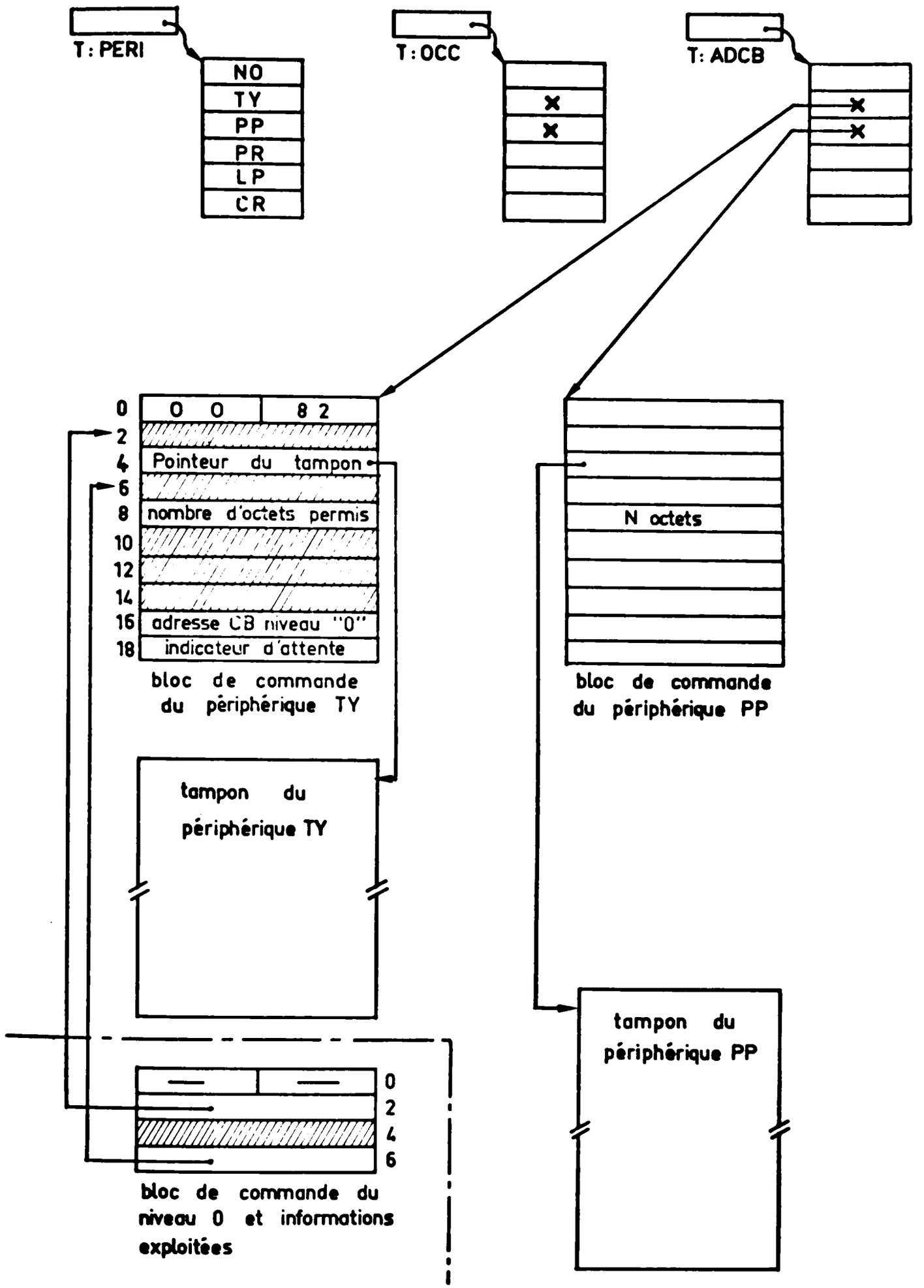


Fig. 6.6. Schéma représentant la chaîne contrôlant l'accès aux périphériques non partageables.

IV.2.2. Lancement d'une entrée-sortie.

a) Les entrées-sorties sur un même périphérique ne sont pas "empilables".

b) Les tampons résidents ont une longueur en octets normalisée sur le standard CII selon les règles suivantes.

SUPPORT.	BIN	AN
Carte	120	80
Ruban	120	80
Imprimante	132	132
K7	120	80

Néammoins ces longueur ne sont qu'un ordre de grandeur minimum.

Un contrôle est cependant effectué pour éviter lors d'un ordre d'entrée ou de sortie de dépasser la longueur permise du tampon résident.

c) A partir de l'adresse du bloc de contrôle du niveau de fond il faut recopier :

- L'ordre d'entrée-sortie et l'étiquette opérationnelle.
- Le compte d'octets mis en jeu par l'entrée-sortie.
- Eventuellement le tampon du niveau "0" dans le tampon résident pointé par le bloc de contrôle destiné au périphérique. Le pointeur au bloc de contrôle niveau "0" est protégé pour le traitement après la fin de transaction. (Mot 9 du bloc de contrôle au niveau superviseur).

d) L'indicateur d'attente est mis à 0 pour le séquencement et le contrôle de la transaction (mot 8 du bloc de contrôle au niveau superviseur).

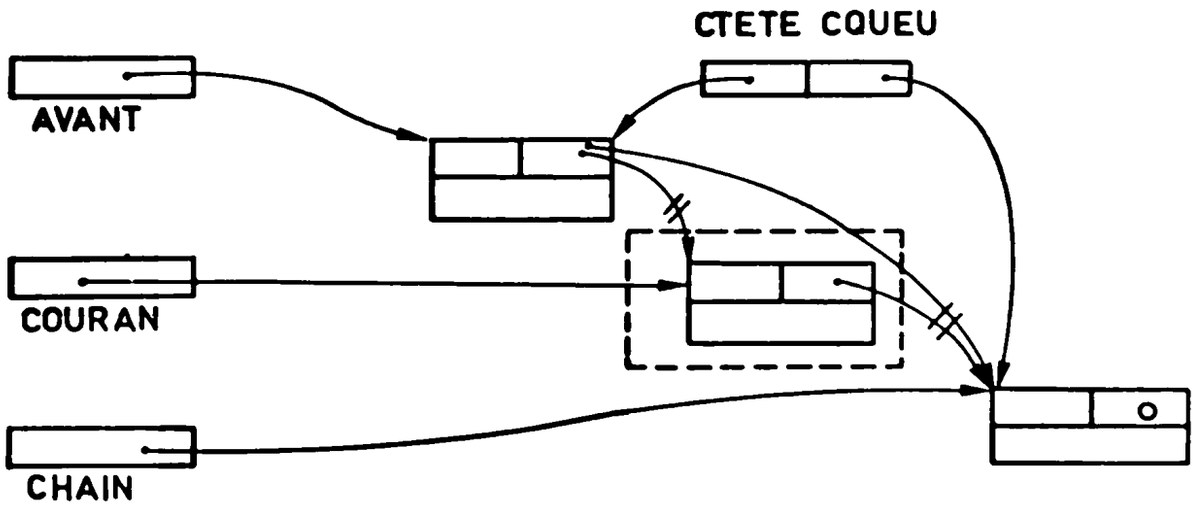
e) Le lancement s'effectue par la section Q:IOWA, en signalant le type de l'entrée-sortie très particulier, c'est à dire un événement lent.

Ceci est suffisant pour préparer la tâche à perdre le contrôle de la mémoire centrale selon le principe vu au chapitre précédent et sur lequel nous reviendrons en détail. Prenons comme hypothèse que la tâche ait quitté la mémoire sans nous préoccuper des modalités de ce départ. La tâche en sommeil, des événements lents peuvent se réaliser. Ceci impose l'attente de la descente de la tâche en mémoire afin de lui retransmettre le compte-rendu de la transaction et, en cas de lecture sur le périphérique, le tampon. Le maintien du souvenir de l'entrée-sortie est réalisé de la façon qui va être décrite.

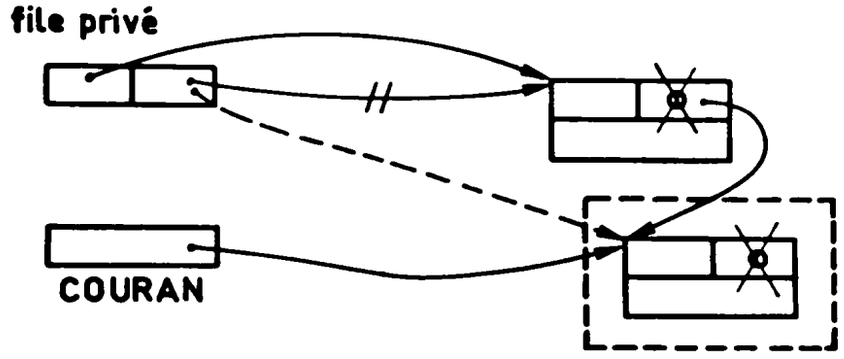
IV.2.3. Création d'une file privée.

Au niveau de la boucle de scrutation l'arrivée d'un événement lent lancé par une tâche qui n'est plus en mémoire déclenche un processus particulier. La case décrivant l'événement est enlevée de la file d'attente et incluse dans la description d'une file protégée dans la zone usager.

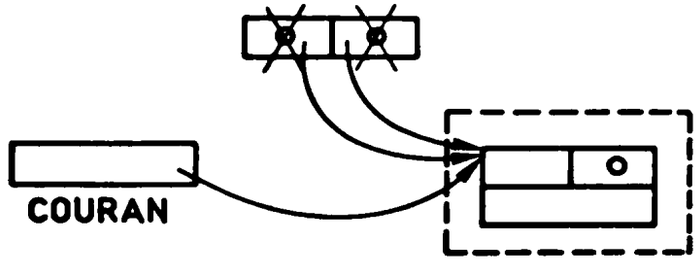
La préservation de l'état de la boucle de scrutation à la détection de l'événement n'est pas mise en cause.



La nouvelle file est ainsi composée : s'il y a déjà un événement lié, le chaînage est établi de la case la plus récemment pointée à la case "entrante", ce qui entraîne la modification du pointeur à la plus récente demande satisfaite.



Si la case est vide :



C'est le programme MINFA qui travaille sur la base L de la scrutation qui effectue ce travail.

IV.2.4. Réinsertion de la file privée dans la file des événements.

Lorsque la tâche réapparait en mémoire, la file où sont préservés les événements toujours à l'état arrivé est réinsérée dans la file d'attente. Une scrutation est de nouveau relancée pour satisfaire la fin de la transaction de chacun des événements. Ceci permet aussi de restituer les cases à la file vide. Cette réinsertion se fait selon le schéma 6.7 grâce au programme INOU qui

travaille sur la base L de l'ordonnancement. Lorsque la réinsertion est terminée, la description de la file est remise à zéro dans la zone usager.

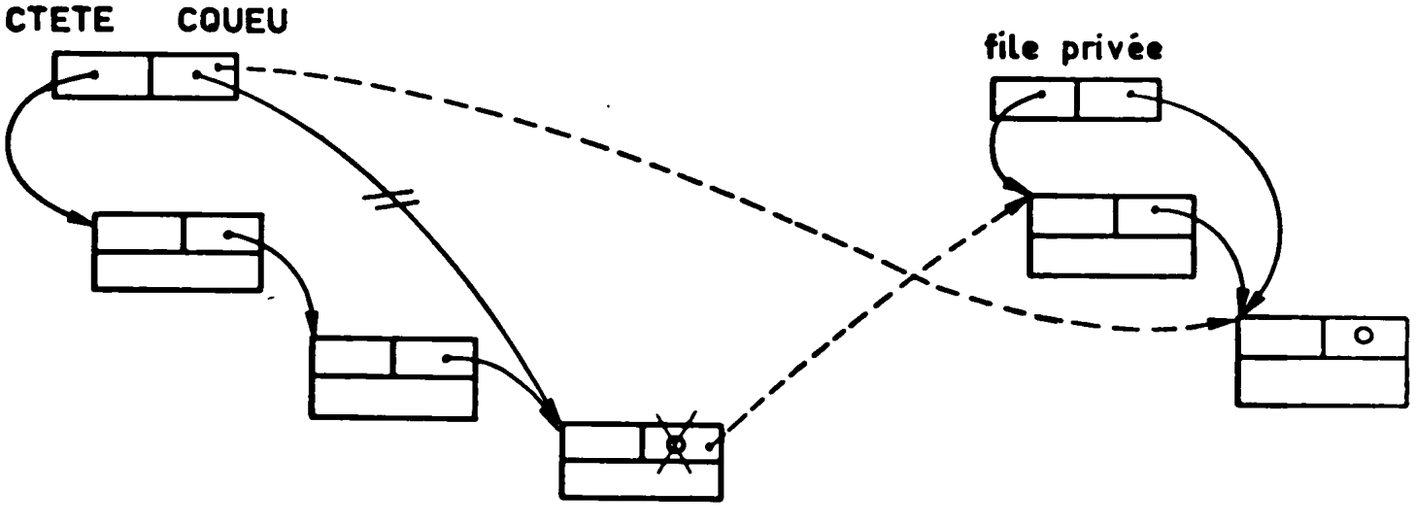


Fig. 6.7. Reinsertion de la file privée dans la file principale des évènements.

V - UNE ENTREE-SORTIE PARTICULIERE : LES ORDRES D'ABANDON.

La mise en réception d'un caractère de reconnaissance est une fonction spéciale des lignes asynchrones, le handler initialisant la ligne pour la réception d'un caractère dont le code est précisé dans le bloc de contrôle. La gestion de caractère de reconnaissance est faite automatiquement :

- Si un caractère est reçu autre que le caractère de reconnaissance il y a réinitialisation de la reconnaissance.
- L'ordre peut être interrompu par tous les autres ordres (réception et émission) et repris en fin d'exécution de ces ordres.

Nous avons vu que la boucle de scrutation passait systématiquement en

revue les blocs de contrôle des caractères de reconnaissance. La case élémentaire n'est pas déliée de la file d'attente lorsqu'il y a détection d'une fin d'événement reconnaissance de caractère. Le code de traitement est organisé pour effectuer les travaux suivants :

- Mettre la console en état de recevoir un caractère retour chariot pour connecter un usager au système.
- Reconnaitre le type de terminal et organiser la zone de l'usager en fonction de ce type.
- Comptabiliser les appuis de caractère d'abandon en régime stationnaire pour satisfaire les spécifications d'arrêt du processeur APL. Le niveau de fond pilote la procédure en faisant la mise à jour de la comptabilité. En effet le tableau donne les actions résultant des appuis sur l'abandon de l'usager.

nombre d'appui d'abandon	usager en entrée	usager en sortie	usager en mode calcul	usager en mode exécution de fonction
1	Le caractère d'abandon ne passe pas par la procédure d'abandon mais est géré par les programmes d'entrée	arrêt de l'impression après le dernier caractère du message en cours.	aucun effet	suspension de l'exécution de la fonction sur la fin de la ligne en traitement
2			arrêt de l'instruction en cours de traitement	abandon de l'exécution de la fonction.

Si la tâche pour le compte de laquelle la boucle de scrutation détecte une arrivée d'ordre d'abandon est en mémoire, l'appui est comptabilisé par incrémentation directe du mot prévu dans la CDS du niveau "0". Si la tâche est en sommeil, les appuis sont comptabilisés dans la zone de l'usager et répercutés dans la CDS de la tâche APL du niveau "0" lors de la réintro-

duction de celle-ci en mémoire. A ce titre le processus de va-et-vient a le pouvoir de contrôler la zone de l'usager.

La gestion du caractère d'abandon intervient au niveau de la synchronisation des tâches comme nous le verrons plus loin.

- 7 -

L'ORDONNANCEMENT DES TRAVAUX DANS LE SYSTEME

I - MECANISME DE FILE D'ATTENTE A LA RESSOURCE UNITE CENTRALE.

I.1. Principe.

Les tâches sont connues par leur numéro (0 étant exclu). Ce numéro ne reflète que le rang de la zone usager de la tâche dans la table des pointeurs des zones, y compris la tâche de type non APL.

L'évolution des tâches et leur concurrence pour l'obtention de certaines ressources (Unité Centrale, sémaphore) sont contrôlées par le jeu de files d'attente. A un moment donné une tâche ne peut appartenir qu'à une seule file d'attente.

Une file d'attente a une ressource constituée par une liste à double pointeur, au sein de tables de longueur égale au nombre de tâches gérées par le sytème.

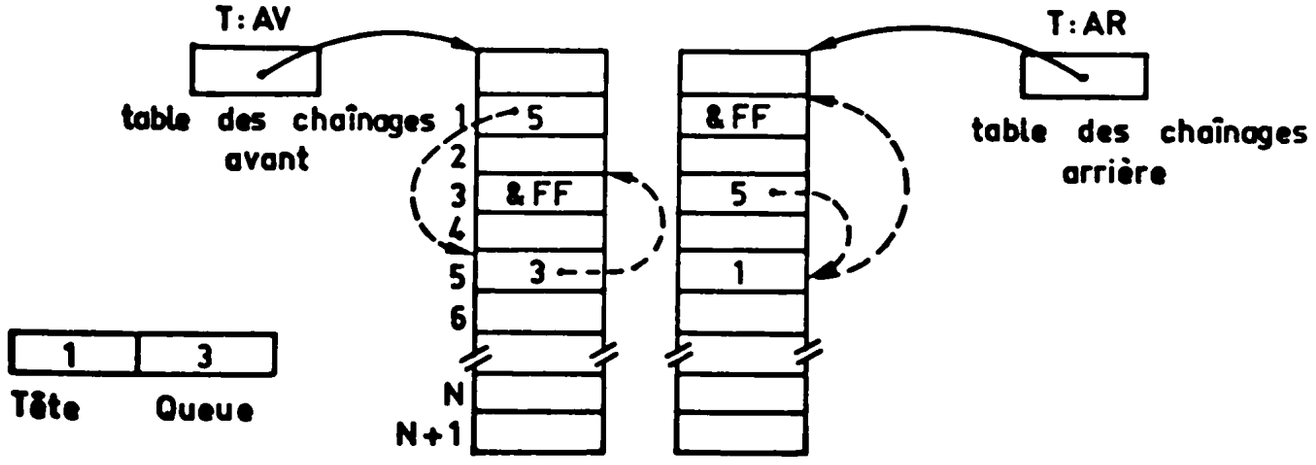


Fig. 7.1. Implantation des tables de chaînage Amont et Aval.

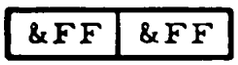
Les tables AV et AR sont des tables d'octets, chaque élément contient :

- 0 pour l'élément libre.
- &FF pour l'élément queue de file ou tête de file d'attente.
- Le numéro de la tâche amont ou aval.

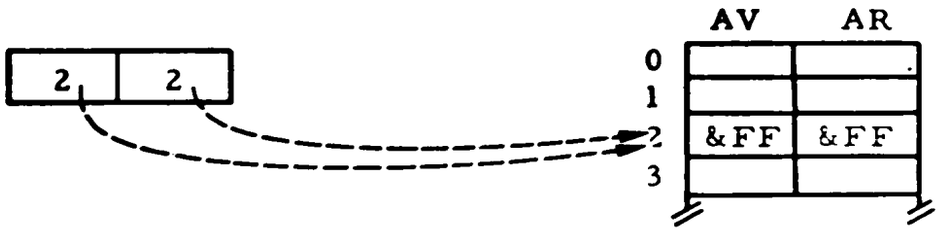
L'exemple 7.1. montre le chaînage des 3 tâches 1, 5 et 3 et l'organisation de celui-ci au sein des tables.

Quelques cas particuliers peuvent se présenter :

a) file vide



b) file à élément unique



La file est exploitée par le planificateur des travaux selon la méthode premier entré, premier sorti, mais le double chaînage permet d'y prélever un numéro de tâche à un emplacement quelconque.

