O R S T O M

Service Hydrologique
***

L'UTILISATION DES MEMOIRES AUXILIAIRES EN INFORMATIQUE AU SERVICE HYDROLOGIQUE DE L ' ORSTOM

par

H. DOSSEUR.

PARIS, septembre 1976

S O M M A I R E

L'UTILISATION DES MEMOIRES AUXILIAIRES EN INFORMATIQUE AU SERVICE CENTRAL HYDROLOGIQUE DE L'ORSTOM

****

Après la saisie de l'information sur un support primaire (qui, généralement, est la carte perforée) son traitement automatique nécessite sa mémorisation sous un faible volume de stockage et selon des critères de regroupements logiques permettant son immédiate disponibilité.

Pour assurer un tel stockage, on dispose de supports informatiques plus ou moins fiables et plus ou moins encombrants ; ce sont les "mémoires auxiliaires".

Ces mémoires sont des supports magnétiques qui doivent satisfaire à des exigences assez contradictoires puisqu'elles doivent permettre d'enregistrer le maximum de données et les restituer dans un minimum de temps.

Le choix de la mémoire auxiliaire et son utilisation pour un traitement donné pose à la fois des problèmes technologiques liés à la nature du support et des problèmes d'organisation (méthode de rangement de l'information sur le support).
Ces deux problèmes sont en fait fortement liés et conduisent en pratique à distinguer 2 types de mémoires auxiliaires magnétiques en fonction du mode d'accès à l'information :

- les mémoires à accès séquentiel : bandes magnétiques
- les mémoires à accès sélectif (ou direct) : disques, tambours magnétiques, feuillets magnétiques ...

Nous nous limiterons dans cette note aux seuls supports informatiques actuellement utilisés au Service Central Hydrologique, à savoir les cartes perforées employées comme support provisoire et les bandes et disques magnétiques employés comme support permanent.

L'utilisation efficace de ces mémoires nécessite d'avoir quelques connaissances sommaires sur la technologie des principaux supports employés et sur les différentes possibilités d'organiser l'information sur ces supports. Nous aborderons donc les notions de fichier et de structure des enregistrements et examinerons les principales méthodes d'accès à l'information en insistant plus spécialement sur les procédés utilisables en langage FORTRAN.

Nous donnerons ensuite les procédures de contrôle permettant la liaison entre le système d'exploitation et l'information ainsi mémorisée, c'est-à-dire les "instructions de contrôle" propre au Centre de Calcul dont nous utilisons actuellement les services (1).

---

(1) - Il s'agit du CIRCE = Centre Inter Régional de Calcul Electronique (CNRS-ORSAY) actuellement équipé d'un ordinateur IBM 370-168-168.

Enfin nous complèterons ce guide pratique en donnant le mode d'emploi d'un certain nombre de "programmes utilitaires" mis à la disposition des programmeurs pour faciliter l'exploitation des fichiers.

PREMIERE PARTIE : LES SUPPORTS D'INFORMATION

Nous n'examinerons dans ce chapitre que les 3 supports utilisés au Service Hydrologique de l'ORSTOM pour la gestion et l'utilisation des fichiers de données :

- les cartes perforées
- les bandes magnétiques
- les disques magnétiques

1 - LES CARTES PERFOREES.

Il s'agit de cartes dites "80 colonnes" mesurant 19 x 8 cm (norme AFNOR 55301).

La carte présente l'avantage d'être individuelle, palpable, de vérification facile et de stockage aisé.

Elle convient parfaitement pour la saisie primaire de l'information.

Ses inconvénients sont essentiellement sa capacité limitée (80 octets) et surtout son encombrement volumineux. En outre sa duplication est lente et coûteuse.

Le poids des cartes est un facteur important pour leur stockage :

- 2 000 cartes (contenu d'une boîte standard IBM) pèsent environ 5 kg
- 300 000 cartes stockées pèsent environ 1 tonne, y compris le matériel de stockage.

2 - LES BANDES MAGNETIQUES.

Une bande est constituée d'un film souple mince servant de support à la "couche magnétique" polarisable.