En principe le système en temps partagé gère des tâches de type APL à priorité égale. La tâche supplémentaire est exploitée à la manière d'un travail de fond, ne faisant pas appel à l'introduction de la notion de priorité.

Néammoins une table décrivant les files d'éligibilité des tâches est réservée.

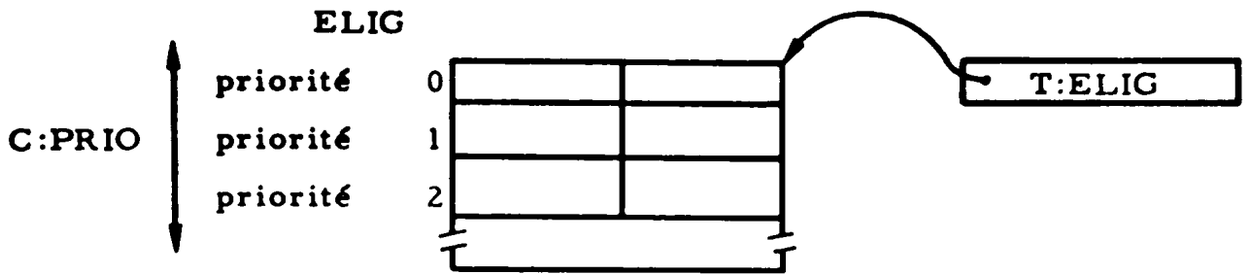
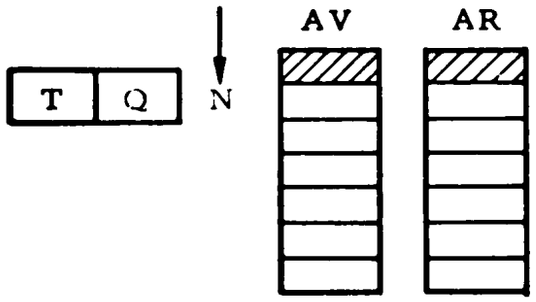


Table d'éligibilité des tâches.

Les sous programmes d'exploitation des files d'attente devront comporter comme arguments, l'adresse relative à la base G du superviseur de la description de la file et éventuellement un numéro de tâche.

I.2. Mise en file d'attente d'un numéro de tâche

L'ajout du numéro N s'effectue par la fin de file, il y a lieu de modifier le pointeur de queue de file:



a) si la file est vide on a T = Q = &FF.

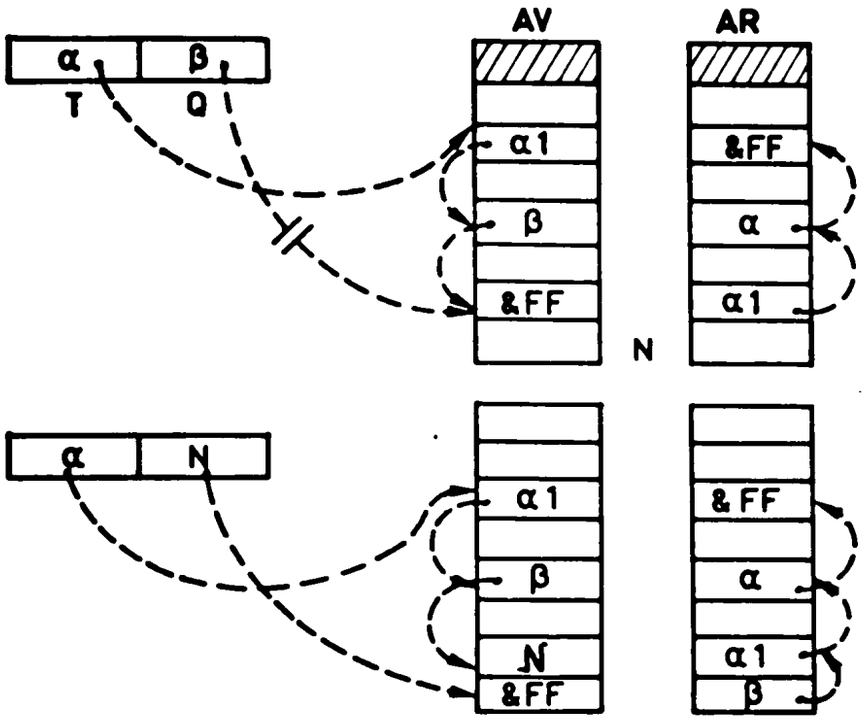
Il faudra faire :

AR (N) = &FF

AV (N) = &FF

N ⟶ T = Q

b) file quelconque



Il faut faire :

AV (β) = N

AR (N) = Q

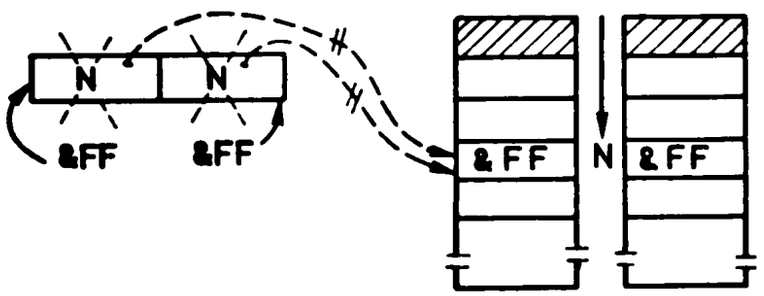
AV (N) = &FF

Et Q est mis à jour en pointant sur N.

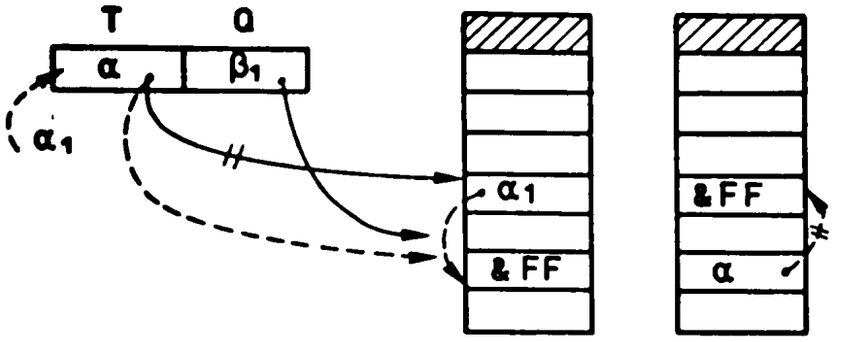
I.3. Prélèvement de la première tâche en attente.

a) Si la file est vide, la sortie est rendue avec un message approprié.

b) Si la file est à élément unique c'est à dire T = Q la mise à jour est simple puisqu'il suffit de mettre T = Q = &FF.



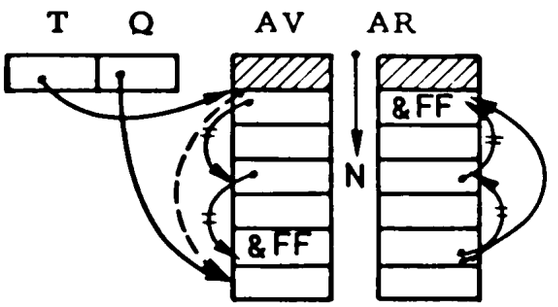
a) La file étant quelconque la mise à jour s'effectue sur la tête de file, telle que AV (α)←T et AR (T)← &FF



I.4. Prélèvement d'une tâche donnée dans une file.

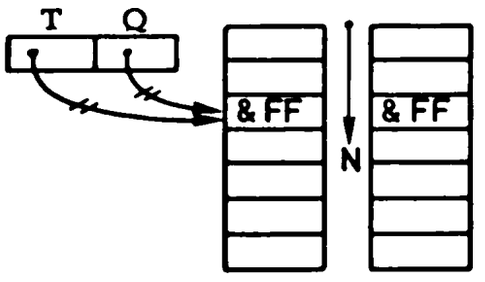
Soit à prélever les tâches N faisant partie d'une file d'attente dont l'adresse est donnée en argument du programme.

a) Tâche prélevée au milieu d'une file.



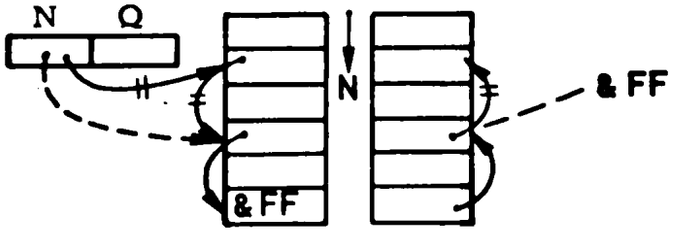
Le numéro est prélevé à condition de rétablir les chaînages de façon à combler la lacune.

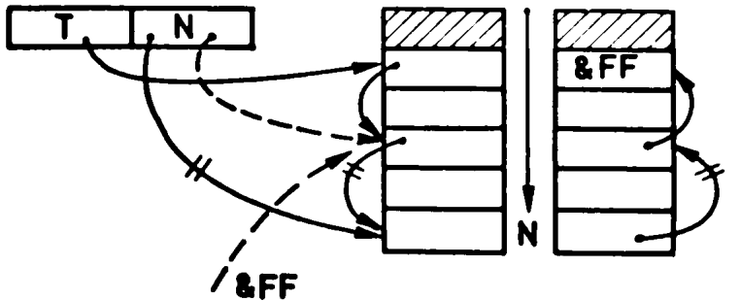
b) File unique.



La file est mise à jour de façon très simple en indiquant la file vide.

c) Tâche en début ou fin de file.





mise à jour d'une tâche en queue de file.

II - LE SEQUENCEMENT DES TRAVAUX DANS LE SYSTEME EN TEMPS-PARTAGE.

II.1. Principe.

La planification des tâches résulte de l'état d'un ensemble de variables gérées à l'occasion du traitement des différentes requêtes présentées au système. Nous pouvons faire la synthèse de ces différentes requêtes.

Boite aux lettres	fonction
Tâche A PL	A création d'un pseudo-évènement avec attente de réalisation
Système	B création d'événement d'entrée-sortie. B1 événement de périphérique non partageable . B2 événement de périphérique partageable. C attente d'un événement de B. D demande de mise à mort. E autre demande.

Et les diverses variables gérées à l'occasion de ces requêtes sont :

indicateur	type d'adressage	fonction.
FNSWAP	local à la tâche	compteur des requêtes pouvant donner lieu à la commutation des tâches.
FNSWAP	local	compteur des requêtes interdisant la commutation de tâches.
DEBT	global	indicateur d'épuisement du quantum de temps.
V:CUR	global	contient le numéro de la tâche en possession de la mémoire centrale.
P:CALL	global	indicateur de demande de commutation de tâche
SEMWAT	index de table	souvenir de la demande d'attente d'un événement lent dans le control bloc correspondant.

Le séquencement consiste a effectuer la commutation des tâches après avoir constaté l'occurrence d'un critère de changement d'utilisateur.

Ceci a lieu en général au début et à la fin d'un traitement de requête et en particulier est indissociable du traitement des entrées-sorties.

II. 2. Séquencement sur attente.

II. 2. 1. Attente sur appel moniteur M:WAI T.

Une demande d'attente sur l'appel moniteur est pris en charge dans le programme PDEMEX, elle concerne :

a) Un transfert sur périphérique partageable.

Aucune action de séquencement ne résulte de cette demande car la tâche qui a lancé la transaction est gelée en mémoire. L'indicateur FNSWAP ayant été incrémenté d'une unité, le traitement se termine par P:END2.

b) Un transfert sur périphérique non partageable.

Le programme du niveau "0" transmet l'adresse relative à la base G du bloc de contrôle sur lequel l'entrée-sortie aurait du être lancée. Celui-ci ayant été "doublé" au niveau superviseur, la correspondance s'établit lors de la recherche du périphérique auquel l'étiquette opérationnelle employée est affectée. Un indicateur d'attente, indexé dans le bloc de contrôle du superviseur par SEMWAT (mot 8), est pris à -1 . Le compteur de transaction commutable FSWAP est incrémenté de une unité, dans le cas où le transfert n'est pas terminé. L'appel du processus de changement de tâche est effectif par positionnement de P:CALL en se branchant dans P:END1 si la tâche n'est pas gelée en mémoire par ailleurs.

II.2.2. Attente de la réalisation d'un événement boite aux lettres clavier.

L'attente sur un événement boite aux lettres clavier est implicite. La tâche est donc déclarée commutable dès le début du traitement par appel à la séquence M2SWAP dont le rôle est de positionner P:CALL et d'incrémenter d'une unité le compteur FSWAP. Une sortie sur périphérique ne donnant pas lieu à un séquencement ne positionne pas de variable à cette occasion.

Néammoins, il peut arriver que la gestion de l'écran ou de la page nécessite une opération particulièrement longue.

- Effacement de l'écran(temps supérieur à une seconde)

- Photocopie (temps de l'ordre de plusieurs dizaines de secondes).
- Saut de page sur machine à écrire (temps de plusieurs secondes).

Nous dirons qu'il y a émission dégradée et tout se passera comme si une entrée sur clavier avait été commandée. L'appel à M1SWAP permet de réaliser cette transformation : après avoir conservé le souvenir de cette opération la séquence M2SWAP est exécutée.

La vitesse de transfert sur le périphérique conversationnel constitue un critère très important du système en temps-partagé. Sur un site d'exploitation peuvent se rencontrer, parfois mélangés, terminaux de type visualisation ou terminaux de type machine à écrire. Si la vitesse de transfert est importante (9600 bauds), la visualisation ne pose pas de problème. En revanche, les machines à écrire aux vitesses de transfert lentes (300 bauds) peuvent parfaitement introduire un problème de commutation de tâche. A cette vitesse, l'émission de 15 caractères demande 500 ms, de l'ordre du temps de commutation des tâches. Dans l'état actuel de la réalisation, cette caractéristique n'est pas exploitée étant donné que les terminaux machine à écrire choisis possèdent un tampon interne, ce qui leur permet une vitesse de transfert de 1200 bauds. La vitesse de 300 bauds constitue donc une frontière critique où il faut envisager :

- La commutation de tâches au niveau du superviseur.
- L'empilement des ordres de sortie au niveau du processeur APL pour que la commutation soit intéressante (l'impression la plus simple est en général exécutée en plusieurs ordres).

II. 3. Séquencement sur réalisation d'événement.

II. 3. 1. Interruption de l'horloge.

La séquence TIMER est exécutée à l'épuisement du quantum de temps : l'indicateur de demande de changement de tâche P:CALL et l'indicateur de débordement de temps DEBT sont positionnés. La séquence est terminée par P:END2. L'appel de la séquence provient de l'examen de l'événement résident dans la file d'attente, le bloc de contrôle du délai est alors inhibé en positionnant l'événement en attente artificielle. Le bloc de contrôle sera réactivé à la prochaîne attribution d'un quantum de temps .

II. 3. 2. Réalisation d'un événement dû à un périphérique partageable.

L'appel à la commutation de tâche est réalisé si la tâche est dégelée de la mémoire centrale (FNSWAP = 0) en situation de débordement de temps, sinon, un quantum de temps n'étant pas épuisé, la tâche peut continuer à travailler sauf si elle a lancé par ailleurs une transaction commutable : il y a alors positionnement de P : CALL par la séquence de fin P:END1

II. 3. 3. Réalisation d'un événement dû à un périphérique non partageable

L'analyse s'effectue après avoir déterminé le type de l'événement en cours de scrutation. En effet il y a décrémentation d'une unité du compteur FSWAP si l'événement a été soumis à une attente et a provoqué le sommeil de la tâche. La tâche est remise en file d'attente à l'éligibilité mémoire pour un nouveau service avec positionnement de l'indicateur P:CALL si les conditions le permettent. Le retour dans la scrutation ne passe par aucun des chemins aménagés dans le superviseur.

II. 3. 4. Fin d'un traitement boite aux lettres "clavier".

Le traitement concerne aussi une sortie dégradée. En entrée,

lors de la détection de l'une des deux commandes :

- caractère "Retour Chariot" pour la fin de ligne,
- caractère "LINE Feed" pour le défilement des tampons.

La séquence TERMES est attaquée. Après la remise à jour de la boite aux lettres (code &8000), il y a décrémentation de une unité du compteur FSWAP. Lorsque celui-ci est significatif de l'absence de transaction commutable (valeur 0), on vérifie que la tâche est bien présente en mémoire, auquel cas le traitement s'arrête là par P:END2.

En revanche, la tâche étant en sommeil, il faut la remettre dans la file d'attente à l'éligibilité mémoire. La séquence P:END0 a pour but d'examiner le statut de la tâche actuellement en mémoire et à positionner l'indicateur : P:CALL si :

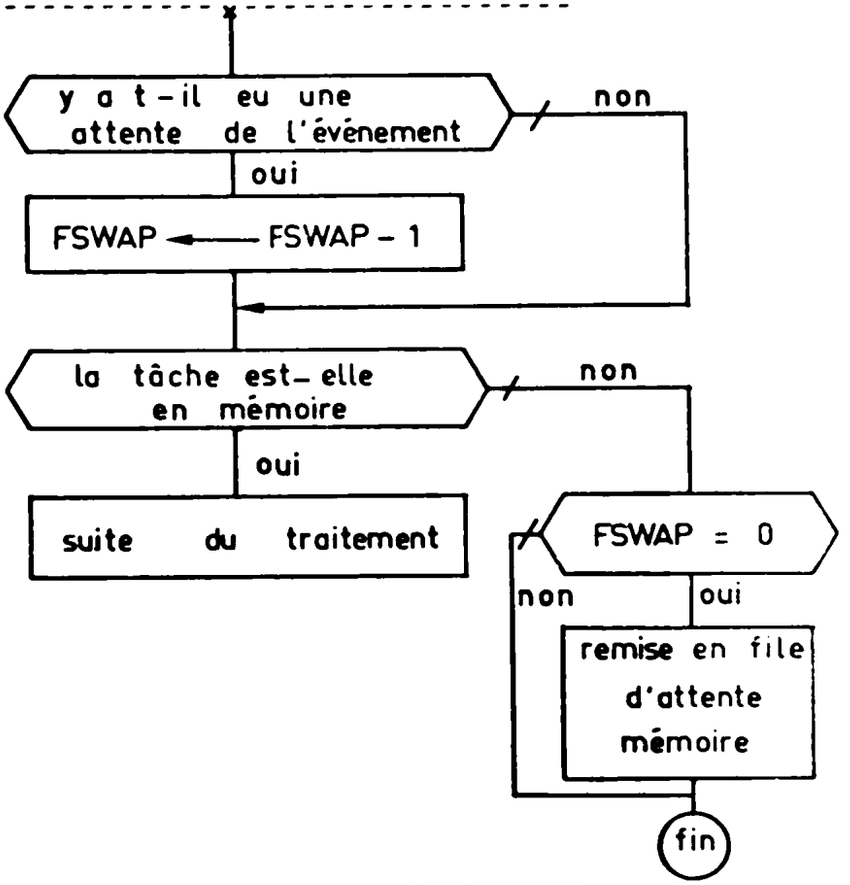
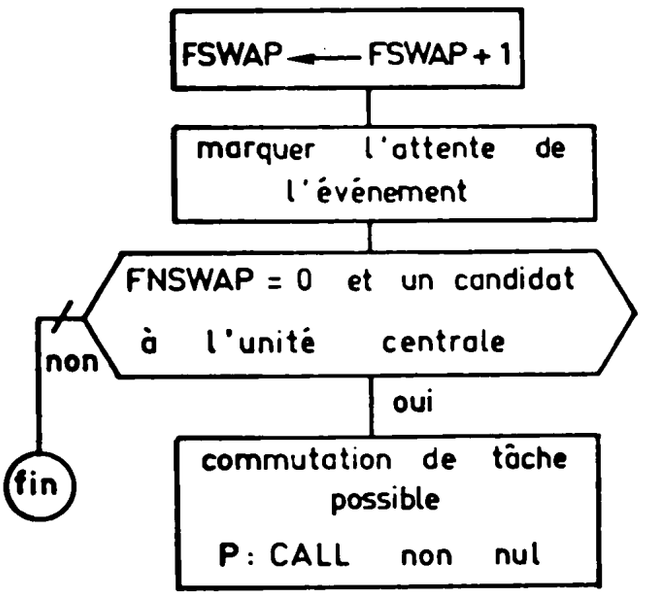
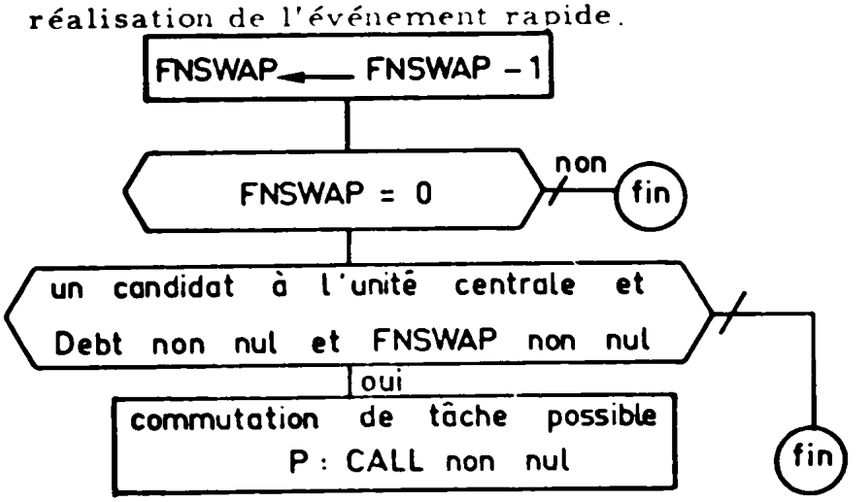
- La tâche en mémoire est en débordement de temps, elle peut être interrompue si son compteur FNSWAP est également nul, signalant qu'elle n'est pas gelée en mémoire.
- La tâche a son indicateur FSWAP différent de zéro, elle peut être interrompue si elle n'est pas gelée en mémoire
- par ailleurs.