Elle se présente comme un ruban d'un demi pouce de large (12,7 mm) et d'en général 2400 pieds de long (730 m environ) enroulé sur une bobine.

2.1. Caractéristiques.

a) parité une bande magnétique supporte dans sa largeur 7 ou 9 bits d'information. (figure 1a)

En général le format d'enregistrement des données est le format "9 canaux" correspondant aux 9 bits d'information et caractérisant la bande comme une "bande 9 canaux".

Ce format utilise 8 des 9 bits pour les données et le dernier comme bit de contrôle ou bit de parité.
La parité est "paire" quand la somme des bits de valeur 1 d'une même verticale (y compris le bit de parité) est un nombre paire.

Si ce nombre est impair la parité est "impaire" (voir figure 1a).
Au CIRCE, le nom générique d'une bande 9 canaux dépend de la densité d'enregistrement (voir paragraphe b) :

2400-4 si l'enregistrement est effectué avec une densité de 800 bpi
BD16 si l'enregistrement est effectué avec une densité de 1600 bpi
BD60 si l'enregistrement est effectué avec une densité de 6250 bpi

De même le format "7 canaux" caractérise la bande comme une bande 7 canaux et n'utilise que 6 bits pour les données plus le bit de parité.

Au CIRCE le nom générique d'une bande 7 canaux est BDE7.

Suivant les constructeurs la parité est paire ou impaire (IBM adopte une parité impaire pour les bandes 9 canaux et soit paire soit impaire pour les bandes 7 canaux).

b) densité : c'est le nombre de caractères (ou octets) enregistrés par pouce. Elle s'exprime en bytes par inch (bpi).

La densité peut être 800, 1600 ou 6250 bpi pour une bande 9 canaux
250, 556 ou 800 bpi pour une bande 7 canaux (1)

Au CIRCE la densité utilisée par défaut est 1600 bpi pour les bandes 9 canaux et 800 bpi pour les bandes 7 canaux.

c) capacité : la capacité maximale théorique d'une bande de densité 1600 bpi et de longueur 2400 pieds (soit environ 730 m) est de l'ordre de 500 000 fois 80 caractères (ou octets). Mais il s'agit là d'une capacité idéale qui exigerait que la bande fût entièrement remplie d'enregistrements d'un bout à l'autre.

En réalité les enregistrements physiques sont séparés par des intervalles appelés "GAP" et la capacité réelle d'une bande dépend donc de la structure de l'information qu'elle contient (figure 1b).

Il y a également sur la bande des caractères de contrôle servant à tester la polarisation à intervalles de temps régulier (méthode d'enregistrement par modulation de phase P.E.). Connaissant la densité, le nombre d'enregistrements physiques (blocs) et leur longueur, le nombre de caractères de contrôle par bloc (généralement 82), la longueur d'un gap (0,6 pouce), il est donc possible de calculer l'encombrement d'un fichier sur une bande, on peut utiliser la formule suivante :

$$C = \frac{L \times B}{\frac{B + s}{d}} + e$$

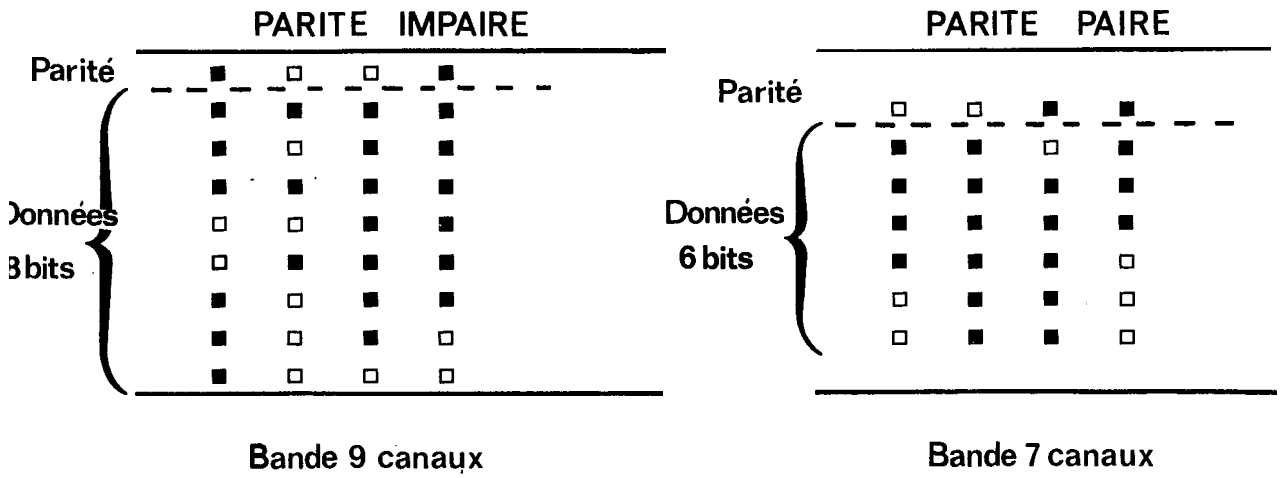
C : capacité de la bande en octets
L : longueur de la bande en pouces
B : longueur de l'enregistrement physique (ou bloc) en octets

---

(1) - A partir d'octobre 1976 l'unique dérouleur 7 canaux du CIRCE sera définitivement supprimé.

CARACTERISTIQUES DES BANDES MAGNETIQUES

Fig. 1

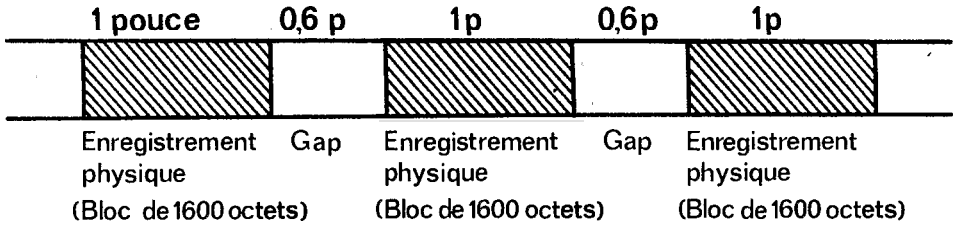


Bande 9 canaux

Bande 7 canaux

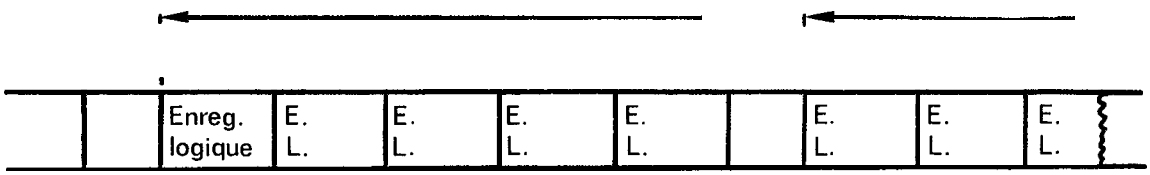
■ Bit à 1
□ Bit à 0

1a PARITE DES BANDES MAGNETIQUES



1b EXEMPLE DE STRUCTURE D'UNE BANDE MAGNETIQUE

(DENSITE 1600 bpi)



1c STRUCTURE D'UNE BANDE MAGNETIQUE

FACTEUR DE BLOCAGE 5

s : nombre de caractères de contrôle de polarité par bloc enregistré (en général 82)
d : densité d'enregistrement en bpi
e : longueur d'un gap (en général 0,6 pouces).

Le tableau 1 donne la capacité approximative des bandes magnétiques 9 canaux enregistrées en 1600 bpi en fonction du facteur de blocage des enregistrements, avec un longueur d'enregistrement logique de 80 octets (c'est-à-dire correspondant à 1 carte perforée 80 colonnes).