REMARQUE :

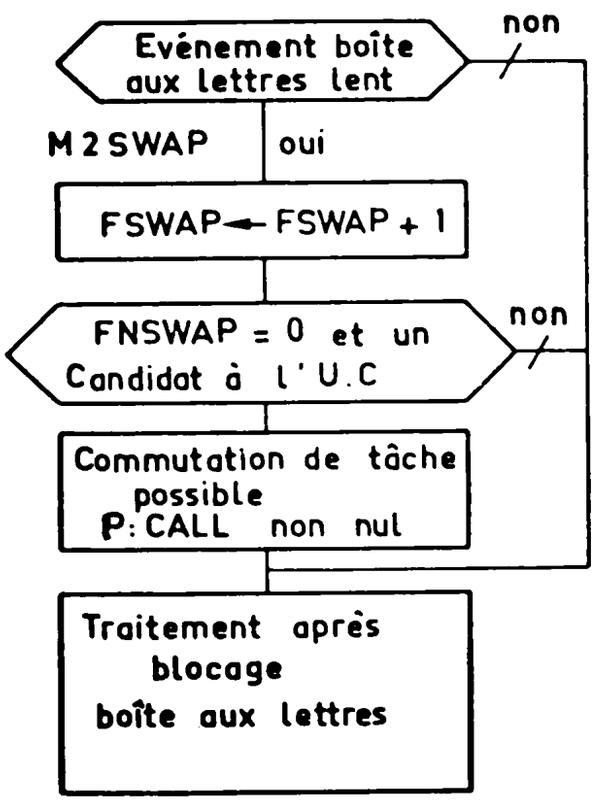
Une tâche qui a été déclarée éjectable par positionnement de l'appel à la commutation P : CALL pour le traitement de la boite aux lettres clavier ou sur entrée-sortie lente peut perdre une partie du quantum de temps si aucune autre tâche n'est éligible à ce moment. En effet, la tâche sera considérée en débordement de temps même si la cause de la commutation cesse entre -temps.

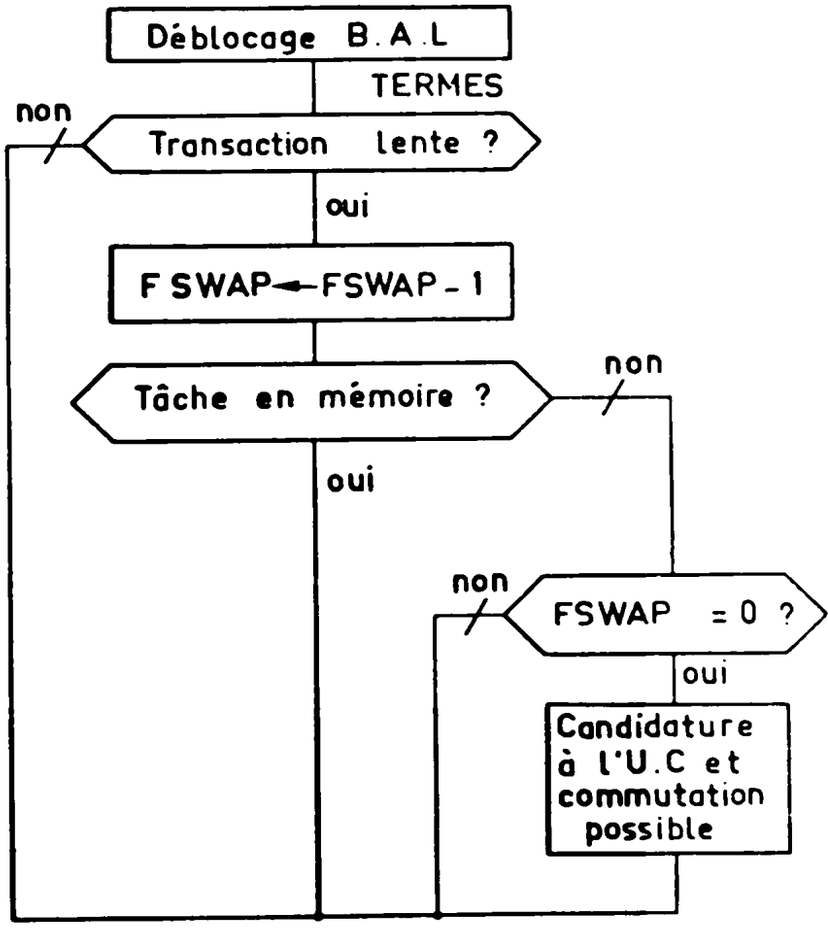
II.4. Synthèse du séquencement.

Nous donnons ici les actions résultant de l'exposé qui vient d'être fait :

sur commande du niveau "0"	sur autre activation du niveau superviseur.
Pas de correspondance.	Interruption horloge. DEBT 0 Si 1 candidat U.C et FNSWAP 0 de la tâche en mémoire : commutation de tâche possible
Création de l'événement lent. Ne rien faire.	Réalisation de l'événement lent 
	
Création de l'événement rapide. FNSWAP FNSWAP + 1	réalisation de l'événement rapide. 
attente de l'événement rapide. ne rien faire.	

demande d'assignation etc. . ne rien faire	pas de correspondance.
demande de mise à mort arrêt des E/S en cours pour le compte de l'usager commutation de tâches possible	libération éventuelle de blocs de contrôle de périphérique non arrêtés par ORDRE STOP.
création et attente d'un événement boite aux lettres.	réalisation d'un événement boîte aux lettres





III - LES ETATS D'UNE TACHE.

III.1. Principe.

La table ETAT en octets est destinée au contrôle de l'état de la tâche pour la planification des travaux selon le code suivant.

Code	valeur	fonction	Moment de modification
K:NIL	80	code d'état pour console libre	Fermeture du fichier de sauvegarde
K:INI K:DEM	0 1	console connectée console démarrée	Traitement du caractère de reconnaissance.
K:CUR K:FIN	2 3	tâche en état courant tâche terminée	changement de tâche traitement d'une fin de tâche.

III.2. Démarrage d'une tâche.

III.2.1. Initialisation du système en temps-partagé et reprise après une fin de tâche.

La table d'état est initialisée à la valeur K:NIL, les blocs de contrôle du caractère de reconnaissance dans l'état événement arrivé.

La scrutation forcée, au démarrage du système en temps-partagé, détecte pour chaque tâche cette pseudo-réalisation.

Le code K:NIL a pour but de mettre la console en état de réception de caractère de reconnaissance. Le traitement comporte la vérification de l'état de la ligne asynchrone : si la ligne ne répond pas à la mise au repos préalable, elle est inhibée de façon

simple en imposant un état d'attente de fin de transfert dans le bloc de contrôle. La mise en réception du caractère de reconnaissance fait basculer l'état de la tâche à K:INI. Lors d'une fin de tâche APL, la réinitialisation de la console est faite selon le même principe. Le caractère de reconnaissance est "Retour Chariot" pour des raisons de contrôle de parité plus aisée.

III.2.2. La console a reçu le caractère de reconnaissance.

La scrutation détecte l'arrivée de la fin de transfert du caractère de reconnaissance. L'aiguillage d'après l'état K:INI de la tâche, lance une séquence de reconnaissance du terminal désirant se connecter. C'est une commande destinée à mettre en relief les caractéristiques spécifiques à chaque terminal utilisant des blocs de contrôle émission et réception. Cette commande provoquant des attentes, le bloc de contrôle du caractère de reconnaissance est inhibé temporairement.

terminal	mode de reconnaissance	résultat de la commande
tektronix 4013/4015	automatique	similaire à une lecture de "spot"
"Diablo" ou "HYPRINT"	automatique	réception de caractère "ACK".
Anderson jacobson 832	manuelle	réception de 5 caractères identiques par 5 appuis.

Lorsque le terminal est reconnu, ce bloc de contrôle voit sa fonction transformée en reconnaissance de caractère d'abandon. "ESCAPE" avec la parité ajustée, le transfert est initialisé et, à partir de ce moment, la tâche est toujours à l'affût de l'abandon par relance systématique. L'état de la tâche est basculé à K:DEM et le numéro de la tâche est mis en file d'attente de la mémoire par la séquence P:END0.

III.2.3. Démarrage de la tâche supplémentaire.

Cette tâche étant initialisée par commande de l'un des usagers, son statut passe de K:NIL à K:DEM sans transition.

III.3. L'état courant d'une tâche.

Quel que soit son type, dès qu'une tâche est admise dans la planification des tâches, son état est passé à K:CUR. Comme nous avons pu le voir ; la tâche peut se trouver sous ce code dans des états spécifique à l'ordonnancement des travaux :

- En mémoire Centrale
- En sommeil sur disque : mise en file d'attente pour un autre service.
- En sommeil sur disque : en dehors de toute file d'attente.
- En sommeil sur disque : en arrêt sur un sémaphore.

C'est le mécanisme du séquencement et de la synchronisation qui agit sur ces sous-états de la tâche d'une manière logique sans s'appuyer sur un code d'état plus complexe.

III.4. Fin d'une tâche.

La séquence est la même que ce soit sur fin normale (EXIT) ou fin anormale (ABRT) du programme en niveau 0. Elle est attaquée soit par les modifications implantées dans le moniteur d'exploitation soit par les séquences de traitement d'erreur à l'occasion de la manipulation des fichiers de sauvegarde.

Le code K:FIN signale l'arrêt de la tâche. La file d'attente des événements

est parcourue pour détecter les événements en cours de type lent et banal appartenant à la tâche. Si les événements sont accomplis ils sont retirés de la file d'attente, sinon un ordre d'arrêt du transfert est lancé sur le périphérique auquel l'étiquette opérationnelle du bloc de contrôle de l'événement en étude est assignée. L'événement est alors retiré de la file d'attente. Certains événements ne peuvent être arrêté de cette manière, il suffit alors de modifier le contexte réduit pour que, à la réalisation de l'événement, le traitement suivant puisse être appliqué. L'étiquette opérationnelle doit être rendue au système en temps-partagé par affectation de celle-ci à T:NO : annulation d'étiquette. Si elle est assignée à un périphérique non partageable celui-ci est aussi rendu au système en temps-partagé.

D'autre part la tâche a pu, au cours de sa vie, manipuler des sémaphores. Ceux-ci peuvent être rendus au système grâce au catalogue privé de la tâche par une opération similaire aux opérations de destruction de sémaphore, les opérations V libérant les tâches en arrêt derrière les sémaphores étant effectués lorsque ceux-ci sont d'exclusion mutuelle.

L'ensemble de ces opérations permet de préparer l'effacement du numéro de tâche du séquencement. La tâche est alors en mémoire, mise sur un contexte d'attente, ses indicateurs FSWAP et FNSWAP mis à zéro et son bloc de contrôle de reconnaissance d'abandon positionné à l'état "arrivé"

L'appel à la séquence P:END1 permet la commutation de tâche. N'oublions pas que l'ordre de fin de programme émane de la boite aux lettres système placé en tête de la boucle de scrutation, le retour en analyse provoqué par P:END1 ira détecter la reconnaissance d'abandon réalisée mais le code K:FIN empêche toute action aussi longtemps que le planificateur SWAP n'a pas réinitialisé la tâche.

IV - ORGANISATION DE LA COMMUTATION DES TACHES.

IV.1. Principe.

IV.1.1. Généralités sur les informations à gérer.

Le changement de tâche consiste à vider la tâche actuellement en mémoire sur le support secondaire et à rappeler en mémoire la première tâche éligible dans la file d'attente à l'unité Centrale. En plus de ces fonctions le planificateur de travaux SWAP appelé par la boucle de scrutation, a pour mission d'initialiser les informations nécessaires à l'établissement des relations entre la tâche entrante et le système en temps-partagé.

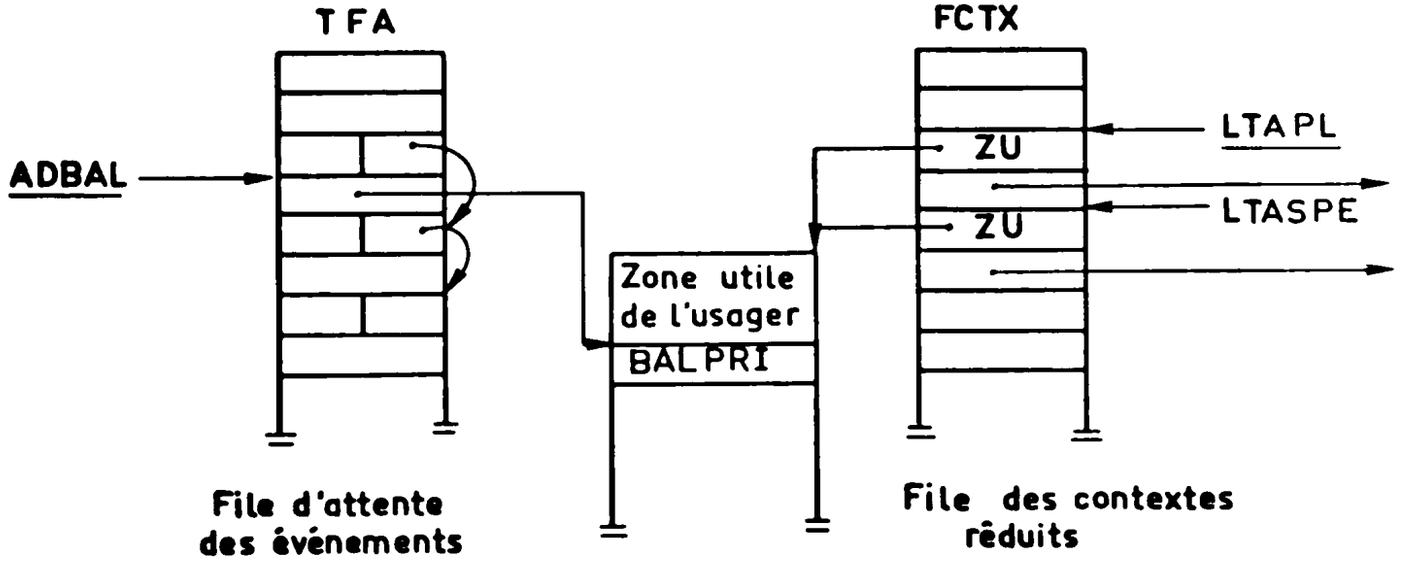


Fig. 7.2. Informations mises à jour par SWAP.

nom du pointeur.	contenu de la case pointée	fonction
ADBAL	pointeur de la boite aux lettres des demandes clavier.	prépare le système à recevoir les ordres d'entrée-sortie.
LTAPL	adresse de la zone usager en mémoire.	prépare le système à exécuter les codes de traitement réentrant
LTASPE	idem.	

Les variables importantes manipulées sont les suivantes :

information	accès	fonction.
BASEGO	général	adresse relative à G superviseur de la base G de la tâche en mémoire.
DEBT	général	Indicateur de débordement de temps.
V:CUR	général	Numéro de tâche contrôlant la mémoire.
V:SWAP	général	Indicateur de changement de tâche en cours.
V:TASK	local	Numéro de tâche contrôlant la mémoire.
V:ELU	local	Numéro de tâche éligible en mémoire
V:ETAT	local	Etat de la tâche éligible.

Lorsque la commutation de tâche est décidée l'indicateur V:SWAP est rempli négativement.

Ceci permet au niveau des entrées-sorties d'inhiber les action suivantes :

- Demande de séquencement, celui-ci étant en cours de déroulement.
- Ecriture ou lecture de données au niveau "0" car elles peuvent avoir été modifiées par l'écriture / lecture disque.

La décision de vider la tâche actuellement en mémoire revient à changer V:CUR par le numéro de tâche prête éligible. La coïncidence entre V:CUR et le numéro de tâche traité par le processus de la file d'attente (NUMAC) étant suffisant dans la plupart des tests.

IV.1.2. Les contrôles préliminaires.

Les indicateurs contenus dans la zone usager de la tâche en mémoire (V:TASK) sont lus :

- FNSWAP non nul provoque le refus de commuter la tâche gelée en mémoire.
- FSWAP non nul empèchera la remise en file d'attente à l'unité centrale après le vidage.

Cette remise en file d'attente sera accomplie, rappelons le, lors de la dernière réalisation d'événement annulant cet indicateur selon les principes donnés lors du paragraphe concernant les séquencements des tâches .

Rappelons enfin que le niveau "0" est mis sur contexte d'attente lorsque la commutation est permise.

IV.2. Introduction d'une tâche en mémoire.

La première tâche de la file d'attente à la ressource est prélevée, ceci remplit :

- V:ELU : numéro de la tâche présentée à l'éligibilité en mémoire.
- LELU : adresse de la zone utile de cette tâche.

IV.2.1. Traitement de la tâche APL.

a) Première introduction de la tâche.

Une PRT et CDS fraîche est copiée en mémoire par lancement d'une entrée-sortie sur le fichier de sauvegarde grâce au bloc de contrôle constitué dans lequel l'étiquette opérationnelle correspond à ce fichier. Le fichier ayant été précédemment assigné à la tâche par CREER.

L'implantation de la CDS fraîche sur ce fichier débute en secteur 0.

Le contexte de démarrage standard de l'APL est aussi recopié.

Les informations servant à établir la communication de la tâche APL et de sa zone utile ont été partiellement abordées, le tableau suivant fait la synthèse de toutes ces informations.

Mot de la CDS ou Octet	APL	Origine en superviseur de l'information	fonction
INTSPV	M	LELU contenant l'adresse de la zone utile.	Pointeur relatif à C de l'APL de la zone utile.
PERIPH	M	SUPPER de la zone utile	type de périphérique connecté.
CBDAB+4	M	AD:BUF de la zone utile	pointeur relatif à C de l'APL.
CBDSQ+2	M	SIZBM2 donnée de génération	longueur en octet du tampon.
CBDSQ+3	O	COUPLE en "SWAP"	étiquette opération-nelle de travail pair pour la tâche.
CSACT	O	V·ELU en "SWAP"	numéro de la tâche.

Le schéma 7.2 est mis en place.

b) Introduction de la tâche APL en régime stationnaire.

Informations concernées	origine	fonction
LELU	SWAP	adresse de la zone utile de la tâche entrante.
OLELU		étiquette opérationnelle d'E/S sur le fichier de sauvegarde
KSWIN		indicateur d'entrée en cas d'erreur d'E/S sur fichier.
FILEAT	zone utile	file d'attente privé des événements lents accomplis
T:DAB		compteur des abandons pendant le sommeil de la tâche.
CEXIT		arrêt de la tâche derrière un sémaphore pointé par ce mot
:TOPOU		références des temps.
:TOPOU+2		

Deux ordres d'entrée-sortie sont lancés dans l'ordre.

- PRT + CDS qui passe par le mécanisme d'entrée-sortie.
- Espace de travail qui est lancé au sein du programme.

Pendant le transfert, les mots SRD sont recopiés dans les emplacements réservés puis l'événement est mis en file d'attente.

Avant d'établir le schéma 7.2. des compte-rendus sont donnés à la tâche APL entrante.

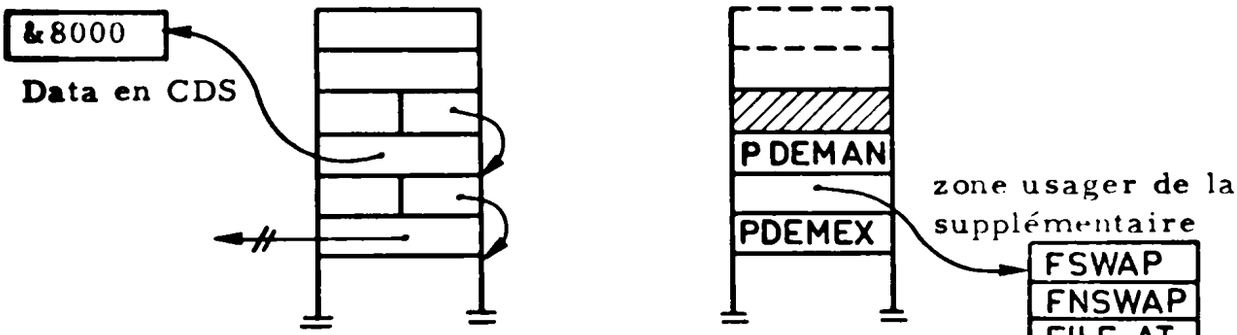
Information en CDS APL	origine	fonction
TESDAB	T:DAB	"TESDAB" mis à jour par la valeur T:DAB souvenir des appuis pendant le sommeil.
CBDAB		mis à zéro si T:DAB positif non nul.
REFTE	:TOPOU	référence du temps (codage sur 2 mots en flottant).
REFTE+2		

IV.2.2. Traitement de la tâche supplémentaire.

a) Première initialisation.

Elle consiste à charger l'IMT de la tâche à partir de la bibliothèque EP, suite aux indications fournies par la tâche mère dans sa zone utile (NOM en EBCDIC). Les informations concernant l'implantation de son contexte et la longueur du programme chargé sont suffisantes pour pouvoir faire les opérations de sauvegarde futures.

Dans l'état actuel des spécifications, la tâche n'est pas sensée communiquer avec la zone utile, aussi le schéma 7.2. n'est que partiellement mis en place puisque la boite aux lettres concernant le clavier est provisoirement inhibée selon le schéma suivant :



file d'attente et inhibition de la case boite aux lettres

file des contextes.

b) Introduction de la tâche supplémentaire en régime stationnaire.

Elle consiste en un ordre de lecture du fichier de sauvegarde, secteur 0. Aucune information n'étant transmise à la tâche entrante l'opération se déroule selon le principe ci-cessus exposé.

IV.3. Sortie de la tâche contrôlant la mémoire.

IV.3.1. Sortie de la tâche de type APL.

Informations concernées	origine	fonction
:TOPOU	zone usager	Référence du temps.
OLTASK	"SWAP"	Etiquette opérationnelle de fichier de sauvegarde
KSWOU		Indicateur sortie en cas d'erreur d'E/S sur le disque
LTASK		Adresse de la zone utile de la tâche.

Après avoir recopié la SRD dans la zone brouillon, un premier ordre d'entrée-sortie recopie sur disque, la zone PRT + CDS puis le deuxième ordre concerne l'espace de travail.

IV.3.2. Sortie de la tâche supplémentaire.

La sortie consiste en un ordre d'écriture sur le fichier de sauvegarde, secteur 0.

IV.4. Traitement des erreurs d'entrée-sortie.

Le traitement d'une erreur d'E/S sur fichier de sauvegarde conduit à tuer le programme de l'usager pour lequel l'incident à lieu. Les critères KSWIN ou KSWOU permette d'aiguiller les traitements. Il y a émission d'un message sur le terminal de la tâche puis exécution du meurtre de la tâche par appel à la séquence de fin, il est à remarquer que si la tâche utilise à ce moment précis des sémaphores le mécanisme inter-tâches peut en être perturbé sans que le système puisse y remédier.

- NUMAC contient le numéro de tâche par V:ELU (KSWIN) ou V:TASK (KSWOU) et sert à positionner NUMAC.

- PORDON est sauvegardé puis forcé à l'adresse de retour en traitement d'erreur.

- LTACHE est forcé à la valeur de la zone utile de la tâche en erreur et sert à positionner la base L avant l'appel à la séquence de clôture, la base locale du module SWAP étant sauvegardée dans un mot d'accès général SLSWAP.

Ceci répond aux spécifications nécessaire au traitement correct de la fin de tâche comme si celui-ci avait été déclenché par l'ordre approprié.

Les retour s'effectuent via PORDON, la base L est de nouveau restituée ainsi que PORDON. Le fichier de sauvegarde est rendu par FERMER.

IV.5. L'affectation d'un quantum de temps.

a) En fin de traitement le quantum de temps est lancé, le débordement de temps DEBT et V:SWAP sont mis à zéro, les variables internes au "module" "SWAP" sont positionnées :

V:ELU	est mis dans V:TASK (numéro de fiche en fonction).
OLELU	est mis dans OLTASK (pointeur à la zone utile de l'usager
BASEGO	est rempli par l'adresse de la CDS du niveau "0" relative au niveau superviseur, le contexte de la tâche installée en mémoire est connecté. Le retour dans la boucle de scrutation est assuré par P:END2.

b) Pour la tâche supplémentaire, considérée comme une tâche de fond, il n'y a pas lieu de lancer le délai et DEBT n'est pas remis à "zéro". Les variables internes sont traitées de la même manière qu'en a).

SYNCHRONISATION DES TACHES

I - GENERALITES.

Les diverses tâches présentes dans le système n'évoluent pas forcément indépendamment les unes des autres.

Il peut y avoir un instant donné utilisation d'une ressource critique durant une phase d'exécution (section critique). La ressource doit être rendue inaccessible aux autres tâches car le système en temps partagé introduit une certaine simultanéité. Cette notion apparaît lors de l'utilisation de bibliothèques ou de fichiers de données. Le système d'exploitation standard SCF 15 des fichiers interdisant le mécanisme d'accès simultané en écriture, l'emploi en temps partagé a nécessité une modification du moniteur pour s'affranchir de cette contrainte. En effet, au niveau de l'ouverture un fichier ouvert assigné en mode

- IN est accessible simultanément par plusieurs étiquettes opérationnelles
- OUTet RW (écriture et mise à jour), n'est accessible que par une seule étiquette opérationnelle.

Ces cas sont résumés dans le tableau suivant :

Ouverture en mode de l'assignation	ouvert en IN	ouvert en RW/OUT	fermé libre	fermé en IN	fermé en OUT
IN	OUI		OUI	OUI	
OUT/RW			OUI		OUI

Dès lors, l'accès au fichier est règlementé par un mécanisme approprié au niveau du système en temps partagé qui met en jeu des sémaphores manipulés par des primitives intégrées au langage APL. Ceci permet aux tâches actives d'organiser une synchronisation, une tâche active pouvant bloquer ou activer une autre tâche.

II - DEFINITION DU SEMAPHORE.

Un sémaphore est constitué par une variable entière e(s) (positive, nulle ou négative) et une file d'attente f(s). Le sémaphore est créé par une déclaration spécifiant la valeur initiale e_0 (s) de e(s) nécessairement entière non négative, la file d'attente est vide. Deux primitives P et V, travaillant sur le nom du sémaphore donné en argument, positionnent sa valeur.

P (sem) diminue de 1 la valeur de e(s)

- si $e(s) \geqslant 0$ l'action de la primitive s'arrête là.
- si $e(s) < 0$ la tâche qui exécute la primitive se bloque et rejoint la file d'attente du sémaphore.

V (sem) augmente de 1 la valeur de e(s)

- si $e(s) \leqslant 0$ une tâche est sortie de la file d'attente du sémaphore et introduite dans la file d'attente à l'unité centrale pour être rendue active par l'allocateur.
- si $e(s) > 0$ l'action de la primitive s'arrête là.

A tout instant on a la relation :

$$e(s) = e_0(s) - np(s) + nv(s).$$

np (s) = nombre d'instruction P exécutées sur le sémaphore s

nv (s) = nombre d'instruction V exécutées sur le sémaphore s.

a) si e(s) est négatif, sa valeur absolue représente le nombre de tâches bloquées dans la file.

b) si e(s) est positif ou nul, sa valeur donne le nombre de tâches pouvant franchir le sémaphore sans être arrêtées.

Le sémaphore est introduit dans ce système sous deux conceptions :

1) Une barrière d'accès à une ressource pour limiter son emploi lorsqu'il est initialisé à une valeur positive. En particulier si $e_0(s) = 1$, en régime stationnaire la valeur du sémaphore est :

a) 1 lorsque n tâches sont actives, aucune en phase critique.

b) - (n-1) lorsque n-1 tâches sont bloquées en attendant que la $n^{ième}$ tâche quitte la phase critique par l'application de la primitive V.

Ce type de sémaphore est désigné sous la qualification de sémaphore d'exclusion mutuelle.

2) Un compteur dont la valeur initiale est quelconque.

III - IMPLANTATION DES PRIMITIVES DE SYNCHRONISATIONS.

III.1. Conditions d'implantation.

a) L'accès aux variables d'états doit être protégé et règlementé.

Ces conditions sont implicitement réalisées par la conception même du système de temps partagé étant donné que la tâche possédant l'unité centrale est seule capable d'accéder au traitement des primitives.

Une primitive qui a pour conséquence la modification de la planification (par exemple blocage de la tâche sur une opération P) se comporte vis-à-vis du système comme une transaction de type lent.

b) Il faut éviter qu'un blocage mutuel puisse durer indéfiniment. Soit a et b deux sémaphores d'exclusion mutuelle, si deux tâches T1 et T2 exécutent respectivement (P(a) ; P(b)) et (P(b) ; P(a)), il y a risque d'interblocage *, chacune des deux tâches devant attendre que l'autre libère son sémaphore avant de continuer. Si l'unité centrale est allouée de telle manière que T1 fasse P(a) et T2 effectue P(b) alors T1 et T2 se bloquent dès l'opération P suivante. Il est difficile de prévoir de manière simple l'apparition de cette situation, une méthode serait en fait de déclarer préalablement les ressources utilisées. L'interblocage est résolu de façon abrupte par le fait qu'un opérateur a toujours le loisir d'abandonner l'exécution de son programme alors que celui-ci est en attente derrière son sémaphore.

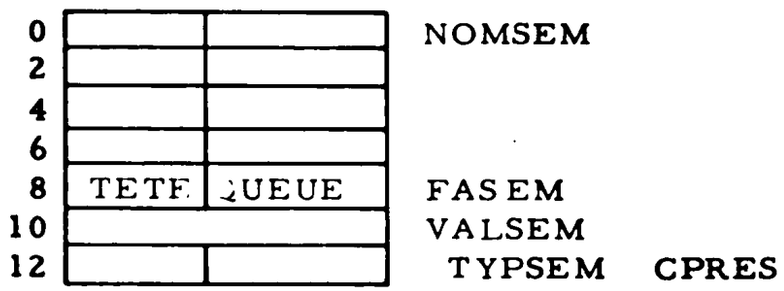
c) D'une façon générale il faut garantir l'indépendance effective des tâches en concurrence. C'est ainsi qu'une tâche qui dérouterait dans la section critique qu'elle a protégée gènerait les autres usagers en attente puisqu'elle ne serait plus en mesure d'effectuer l'opération V. Le système multi-console contrôle cette éventualité car les opérations P et V qu'une tâche effectue sur ses sémaphores sont notées. Si une tâche est tuée durant sa section critique alors qu'elle aurait positionné un sémaphore d'exclusion mutuelle, il faudra sortir de la file d'attente une éventuelle tâche en blocage et réajuster en conséquence la valeur de ce sémaphore en provoquant une opération V artificielle. S'il s'agit d'un sémaphore d'un autre type aucune précaution particulière n'est prise.

L'ensemble de ces mesures assure une protection efficace au niveau de l'usager contre les divers cas de blocage du système. Nous rappellerons néammoins que le système ne peut rien contre d'éventuelles fautes ou manques de programmation des sémaphores. L'abandon d'un programme APL est à tout le moins un remède dans les situations critiques.

* C'est l'étreinte fatale'

III. 2. Les bibliothèques de sémaphores et la communication.

III. 2. 1. La bibliothèque T:SEM.



Description d'une case de la bibliothèque.

La bibliothèque de sémaphores contient K:SEM cases définies à la génération dont la structure est :

NOMSEM : Nom du sémaphore codé sur 8 octets.

FASEM : Description de la file d'attente derrière le sémapho

VALSEM : Valeur entière positive ou négative du sémaphore.

TYPSEM : Type du sémapthore code sur un octet.

- 0 exclusion mutuelle
- Non nul extension.

CPRES : Compteur des accès au sémaphore.

Une case est libre lorsque les octets de NOMSEM sont nuls.

III. 2. 2. Catalogue privé d'une tâche.

Chaque tâche emporte dans son fichier de sauvegarde une zone où sont conservées les références faites aux sémaphores auxquels la tâche accède. C'est une table comportant TACSEM cases définies à la génération (TACSEM est le quota de sémaphores imparti à la tâche), pointée par A:ZONE en accès général.

Une case a la structure suivante :

Où :

CTRPTR contient le pointeur à la case de la table T:SEM du sémaphore auquel la tâche fait référence.

ETASEM contrôle les opérations P et V lorsque le sémaphore est d'exclusion mutuelle.

ETASEM est "pris" (valeur non nulle) sur une opération P passante sur le sémaphore.

ETASEM est "libre" (valeur nulle) si l'opération P sur le sémaphore est bloquante ou si une opération V est effectuée sur un sémaphore "pris". Autrement dit, l'état de la ressource constituée par le sémaphore est décrit par ETASEM.

Une case en catalogue privé est libre lorsque CTRPTR est nul.

bibliothèque de sémaphores.

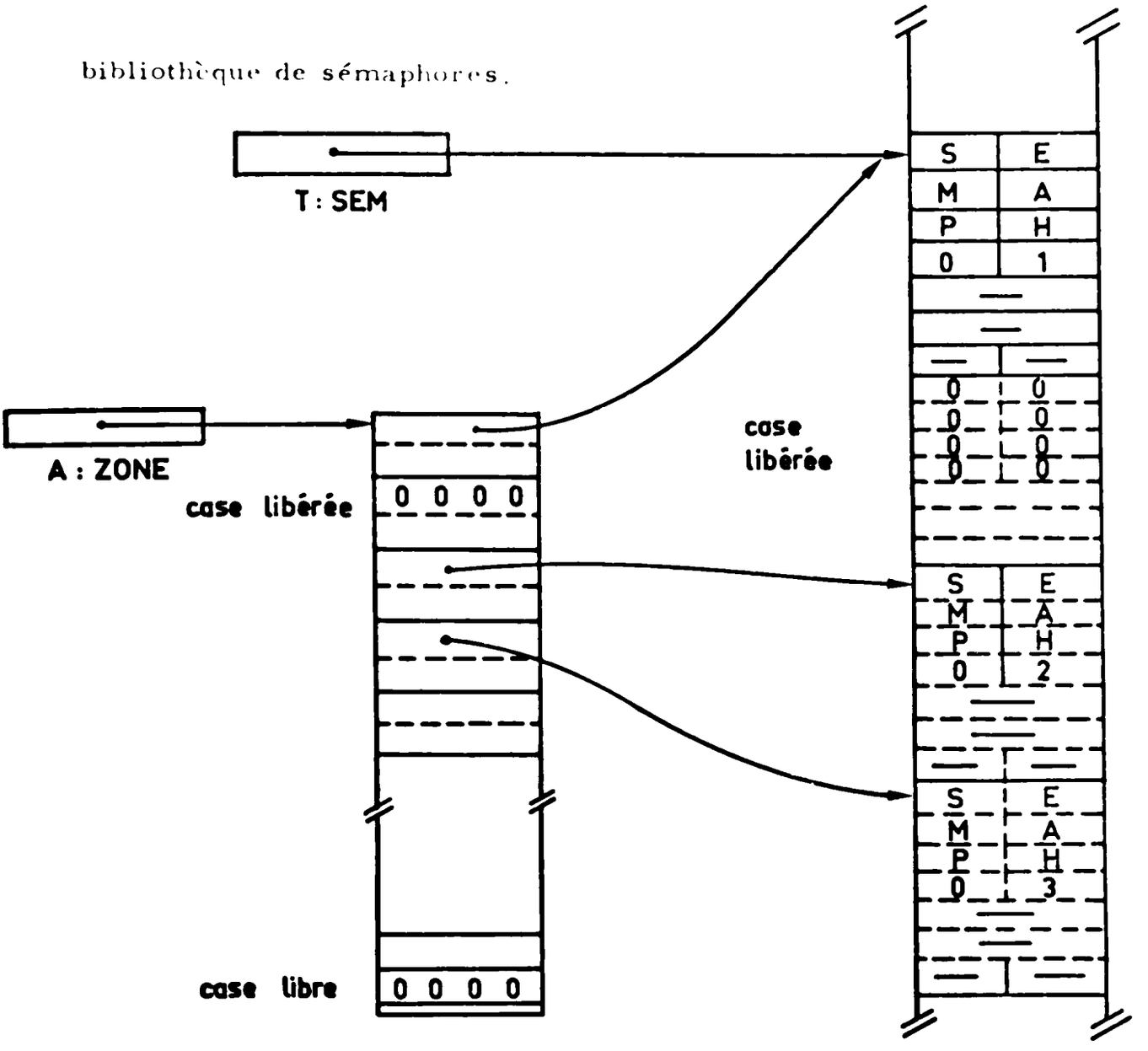


Fig. 8.1. Catalogue privé de la tâche en mémoire.

Les recherches sur la bibliothèque générale de sémaphores et sur le catalogue privé s'effectuent suivant le même principe, les initialisations étant différentes : les spécifications ci-après permettent de résoudre toutes les configurations de recherche pour manipuler les sémaphores.

entrée d'exploration.		
registre A	Registre E	signification.
pointeur relatif à G superviseur	indifférent	Pointeur du nom du sémaphore à rechercher dans la table T:SEM.
idem		pointeur du mot contenant un pointeur à rechercher dans le catalogue privé A:ZONE
sortie d'exploration		
Nul	négatif	La table explorée est complète.
différent de zéro		L'argument proposé n'a pas été trouvé dans la table explorée, la dernière adresse de case libre trouvée, relative à G superviseur, est rendue.
	nul.	l'argument proposé a été trouvé dans la table explorée, l'adresse relative à G de la case est rendue.