TABLEAU I

CAPACITE APPROXIMATIVE DES BANDES 9 CANAUX - 1600 BPI
EN EQUIVALENT CARTES 80 COLONNES

nombre de cartes par bloc (facteur de blocage)	nombre de caractères (ou octets) par bloc	longueur du bloc (inches)	longueur bloc + gap (inches)	nombre de blocs		nombre de cartes	
				bande 1200 feet	bande 2400 feet	bande 1200 feet	bande 2400 feet
1	80	0.05	0.65	22153	44307	22153	44307
2	160	0.10	0.70	20571	41142	41142	82285
3	240	0.15	0.75	19200	38400	57600	115200
4	320	0.20	0.80	18000	36000	72000	144000
5	400	0.25	0.85	16941	33882	84705	169410
6	480	0.30	0.90	16000	32000	96000	192000
7	560	0.35	0.95	15157	30315	106099	212205
8	640	0.40	1.00	14400	28800	115200	230400
9	720	0.45	1.05	13714	27428	123426	246852
10	800	0.50	1.10	13090	26181	130900	261818
12	960	0.60	1.20	12000	24000	144000	288000
20	1600	1.00	1.60	9000	18000	180000	360000
30	2400	1.50	2.10	6857	13714	205710	411420
40	3200	2.00	2.60	5538	11076	221520	443040
50	4000	2.50	3.10	4645	9290	232250	464500
60	4800	3.00	3.60	4000	8000	240000	480000
70	5600	3.50	4.10	3512	7024	245840	491680
80	6400	4.00	4.60	3130	6260	250400	500800
90	7200	4.50	5.10	2823	5647	254070	508230
100	8000	5.00	5.60	2571	5142	257100	514200
110	8800	5.50	6.10	2360	4721	259600	519310
120	9600	6.00	6.60	2181	4363	261720	523440
130	10400	6.50	7.10	2028	4056	263640	527280
140	11200	7.00	7.60	1894	3789	265160	530460
150	1200	7.50	8.10	1777	3555	266550	533250
160	12800	8.00	8.60	1674	3348	267840	535680
162 *	12960 *	8.10	8.70	1655	3310	268110	536220

(*) - Ces valeurs correspondent au nombre maximal d'image-carte que l'on peut mettre sur 1 piste de disque (13030 octets) de l'unité 3330.

2.2. Structure d'une bande.

Un enregistrement est une suite contigüe de données sous forme de 8 (ou 6) bits.

L'enregistrement est l'unité de base pour le programme de traitement.

Un fichier est une collection d'enregistrements.

La fin d'un fichier est marquée physiquement sur la bande par un enregistrement spécial appelé "TAPEMARK" (marque de bande).

On peut mettre plusieurs fichiers sur une même bande. On parle alors de volume multifichiers (voir figure 5).

Un seul fichier peut s'étendre sur plusieurs bandes. On parle alors de fichier multivolume (figure 5).

Chaque bande utilisée doit être identifiée. Pour cette identification on utilise un enregistrement spécial écrit au début de la bande appelé "LABEL VOLUME".

Sur une bande l'organisation des données est séquentielle, c'est-à-dire qu'un enregistrement n'est accessible qu'après avoir lu tous ceux qui le précèdent.

Tous les fichiers sur une même bande doivent être enregistrés avec la même densité.

La zone située entre 2 gaps est nommée "bloc" ou enregistrement physique

Elle se subdivise en plusieurs secteurs qui sont ceux des enregistrements logiques le nombre d'enregistrements logiques contenus dans un bloc est appelé "facteur de blocage".

Les avantages de la bande magnétique sont évidents :

- elle permet l'enregistrement de fichiers, même très importants, sous un faible encombrement,
- elle peut être effacée et réutilisée,
- elle peut être facilement et rapidement dupliquée,
- le coût du caractère stocké est très faible (le prix de vente actuel des bandes est de l'ordre de 100 F.

L'inconvénient essentiel de la bande est qu'elle ne permet pas l'accès direct à l'information. D'autre part la gestion des fichiers sur bande nécessite la recopie sur une nouvelle bande, toutes les fois qu'il y a des insertions, suppressions ou corrections à effectuer. (Cette procédure présente par ailleurs l'avantage d'assurer automatiquement la sécurité des fichiers).