En sortie d'exploration , le registre X rend systématiquement le numéro de la case trouvée occupée ou libre dans la table.

III.2.3. Le principe de communication et la zone de communication.

Le principe de la communication est identique à celui rencontré pour les traitements des entrées-sorties sur terminal conversationnel. Le programme de traitement est déroulé par examen de la boîte aux lettres BALPRI de la tâche dans la section PDEMAN. La zone de communication est maintenant le tampon d'entrée-sortie sur le terminal car celui-ci ne va pas être utilisé durant ce type de transaction. Une description sommaire donne l'état des arguments pris en compte pour le traitement. La réponse à la transaction est aussi consignée dans cette zone.

position de l'information	Fonction
:INDK	Code les transactions sur sémaphore
:BUFFR 0	
2	Réservé à la gestion des arguments.
4	
6	
8	Pointeur relatif à G de l'APL de l'élément droit D
10	
12	Réservé à la gestion des arguments.
14	
16	Pointeur relatif à G de l'APL de l'élément gauche G
18	
20	nom du sémaphore
22	D
24	
-	
-	
-	
-	G valeur d'argument.
-	
-	
-	

Etat simplifié des arguments concernant les sémaphores.

III. 3. Le séquencement.

La commutation de tâche est obligatoirement positionnée, or les primitives concernant les sémaphores ne donnent lieu à commutation de

tâche que sous certaines conditions. Il suffira, lorsque celle-ci ne devra pas être provoquée a priori, de restituer l'ancien indicateur d'appel. Le séquencement a lieu sur la remise en file d'attente d'éligibilité à la mémoire des tâches jusqu'alors bloquées derrière des sémaphores et que l'action des primitives réactive.

La décision de séquencement est donc similaire à une fin d'entrée-sortie sur clavier, à savoir :

- Débordement de temps de la tâche actuellement en mémoire.
- Tâche actuellement en mémoire en état de perdre la mémoire, c'est à dire non gelée en mémoire et FSWAP non nul.

primitive	séquencement permis	Situation de P:CALL
ouverture	Non	P:CALL non modifié
V	Non	P:CALL non modifié
	Oui	P:CALL rendu négatif.
P	Non	P:CALL non modifié
	Oui	P:CALL rendu négatif
fermeture	Non	P:CALL non modifié
	Oui	P:CALL non modifié

IV - LES DIRECTIVES DE SYNCHRONISATION.

a) Les sémaphores ainsi définis sont directement utilisables par le processeur APL pour protéger des phases critiques de manipulation de tables du système. Par exemple, l'attribution d'une étiquette opérationnelle

de travail pour un fichier d'un usager se fait au sein d'une section critique comportant d'une part la recherche d'une étiquette libre, d'autre part la déclaration d'assignation de celle-ci au fichier. Ces opérations doivent être ininterruptibles d'un point de vue logique. Le processeur APL dispose donc de sémaphores réservés dont le nom est fixé une fois pour toute.

b) L'usager peut programmer ses sémaphores grâce aux directives incluses dans le langage APL. L'écriture des directives répond aux critères généraux de la sémantique du langage. Leurs spécifications sont les suivantes.

R étant la variable à laquelle est affecté le résultat de la primitive.

fonction de la primitive	type	primitive	signification des arguments
ouverture du sémaphore	dyadique	R←ARG □SS 1 ARD	ARC vecteur de caractères, nom du sémaphore. ARD vecteur flottant. - ARD 1 type du sémaphore - ARD 2 valeur initiale.
opération P sur un sémaphore	monadique	R←ARC □SS 2	ARC : vecteur de caractères nom du sémaphore
opération V sur un sémaphore	monadique	R←ARG □SS 3	ARC: matrice de caractères, noms de sémaphores.
fermeture d'un sémaphore	monadique	R←ARG □SS 4	ARC : matrice de caractères, noms de sémaphores. Si vide, libérer tous les sémaphores de la tâche.

IV.1. Déclaration d'un sémaphore.

Le programme de recherche permet de trouver une case libre dans la bibliothèque des sémaphores, ou lorsque le sémaphore est connu, de retrouver cette case.

a) Le sémaphore est initialisé.

La case en table T:SEM est remplie selon les directives implantées dans la zone tampon.
FASEM est rempli de façon à décrire une file vide.
VALSEM est rempli à la valeur initiale demandée par l'opérateur.
NOMSEM est rempli avec les caractères du nom.
TYPSEM est rempli avec le type de sémaphore prévu.
CPRES est initialisé à 1.

Une case est aménagée dans le catalogue privé de la tâche en mémoire et CTRPTR est rempli avec le pointeur de la case du sémaphore en bibliothèque générale. ETASEM est remis à zéro

b) Le sémaphore est déjà déclaré.

Le contrôle porte sur le type du sémaphore réclamé par la tâche en mémoire par coïncidence avec la valeur contenue en TYPSEM de la case détectée en bibliothèque générale. Il y a incrémentation d'une unité du compteur de réservation et l'aménagement d'un entrée dans le catalogue privé. (Procédure a)).

IV.2. Primitive P

Le programme de recherche localise les cases du sémaphore dans la bibliothèque générale et dans le catalogue privé de la tâche en mémoire.

IV.1. Déclaration d'un sémaphore.

Le programme de recherche permet de trouver une case libre dans la bibliothèque des sémaphores, ou lorsque le sémaphore est connu, de retrouver cette case.

a) Le sémaphore est initialisé.

La case en table T:SEM est remplie selon les directives implantées dans la zone tampon.
FASEM est rempli de façon à décrire une file vide.
VALSEM est rempli à la valeur initiale demandée par l'opérateur.
NOMSEM est rempli avec les caractères du nom.
TYPSEM est rempli avec le type de sémaphore prévu.
CPRES est initialisé à 1.

Une case est aménagée dans le catalogue privé de la tâche en mémoire et CTRPTR est rempli avec le pointeur de la case du sémaphore en bibliothèque générale. ETASEM est remis à zéro

b) Le sémaphore est déjà déclaré.

Le contrôle porte sur le type du sémaphore réclamé par la tâche en mémoire par coïncidence avec la valeur contenue en TYPSEM de la case détectée en bibliothèque générale. Il y a incrémentation d'une unité du compteur de réservation et l'aménagement d'un entrée dans le catalogue privé. (Procédure a)).

IV.2. Primitive P

Le programme de recherche localise les cases du sémaphore dans la bibliothèque générale et dans le catalogue privé de la tâche en mémoire.

La valeur du sémaphore VALSEM est diminuée d'une unité. Si la valeur devient négative, le numéro de la tâche rejoint la file d'attente du sémaphore dont l'adresse relative à G superviseur est fournie au programme INEL. La tâche étant préparée à perdre la mémoire, le pointeur d'arrêt est conservé dans la zone usager.

Le traitement se terminant par P:END2, l'indicateur de fin de transaction BALPRI n'est pas positionné. Si la tâche n'est pas changée faute de prétendant à l'unité centrale à ce moment précis, sa progression est arrêtée en débordement de temps ; le niveau 0 bouclant sur le signal d'arrivée de la fin de la transaction. Le pointeur de la case en table T:SEM est déposée dans la zone de l'usager. Dans la case de la table A:ZONE, le mot ETASEM n'indique pas la prise de la ressource sémaphore. La tâche part en sommeil dans la situation exposée. Lorsque la tâche sera réactivée par une primitive V sur le sémaphore correspondant, le processeur de commutation des tâche remarque que la tâche était jusqu'alors en sommeil par blocage derrière un sémaphore. Celle-ci est donc relancée après lui avoir attribué la ressource du sémaphore, c'est à dire, après avoir positionné le mot ETASEM à l'état "pris". Le processeur remet à jour l'information de blocage.

Si l'action de la primitive est passante le contrôle est rendu à l'usager. Le mot ETASEM de la case pointée en catalogue privé est positionné si le sémaphore est d'exclusion mutuelle. La ressource constituée par le sémaphore est donc prise.

IV.3. Primitive V

L'opération est l'opposée de l'opération P décrite ci-cessus. Elle n'est permise pour un sémaphore d'exclusion mutuelle qu'à la condition que celui-ci ait été positionné (ou pris) par une opération P au préalable.

- La valeur du sémaphore est augmentée d'une unité en VALSEM. et l'opération est consignée par la mise à jour du mot ETASEM. Pour l'exclusion mutuelle.

- Si la primitive est passante, le contrôle est entièrement rendu à la tâche demandeuse, code fin de transaction positionné dans la boite aux lettres. En revanche, tant que la valeur du sémaphore est inférieure ou égale à 0, il faut prélever la première tâche de la file d'attente du sémaphore et la libération de la tâche sélectionnée se fait en imposant artificiellement les opérations suivantes communes à toute fin de requêtes commutables.

- Code fin de transaction dans la boite aux lettres BALPRI de la tâche (code &8000).

- Diminution d'une unité du compteur de requête commutable. Par ailleurs, la tâche est réintroduite en file d'attente à la ressource Unité Centrale, si son compteur FSWAP a été annulé par la dernière opération et si elle n'est pas gelée en mémoire.

IV.4. Fermeture d'un sémaphore.

L'action principale qui en découle est la décrémentation d'une unité du compteur de réservation CPRES de la case trouvée après la recherche en table.

a) CPRES garde une valeur positive.

Il y a effacement de la référence au sémaphore dans le catalogue privé de la tâche. Si le sémaphore est d'exclusion mutuelle et qu'une opération P ait été précédemment lancée, il y a libération "forcée" d'une tâche bloquée derrière le sémaphore si sa valeur actuelle est par ailleurs négative ou nulle (nous sommes ramenés au cas IV.3)

b) CPRES est annulé.

Il y a effacement de la référence au nom du sémaphore dans le catalogue privé de la tâche et dans la bibliothèque générale de sémaphores.

La manipulation des sémaphores demande des contrôles particulièrement élaborés dont voici les spécifications.

primitive	réponse et significations	
déclaration	R 1 = 0	déclaration effectuée
	R 1 0	sémaphore déjà déclaré
	R 1 = 1	ce sémaphore existe avec un autre type.
	R 1 = 2	ce sémaphore existe avec une autre valeur initiale
	R 1 = 3	quota particulier de sémaphores épuisé
	R 1 = 4	quota général de sémaphores épuisé
	R 2	type du sémaphore
	R 3	valeur initiale du compteur
	R 4	valeur courante du compteur
	R 5	nombre d'utilisateurs de ce sémaphore avant appel
	R 6	numéro du sémaphore 0.
P	R = 0	l'opération est effectuée
	R = -n	le sémaphore est d'exclusion mutuelle et vous avez déjà pris les n ressources associées, l'opération est transparente.
	R = 1	vous n'avez pas déclaré ce sémaphore.
V	R = 0	l'opération est effectuée
	R = 1	vous n'avez pas déclaré ce sémaphore.
	R = 2	le sémaphore est d'exclusion mutuelle et vous n'avez pas la ressource associée.
fermeture	R i;1 = 0	le sémaphore correspondant est libéré
	R i;1 = 1	vous n'avez pas déclaré ce sémaphore
	R i;1 = -n	le sémaphore est d'exclusion mutuelle et le système a fait "n" opération "V" sur ce sémaphore avant la libération
	R i;2	nombre d'usagers restant du sémaphore correspondant
	R i;3	numéro du sémaphore correspondant.

V - LA GESTION DE L'ABANDON.

Deux appuis sur la touche "ESC" lors de l'arrêt d'une tâche derrière un sémaphore provoquent l'abandon de la requête au même titre que le double "ESC" provoque l'arrêt immédiat du calcul en APL. L'usager est considéré en cours de "calcul" de la primitive P qui l'a bloqué. Le système effectue donc un "V" sur le sémaphore et retire le numéro de tâche de la file d'attente du sémaphore. Le pointeur de la case du sémaphore en table T:SEM étant disponible en CEXIT de la zone utile de l'usager, il est possible de retrouver la file d'attente et de sortir le numéro par OUEL. Les opérations qui suivent s'apparentent à une fin de requête commutable. La base L pointe alors sur la bonne zone de l'usager.

- L'indicateur de transaction BALPRI est positionné à fin de transaction (&8000).
- Le compteur d'attente de transaction commutable FSWAP est diminué d'une unité. La remise en file d'attente à la mémoire centrale est faite si la tâche nest plus en mémoire et FSWAP étant nul.

Le mot CEXIT est remis à zéro que la tâche abandonnée soit ou ne soit pas en mémoire. Il n'y a pas lieu de s'occuper de la case du catalogue privé étant donné que la prise de la ressource sémaphore (exclusion mutuelle) n'est pas effective.

Le traitement de l'abandon se termine par la relance de la reconnaissance du caractère d'abandon.

La situation présente est fondamentalement différente d'un abandon provoqué au sein d'une section critique protégée par une primitive P (sem) : l'usager garde dans ce cas la prise de la ressource sémaphore.

CONCLUSION

Le système en temps partagé que nous avons décrit donne entière satisfaction sur le site d'exploitation du service d'électronique de Saclay où la réalisation a été développée, et sur les diverses installations où il a été implanté.

La réalisation d'un système en temps partagé sur mini ordinateur, s'articule autour de quelques idées "philosophiques" simples, comme nous avons pu le prouver. Le logiciel qui en découle n'est certes pas aussi puissant que celui d'un gros ordinateur mais du moins a-t-il le mérite d'être taillé sur mesure pour supporter le processeur APL, et par la même d'utiliser au maximum ses possibilités. Il est de plus relativement fiable et aisé à maintenir par suite de la simplicité de sa conception. Il faut rappeler que le superviseur du temps partagé fonctionne sous système MTRD.E. ce qui a imposé une contrainte supplémentaire à la réalisation.

Dans l'application actuelle c'est un produit qui peut évoluer, sa conception permettant des extensions éventuelles ainsi que le témoigne le mécanisme de synchronisation des tâches mis en place.

Certaines optimisations du système en temps partagé semblent être souhaitables : citons par exemple la réduction du temps de commutation des tâches.

Quelles que soient les améliorations et extensions pouvant être introduites il ne faut pas oublier que l'accroissement du code en résultant est au détriment des performances du processeur APL 15. Ce sont en définitive les performances de ce processeur qui déterminent la qualité du système présenté.

GLOSSAIRE

A	**Registre Accumulateur.**
C	Indicateur programme "CARRY" à usage multiple.
CDS	Segment de Données Communes.
CLS	Instruction d'appel d'une Section Programme.
CPT	Table des adresses des contextes programme.
CTX	Table de protection du contexte programme.
CSV	Instruction d'appel d'une section superviseur.
DIT	Instruction de désactivation de l'interruption. (avec rangement du contexte programme)
DVT	Table des mots de désactivation des interruptions.
E	Registre d'Extension de l'accumulateur.
E/S	Entrée/Sortie périphérique.
FIFO	Pile du type FIRST-IN/FIRST-OUT.
G	Base générale.
IDS	Segment de données Indirectes.
IMT	Image Mémoire Translatable. Code écécutable résultant d'une édition de liens.
IT	Interruption.
L	Registre de base locale.
LDS	Segment de Données Locales.
LPS	Segment de programme.
MA	Indicateur programme de Masque des interruptions.
MS	Indicateur programme du mode Maître/Esclave.
O	Indicateur programme "OVERFLOW" à usage multiple.
P	Compteur ordinal - registre d'adresse programme.
PM	Indicateur programme de protection Mémoire.
PR	Violation de protection en mémoire Centrale.
PRT	"Program Relocation Table". Table de double-mot d'affectation.
PRTS	Table de double-mots d'affectation des sections moniteurs.

R8	Registre Rapide 8 donnant 2 fois le niveau d'interruption en cours.
RD	Instruction de lecture directe périphérique.
RSV	Instruction de retour Superviseur.
RTS	Instruction de Retour Section.
SRD	"Section Relocation Double-Word". Double-mot d'affectation associé à un segment de programme.
TWB	"Task Working Block". Bloc de travail pour les modules moniteurs.
UC	Unité Centrale.
UE	Unité d'Echange.
UT	Unité de Traitement.
VM	Violation de Mode.
WD	Instruction d'écriture directe périphérique.
X	Registre programme d'Index.

ANNEXE A

I - GENERATION DU SYSTEME.

Le processeur spécialisé CSYSS fabrique à partir de commandes constituées par l'usager un moniteur d'exploitation autour d'un noyau sous forme d'IMT disponible en bibliothèque EP.

L'apport et la modification de modules moniteurs ayant necessité la constitution d'un tel noyau par édition de liens des BT des modules standard, modifiés ou spéciaux, CSYSS réalise :

- L'incorporation ou le remplacement de section programme LPS disponible en IMT.
- L'insertion de tables.
- La définition de références et de modifications.
- La définition de "Handlers" à partir de modules IMT.
- La définition de tâches immédiates, l'élément de DVT, le contexte de lancement étant associés. Un processeur de liste automatique peut être implanté en utilisant les propriétés de rétablissement de la tension lors d'une coupure secteur.

En effet le programme de retour secteur fait appel à la section M:INIT qui initialise le système. Des séquences particulières peuvent être incluses dans cette section telles que armement, validation, excitation de niveaux d'IT avant désactivation. Une tâche immédiate simplifiée a été introduite au niveau du moniteur d'exploitation sur le niveau réservé au système en temps-partagé. A l'initialisation du système d'exploitation, cette tâche est activée sans aucune conséquence. Lorsque le chargeur installe le superviseur multi-console il substitue dans la tableCPT le pointeur de contexte du superviseur à celui de la tâche immédiate. En provoquant la coupure et le retour secteur,

le niveau du système multi-console est de nouveau réactivé, ce qui peut être mis à profit pour lancer une liste.

II - GENERATION DES TABLES SYSTEME.

Le processeur GENTES crée toutes les tables des systèmes d'exploitation à partir des commandes décrivant le matériel et le logiciel du système à générer.

Les tables délivrées sont :

a) Les tables d'entrées-sorties décrivant l'équipement et le logiciel y afférant. Nous y remarquons notamment la description des étiquettes opérationnelles pour mémoriser les assignations de celles-ci ainsi que leurs utilisateurs. La conception du système réclame notamment l'abandon de la notion de "back-ground" et de "foreground" au profit de cette dernière, "OLTB", "OLTF" tables confondue. Les tables d'entrées-sorties ont été ainsi modifiées et l'option de dédoublement des étiquettes opérationnelles supprimé dans les modules superviseur concernés par la fonction d'Entrée-Sortie.

b) Les tables pour la gestion de fichier SGF15E.
Ces tables gèrent les fichiers. A ce titre, la table parallèle à la table "OLFI" permet plus particulièrement de déterminer les caractéristiques des fichiers auquel l'OL est affectée. Ceci permet de prendre en compte, au niveau du système en temps partagé les séquences de traitement d'entrées-sorties sur fichier.

c) D'autres tables dont les caractéristiques respectent le profil d'une installation standard et à ce titre n'ont pas été modifiées dans le cadre de cette étude.

```
UC
TY,PR,PP,CR,LP,DC,AL;  Définition des périphériques accrochés à l'U.C.
OL
DC,DA,UL:15;    définition des tables d'étiquettes opérationnelles.
M:BI,T:CR;      entrée binaire sur lecteur de cartes.
M:BO,T:PP;      sortie binaire sur perforateur de ruban.
M:EO,T:PP,EN;   sortie d'éléments sur perforateur de ruban.
M:CI,T:CR,AN;   entrée de commandes sur lecteur de carte.
M:OC,T:TY;      organe de commande sur télétype.
M:EI,T:CR,EN;   entrée d'éléments sur lecteur de carte.
M:LO,T:LP;      sortie de liste sur imprimante.
M:LL,T:LP;      journal de bord sur imprimante.
M:DO,T:LP;      sortie de diagnostic sur imprimante.
M:SI,T:CR,AN;   entrée source sur lecteur de carte.
M:SY,T:DC;      disque système.
%EOD            fin d'entrée des commandes.
```

Liste des commandes pour la définition des tables d'entrée-sortie du moniteur - (GENTES).

```
%JOB/7161    GENERATION DU MTRD-E SPECIAL POUR MULTI-CONSOLE
%C/GSYSS/GO,(APLI.01)                    *nom moniteur et numéro de version.
SV/(APLI),EP                             *noyau moniteur sous forme d'IT
MO/SO1,@&64:&754,@&68:&754               *TABLES OLTB ET OLTF
MO/SO1,@&3AE:&1100                       *AFFECTATION UGF15 D:0,D:4
MO/SO58,P,@&1F8:&F408,&2121,&2E02,&2D04,&F403   *M:INIT ACTIVE TK PP4
DR/(TRAPX2),LP                           *traitement des déroutements
RF/@&0028:L+0000                         *
TK/(COUSEC),EP,N31                       *coupure secteur, niveau d'IT 31.
TK/(RETSEC),EP,N30                       *retour secteur, niveau d'IT 30.
TK/(HORL10),EP,3:9                       *tâche immédiate horloge.
TK/(LANCE7),EP,N5                        *tâche immédiate "bootstrop" muli-console.
MO/@&48:&006E
MO/P,@&12:&C70A,@&54:&6008,@&76:&F701
IN/(HTY062),EP,TY                        *handler ASR 33.
IN/(HFR002),EP,FR                        *handler lecteur de ruban.
IN/(HFP002),EP,FP                        *handler perforateur de ruban.
IN/(HCR002),EP,CR                        *handler lecteur de cartes.
IN/(HLP030),EP,2:6                       *handler logabax.
IN/(HDM004),EP,DM                        *handler disque.
IN/(HAL030),EP,AL                        *handler lignes asynchrones.
ZC/810                                   *longueur de la zone commune.
%EOD                                     *fin des commandes sur cartes.
```

Liste des commandes pour la génération du moniteur spécial supportant le système multi-console

ANNEXE B

ORGANI CRAMMES

S1 Boucle de scrutation

S2 Traitement des événements lents dans la boucle de scrutation.

S3 Fins de traitement d'entrée-sortie sur terminal conversationnel.

S4 Sections Q :IOWA et Q:WAIT.

S5 Algorithme de la section SWAP.

S6 Suite de l'algorithme de la section SWAP.

S7 Algorithmes d'assignation, d'ouverture et de fermeture d'une étiquette opérationnelle.

S8 Algorithme du lancement d'une entrée-sortie.

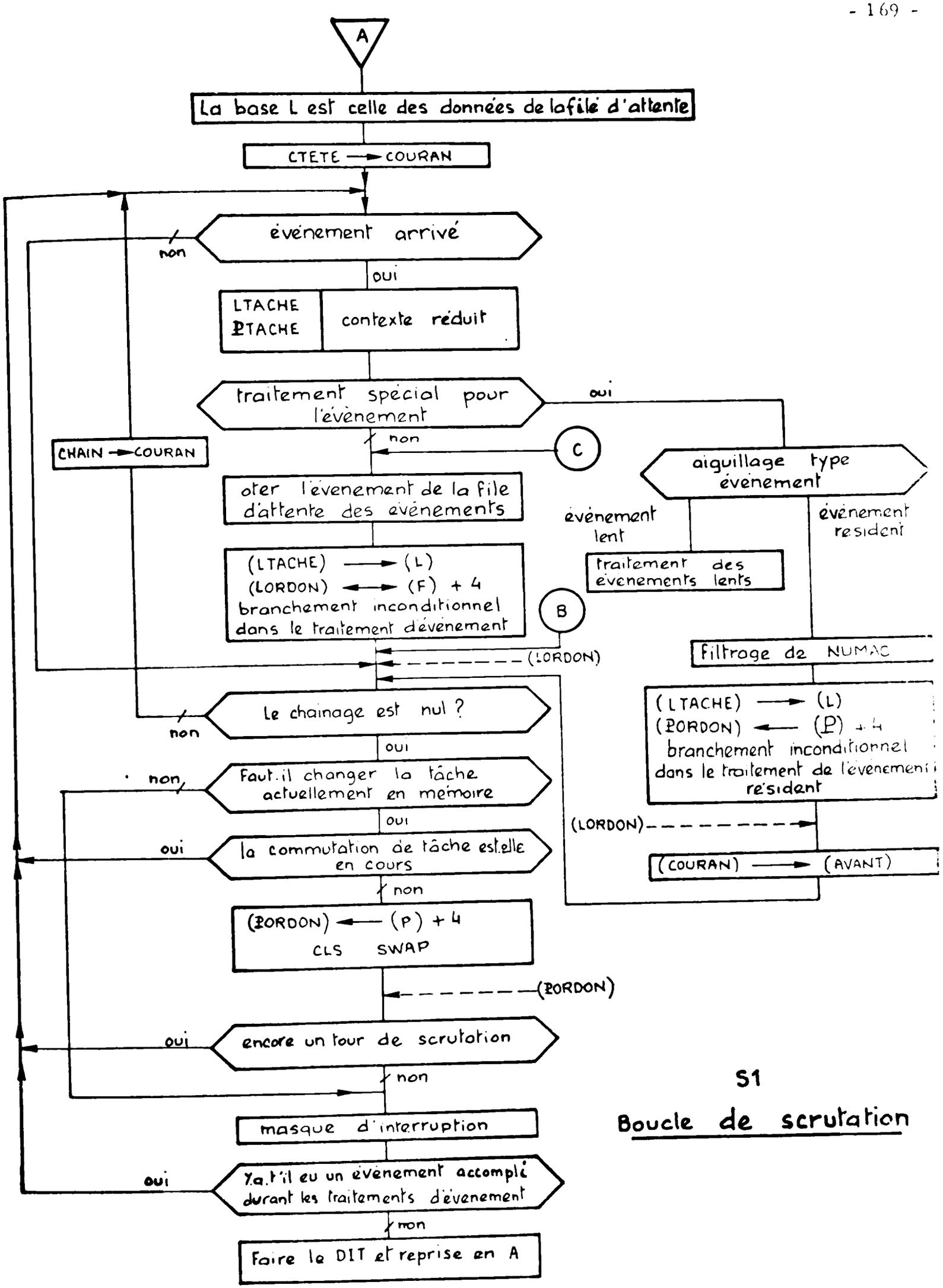
S9 Attente d'une entrée-sortie.

S10 Traitement de la primitive de déclaration de sémaphore.

S11 Traitement de la primitive V sur un sémaphore.

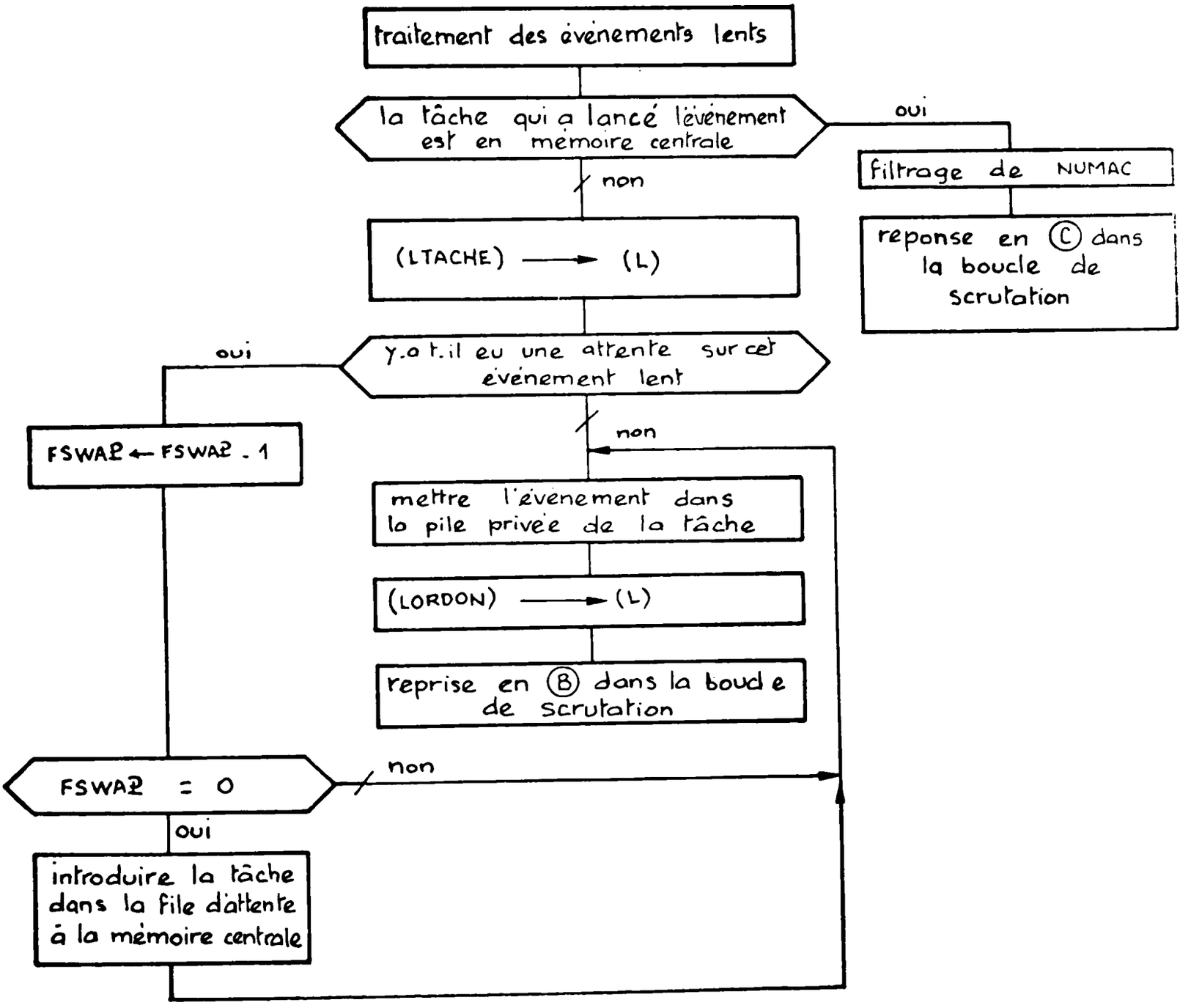
S12 Traitement de la primitive P sur un sémaphore.

S13 Traitement de la primitive de fermeture d'un sémaphore.