3 - LES MEMOIRES A ACCES SELECTIF M.A.S. (Disques magnétiques).

3.1. Généralités.

- Une unité de disques est composée de disques métalliques enfilés sur un axe commun et enfermés dans un container.

L'unité de disques est appelée VOLUME.

On appelle MODULE l'ensemble constitué par les surfaces de disques empilés.

L'inscription se fait sur les faces des disques, une des faces extérieures (aux extrémités de la pile) n'étant pas utilisée pour les données.

Chaque face comporte un certain nombre de PISTES concentriques utilisables (plus quelques pistes de réserves) formant des cercles concentriques. Les pistes correspondantes de l'ensemble des faces constituent un CYLINDRE, (figure 2).
Les unités de disques à l'usage des utilisateurs du CIRCE sont du type 3330 compactes et amovibles (désignées sous le nom de DISPAC).

Certains disques sont toujours montés et sont dits "ON LINE" ou "résident"

Les autres sont montés uniquement sur demande et sont dits en "SETUP"

Au CIRCE l'utilisation des volumes se fait selon certaines conventions :

- Montage à la demande : SETnnn : le volume est monté spécialement pour le travail qui l'utilise. Cette opération est comptabilisée et nécessite la disponibilité d'une unité. Actuellement 12 unités en Setup sont disponibles (SET001, ..., SET012).
- Volume banal : BANinn : volumes utilisables en permanence mais aucun fichier créé sur l'un de ces volumes ne peut être conservé après la fin du travail qui l'a créé. L'utilisation de ces volumes n'est pas comptabilisée.
- Volume privé en ligne (ou résident) : RESinn : volumes utilisables en permanence réservés aux fichiers permanents des utilisateurs. L'espace utilisé leur est comptabilisé.

  Actuellement 4 unités "résident" sont disponibles RES301, RES302, RES303 et RES304.
- Volume privé provisoire : PP3330 : monté et utilisable en permanence mais tous les fichiers qu'il contient sont supprimés 1 fois par jour (entre 7 et 9 heures du matin). L'espace utilisé sur cette unité n'est pas comptabilisé.

3.2. Caractéristiques des unités 3330 : Ces caractéristiques sont rassemblées dans le tableau 2.

TABLEAU 2

CARACTERISTIQUES	3330-1	3330-11
Nombre de cylindres par module	404 (+7)	808 (+7)
Nombre de pistes par cylindre	19	19
Nombre de pistes par module	7676	15352
Capacité par piste	13030 octets	13030 octets
Capacité par cylindre	247570 octets	24570 octets
Capacité par module	100 millions octets	200 millions octets
Temps d'accès moyen	30 ms	30 ms
Vitesse de rotation	3600 t/mn	3600 t/mn
Débit	806000 octets /s	806000 octets/s

Actuellement 6 unités de disques du CIRCE sont du type 3330-11. Il s'agit des volumes BAN101, PP3330, RES301, RES302, RES303 et RES304.

Toutes les autres unités sont du types 3330-1.

3.3. Organisation d'un volume M.A.S.

- Le premier enregistrement contient le nom du volume. Il est appelé "VOLUME LABEL".

  Par exemple au CIRCE on aura selon le volume les noms suivants :

  SETnnn pour un disque à montage à la demande (Setup)
  BANinn pour un disque banal (ou disque scratch)
  RESinn pour un disque privé en ligne (résident)
  PP3nnn pour un disque privé provisoire.

  avec nnn ou inn = numéro d'ordre.

- Les enregistrements suivants contiennent la table des matières des noms des fichiers qui résident sur le volume. Elle est appelée VTOC ("volume Table of Contents"). Elle fournit les caractéristiques de chaque fichier et la place encore disponible.

  Le fait d'enlever de la VTOC le nom d'un fichier équivant à effacer le fichier du volume (sans pour autant récupérer la place de ce fichier). Cette opération s'appelle le "SCRATCH" du fichier.