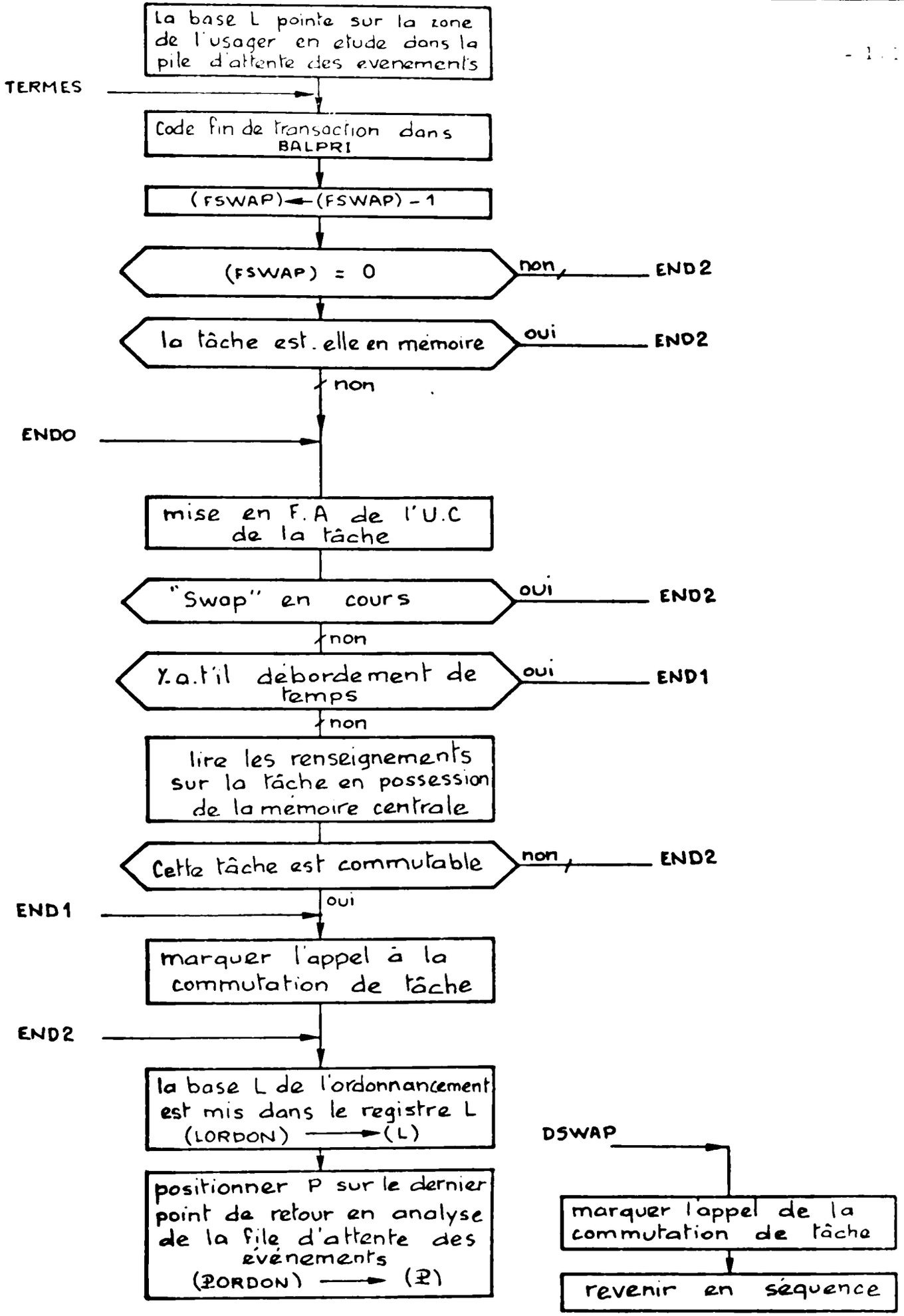
S1

Boucle de scrutation



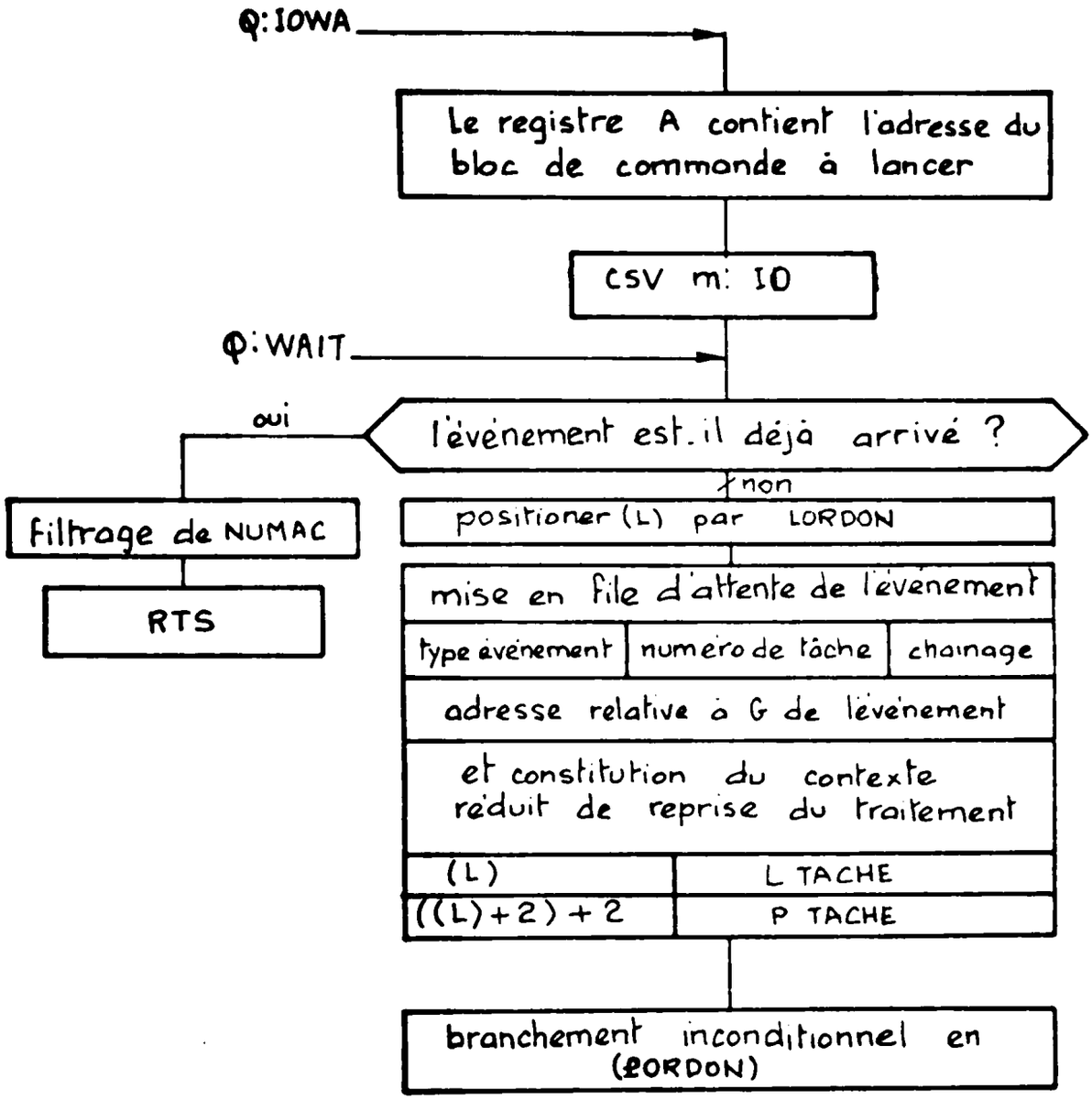
S2

Traitement des événements de type lent dans la boucle de scrutation



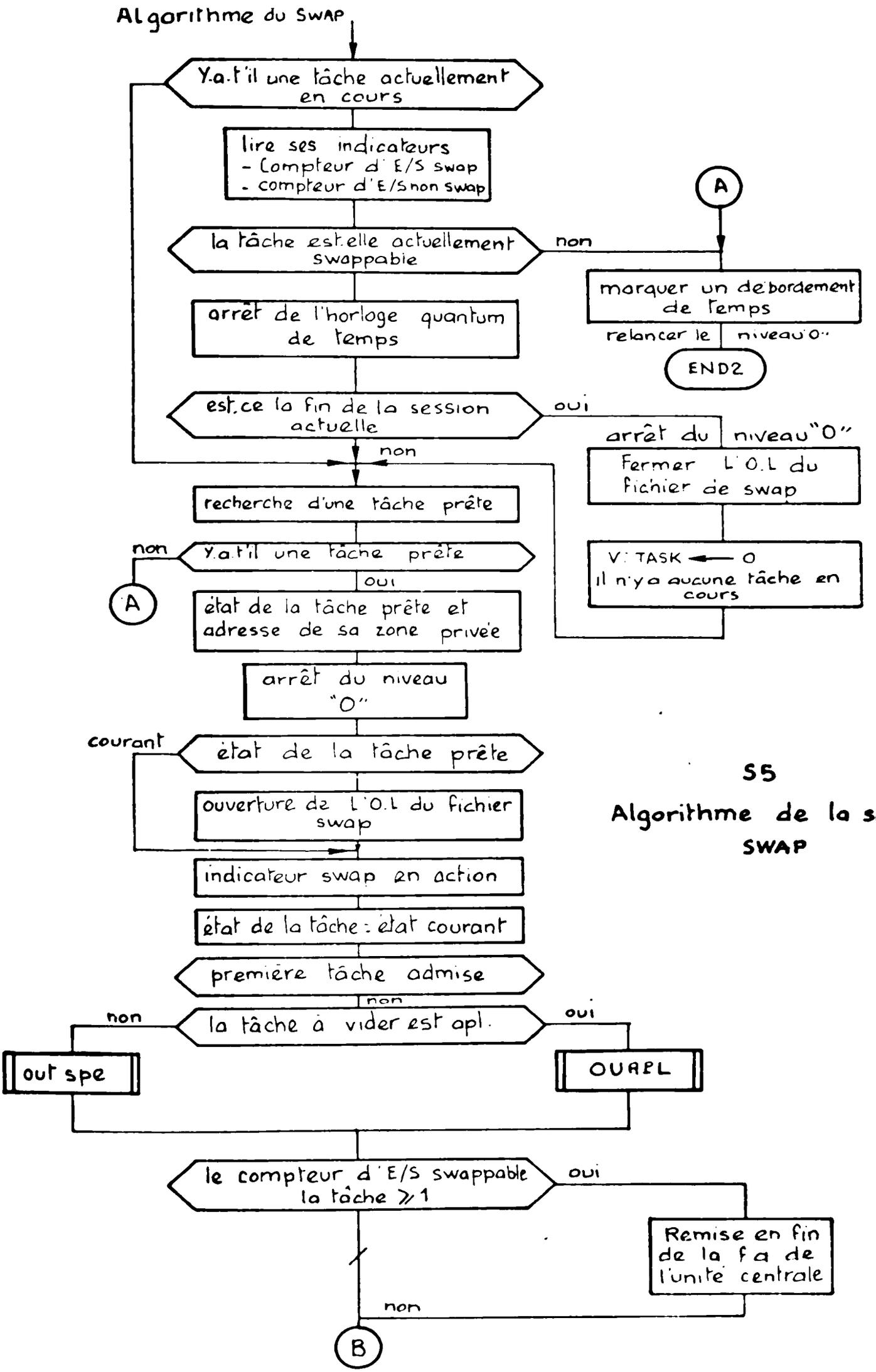
53

Fins de traitement d'entrée. sortie sur terminal conversationnel



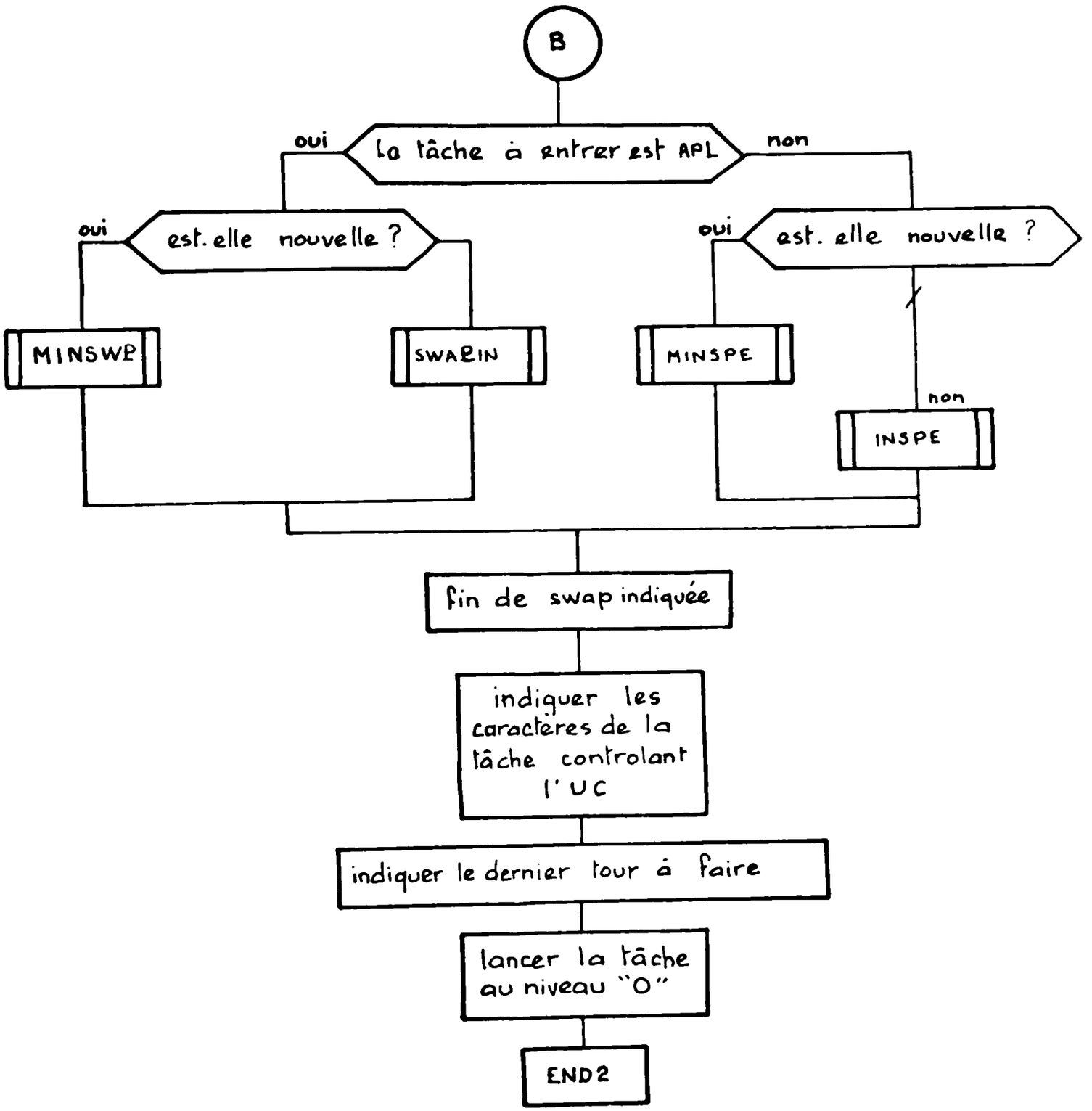
S4

Sections Q : IOWA et Q : WAIT



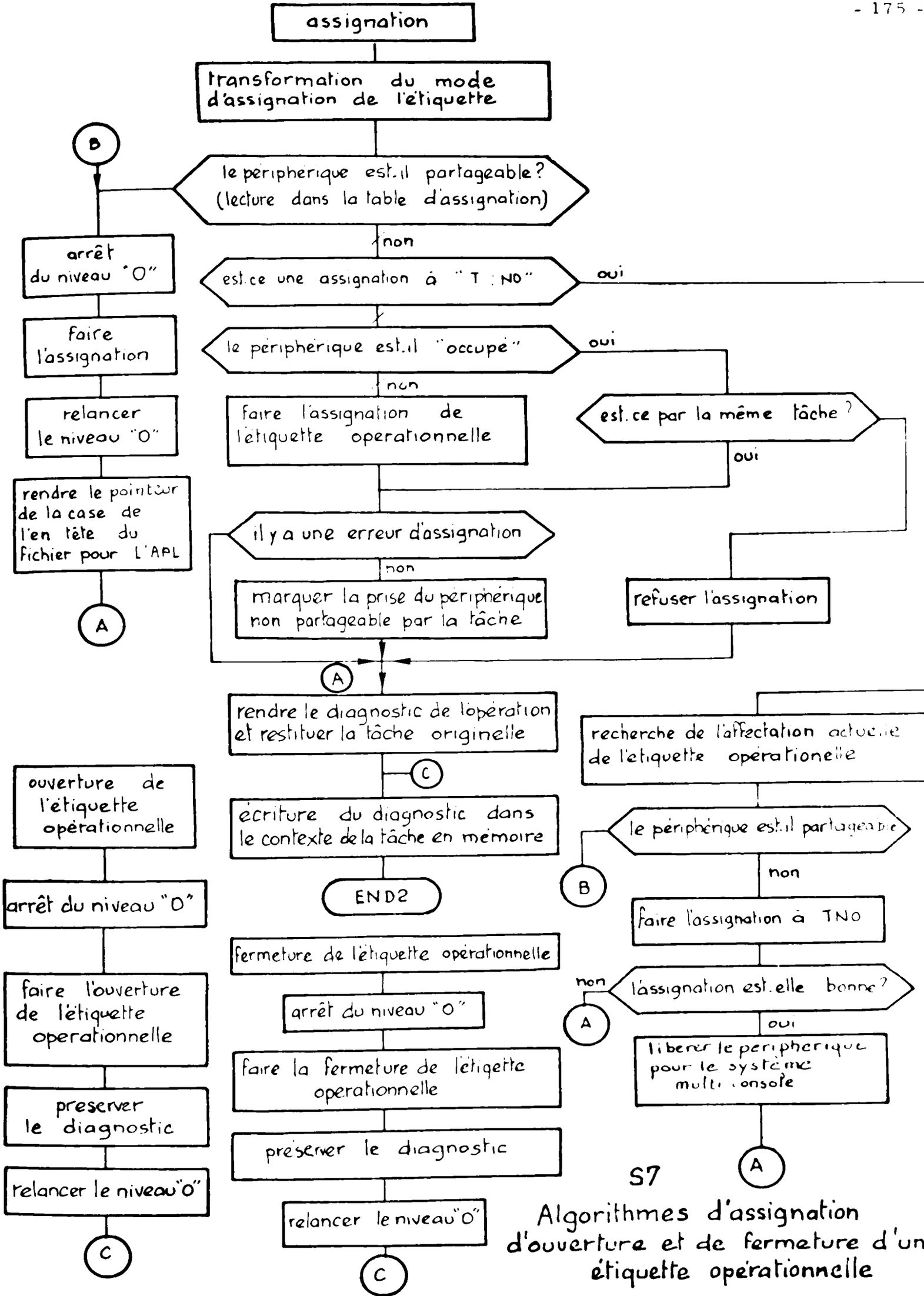
S5

Algorithme de la section SWAP



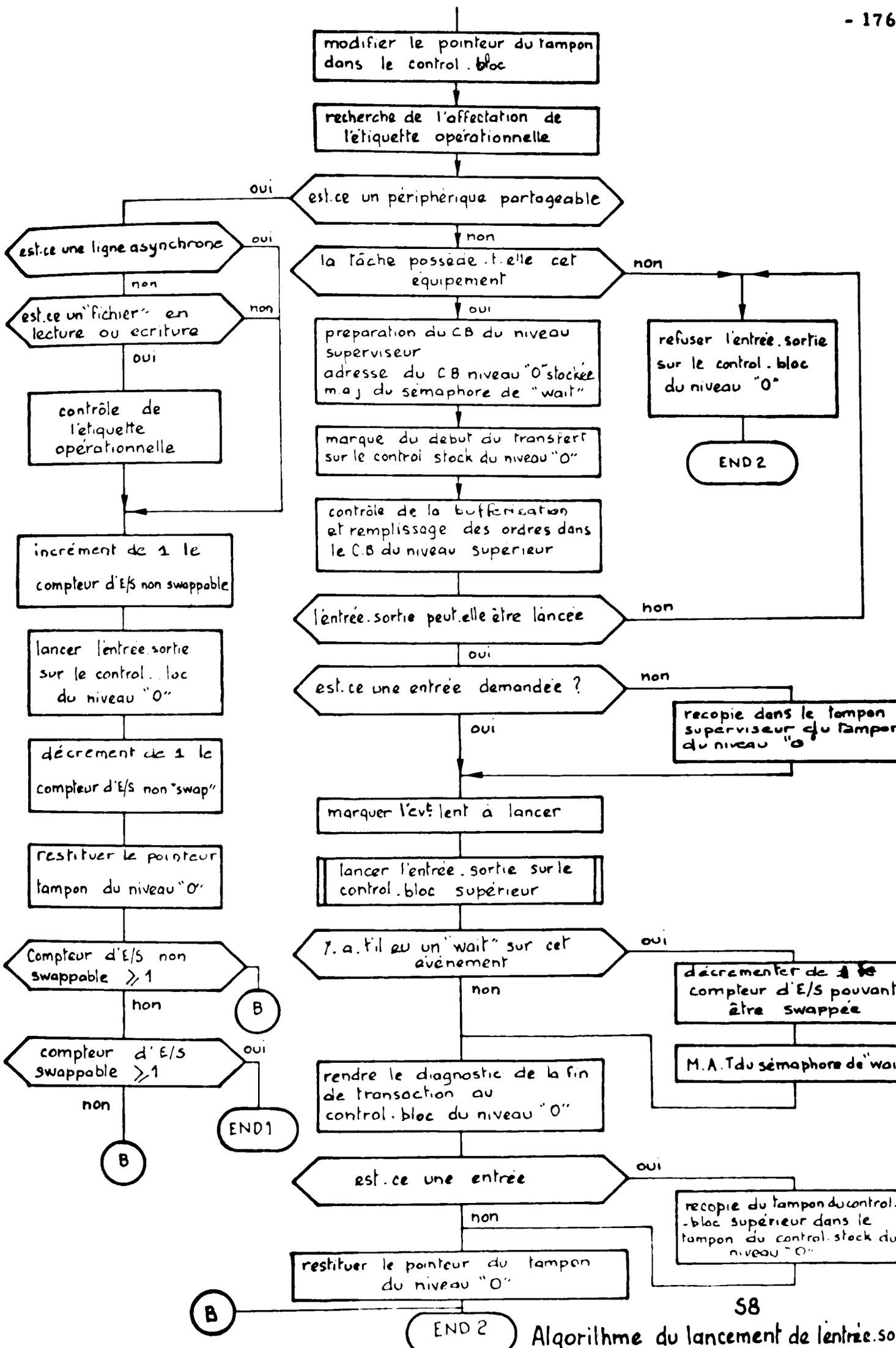
S6

Suite de l'algorithme de la section SWAP



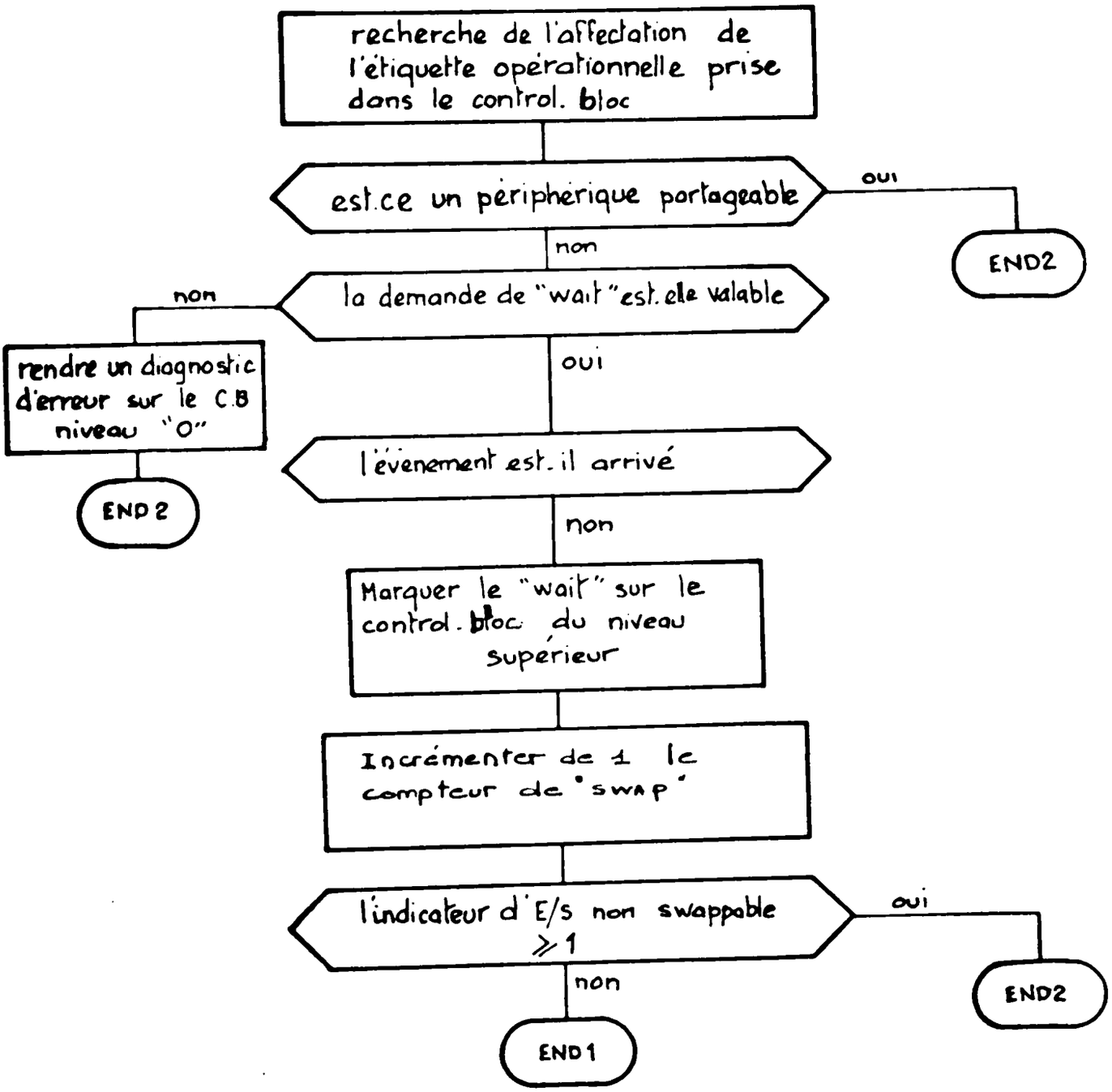
S7

Algorithmes d'assignation d'ouverture et de fermeture d'un étiquette opérationnelle



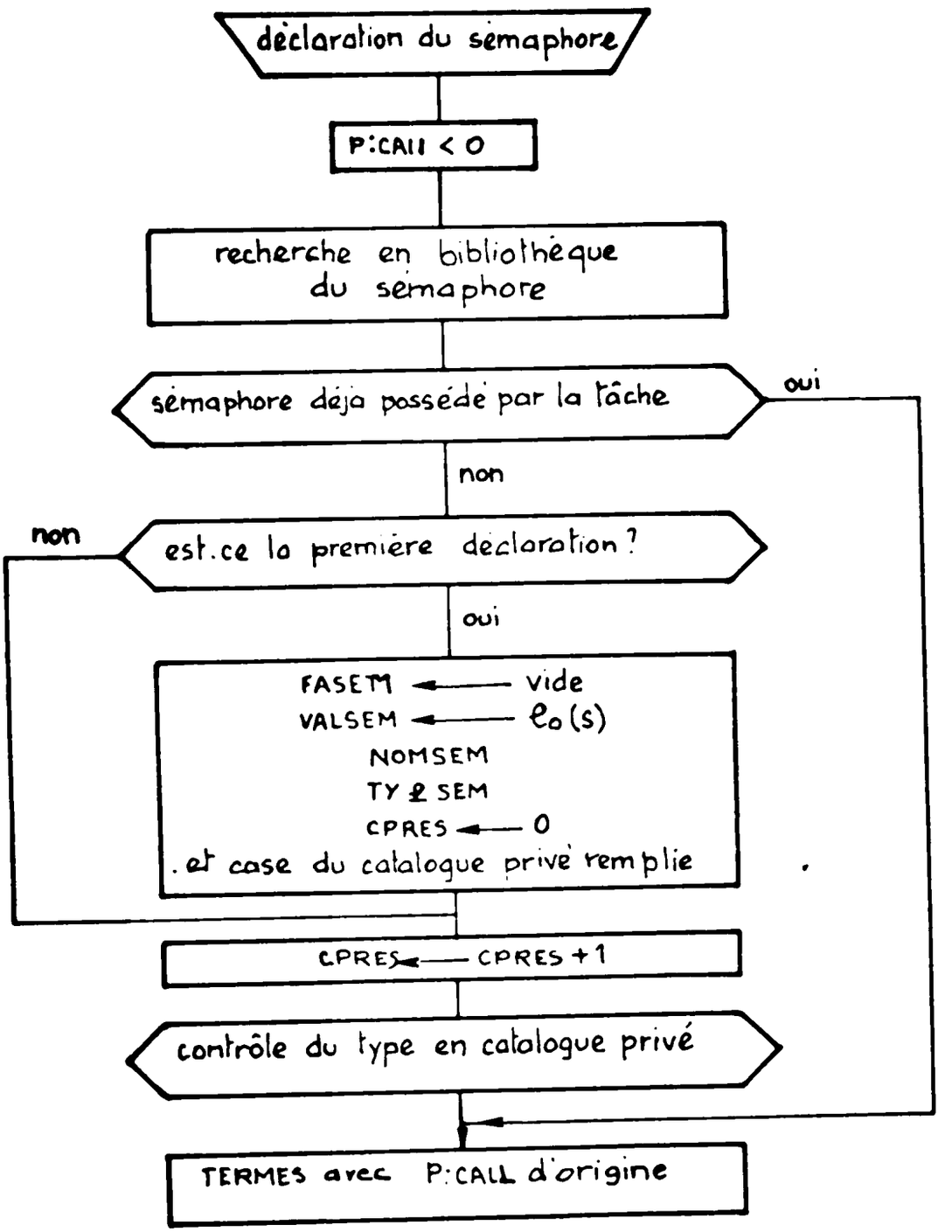
S8

Algorithme du lancement de l'entrée.so



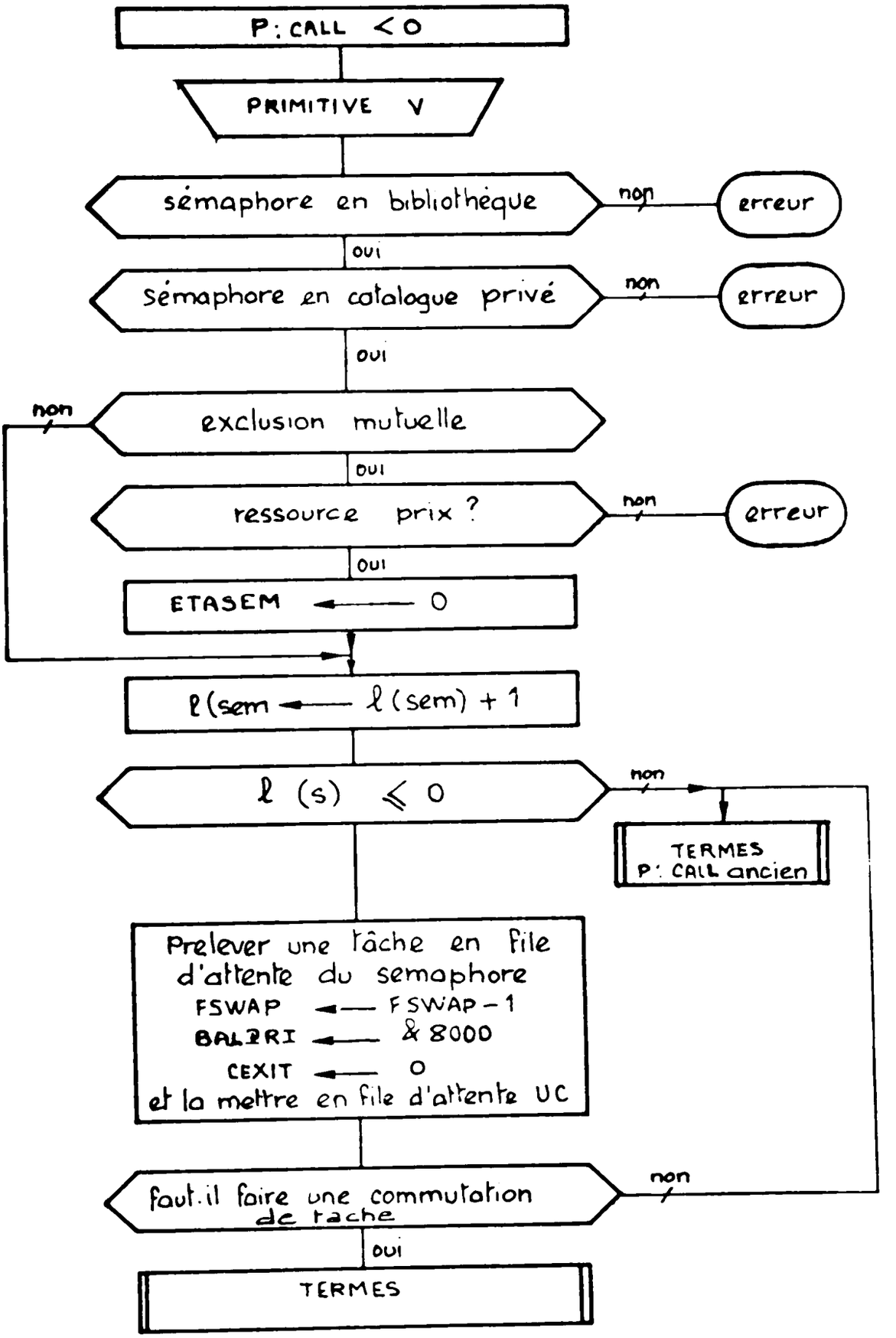
S9

Attente d'une entrée. sortie



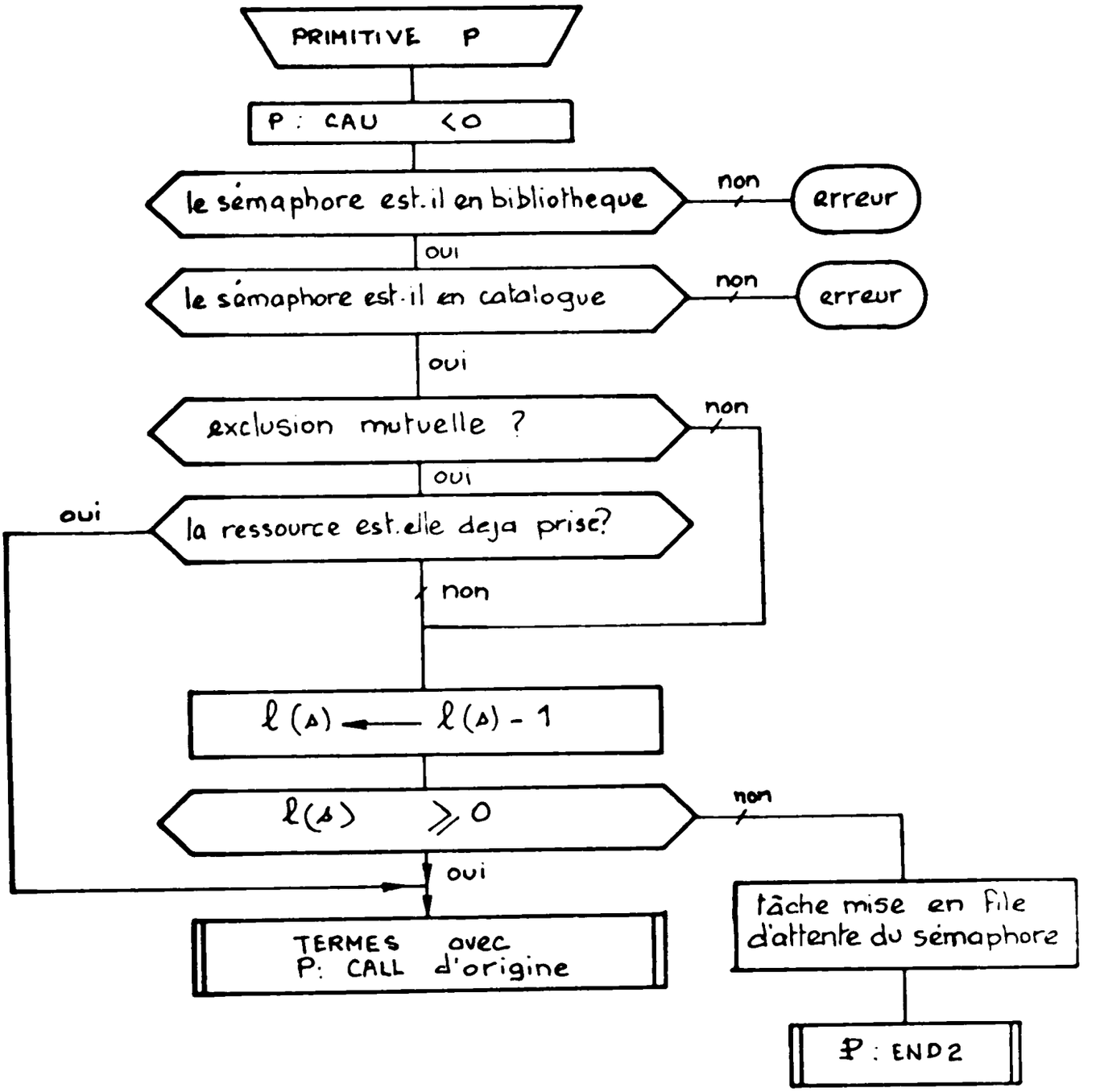
S10

Traitement de la primitive déclaration de semaphore



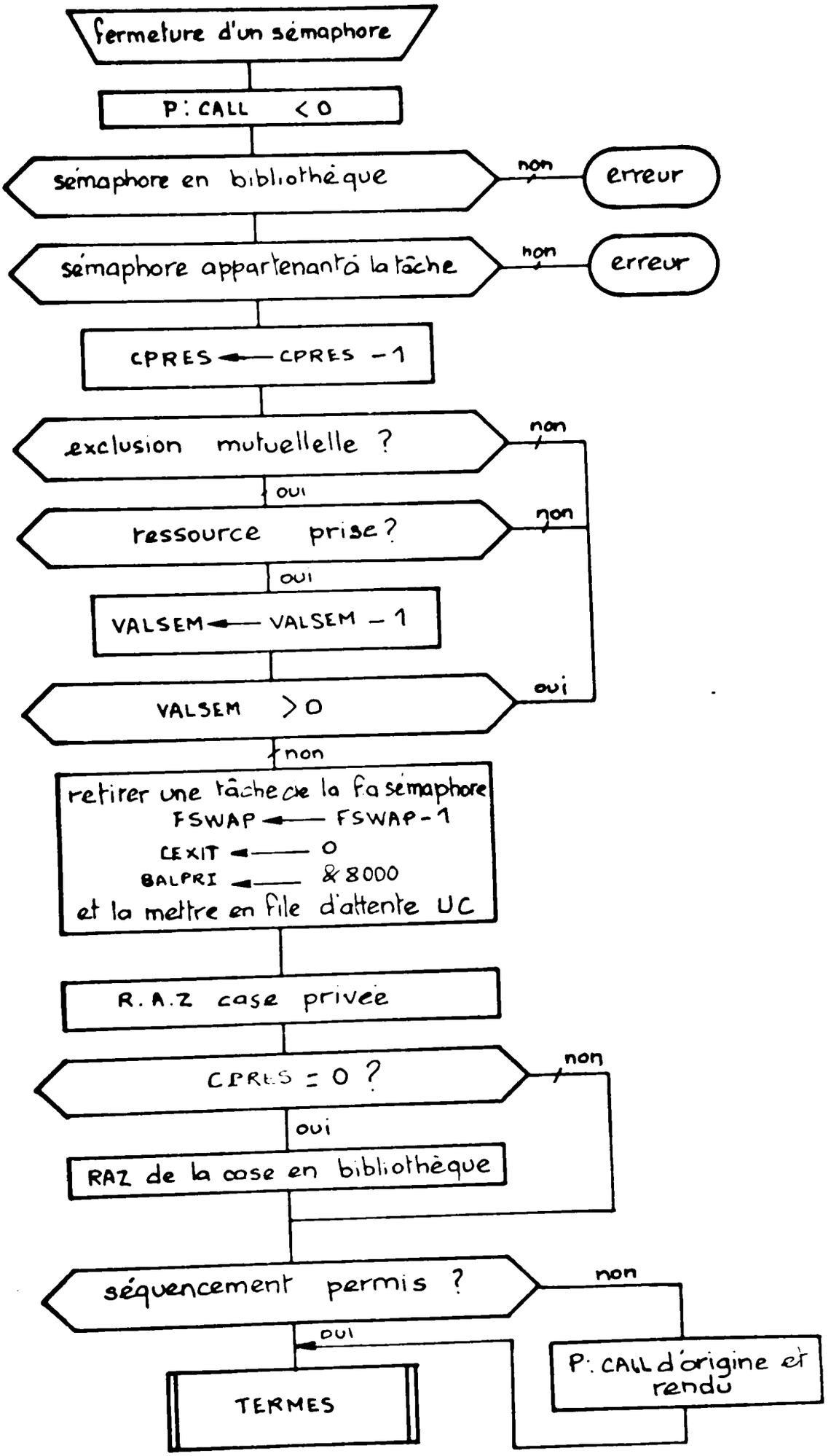
S11

Traitement de la primitive V sur un sémaphore.



S12

Traitement de la primitive P sur un sémaphore



S13

Traitement de la primitive de fermeture d'un sémaphore

ANNEXE C

LES PROBLEMES DE LA PROTECTION.

Le problème de la protection se pose nécessairement étant donné les communication entre les différents logiciels présents dans la machine. Le processeur EXEMCO, étant donné sa fonction, doit pouvoir accéder à toute la mémoire (tables du moniteur). L'interpréteur APL ne doit pouvoir accéder au moniteur d'exploitation (sauf par appel CSV) et dans une certaine mesure au superviseur multi-console. Rappelons que la tâche APL doit en effet accéder à son interface ainsi qu'il a été défini au chapitre 6 EXEMCO sera totalement protégé, ainsi que la zone privée de la tâche à l'exception de l'interface de la tâche en mémoire. Le planificateur SWAP assure tout à tour la protection / deprotection de l'interface de la tâche sortante/entrante.

La racine de l'APL est entièrement protégée en ce qui concerne les LDS et LPS de toutes les sections à l'exception des SRD de chaque LDS En effet, le processeur APL n'ayant pas accès aux zones protégées, les appels CLS ne forçant pas l'indicateur PR (à l'opposé du CSV) , le microprogramme ne peut sauvegarder les acresses de retour programme dans le SRD du sous programme : ceci entraine alors un déroutement. Les autres éléments du processeur constituant de fait des données propres à la tâche APL en mémoire, grâce au mécanisme de va-et-vient, il y a indépendance des tâche.

La tâche supplémentaire n'est pas protégée et ne doit pas pouvoir accéder à toute la mémoire.

TABLE DES MATIERES

BIBLIOGRAPHIE

I - SUR LE SYSTEME MITRA 15 - MITRA 125.

- Manuel d'utilisation moniteurs (SEMS)
  4557 UI/FR

- Manuel d'utilisation SCF15 - UCF15 (CII)
  4334 U/FR.

- Disque super DRI. 5440 URC 15280 (CII).
  TF D 01-09-01-A0.

- Génération de système (CII).
  1255 U2/FR

- Manuel de référence entrées/sorties , tome 2. (CII).
  4058 U/FR. 4058 8/FR REV 1.

- Guide du couplage (CII).
  4245 U2/FR.

- "Le Mitra-15 étude et réalisation d'un réseau multipoint centralisé à 10 Mégabauds".
  Thèse de Monsieur Michel MESTRALLET.
  21 juin 1975, Université PARIS XI.

II - SUR LE SYSTEME APL/15.

- "Conception et réalisation d'un interpréteur APL pour l'ordinateur CII Mitra 15".
  Thèse de Monsieur SMOUCOVIT A.
  9 novembre 1973, Université PARIS VI.

- "Interpréteur APL sur Mitra 15".
  Thèse de Monsieur DAVCO D.
  27 juin 1975, Université PARIS XI.

- "APL/MITRA 15 : Concept de mémoire virtuelle, optimisation et notions de structures".
  Thèse de Monsieur J.NEVIANS.
  26 juin 1975, Université PARIS XI.

- Le langage APL/MITRA.
  CISI/GIXI, juillet 1976.

III - SUR L'EVALUATION DU SYSTEME.

- Elements of probability for system design.
  A. O. ALLEN.
  IBM SYSTEMS JOURNAL Vol 13, n° 4, 1974.

- Elements of queuing theory for system design.
  A. O. ALLEN.
  IBM. SYSTEMS JOURNAL Vol 14, n°2, 1975.

- Single - server queuing processes in computing systems.
  W. CHANG.
  IBM SYSTEMS JOURNAL Vol. 9, n°1, 1970.

- Mesures et évaluation des systèmes informatiques.
  Journées ACM/IRIA (Colloques IRIA).
  Roquencourt, 21 - 22 octobre 1974.

- Introduction aux techniques d'évaluation et de mesure des systèmes informatiques.
A.SARZOTTI.
Editions Eyrolles 1977.

IV - SUR LE SYSTEME EN TEMPS PARTAGE.

- Logiciels de gestions de données et logiciels de transactions sur mini-ordinateur.
P.G. BAZELTOU et C.MEHL. Septembre 1976.
IRIA.

- L'exploitation partagée des calculateurs.
J.BERTIN, M.RITOUT, J.C. ROUGIER.
Monographies d'informatique AFI RO. DUNOD, 1967.

- Gestion des processus.
Thèse de docteur Ingénieur de la faculté des sciences de Toulouse.
J. FERRIE.
8 novembre 1971, IRIA ESOPE.

- Allocation des ressources.
Thèse de 3eme Cycle de la faculté des sciences de PARIS.
J. MOSSIERE.
9 juin 1971, IRIA ESOPE

- Conception et réalisation de systèmes à accés multiple : gestion du parallélisme.
C. KAISER.
IRIA , Note de travail ESOPE A 23, février 1973.

- Système d'exploitation des ordinateurs.
CROCUS.
DUNOD, 1975.

- L'informatique et les systèmes.
DUPUY
DUNOD, 1972.

- Machines de traitement de l'information.
Tome I et II.
P. DEBRAI NE.
MASSON et Cie., 1969.

- Operating system principles.
PER BRINCH HANSEN.
Prentice Hall Inc. 1973.

- Principes des systèmes d'exploitation des calculateurs.
Notes et polycopiés d'ateliers, CNAM, AURON, stage d'hiver, 17-25 février 1975.

- Conception et réalisation de systèmes à accès multiple :
Allocation de ressources.
Sacha. KRAKOWIAK,
Note de travail ESOPE A 24, février 1973.

- Ordonnancement des processus temps réel sur une ressource préemptive.
J. LABETOULLE.
Rapport de recherche n°74 IRIA. mai 1974.

- Aspects théoriques et pratiques des systèmes d'exploitation.
Colloques IRIA, Rocquencourt, 23-25 avril 1974.

- Time-sharing computers systems.
M.V. WILKES.
Mac Donald and Jone's, 1975.

- La structure de listes et leurs applications.
E&A SITBON.
MASSON, 1973.

- Introduction à l'informatique structure et programmation des ordinateurs.
I. DONDOUX, Ph. MARANO, J.C. MERLIN.
Armand COLIN, 1971.

- The art of computer programming. Vol I, II, III.
D.E. KNUTH.
Add-Wesley pub. Company, 1968, 1969, 1973.

- Cooperating Sequential Processes.
E.W. DIJKSTA.
Technological University, Eindhoven, 1965.