  Les noms des fichiers sur un même volume doivent donc être tous différents.

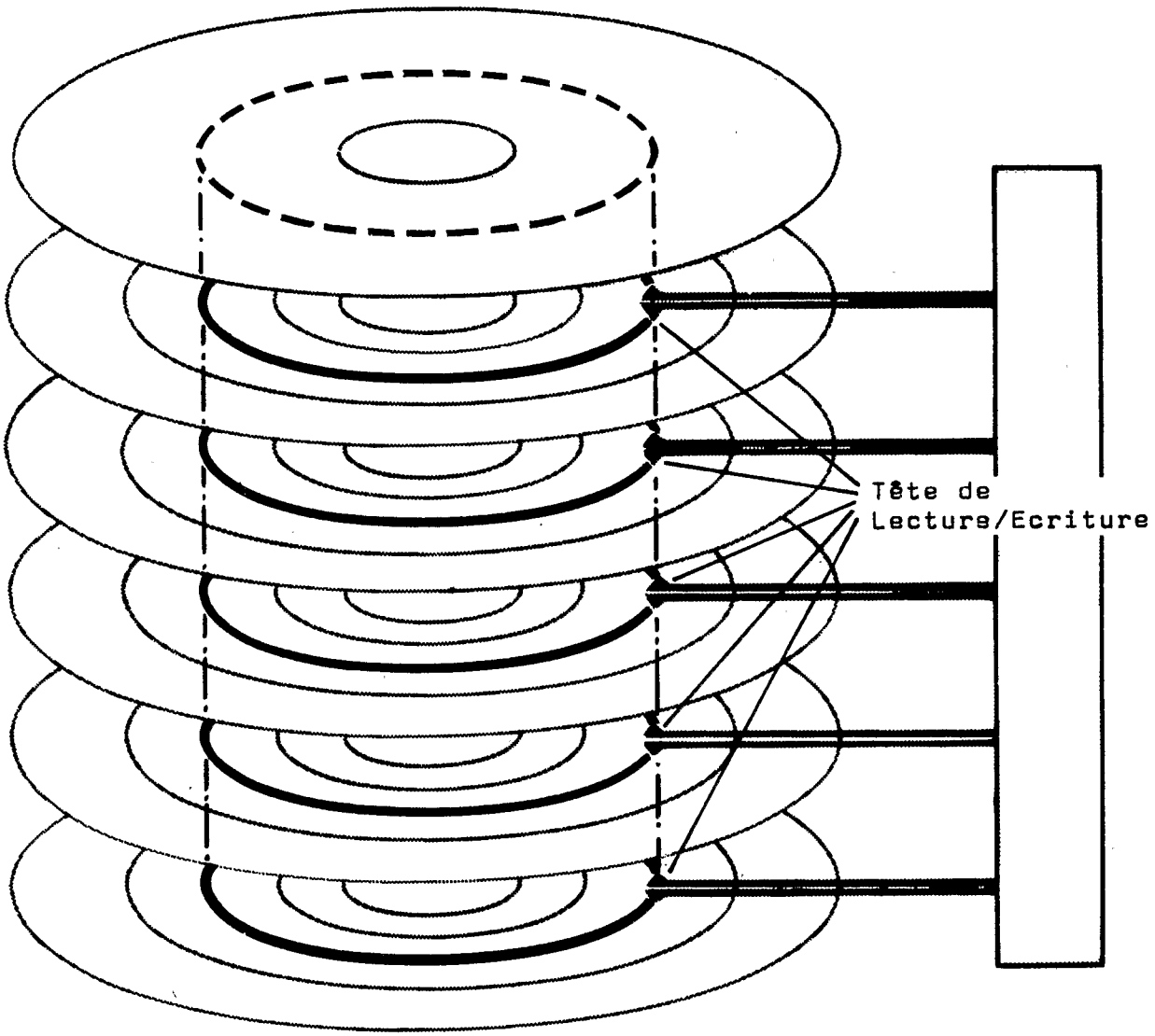
- En plus des données, chaque piste contient un certain nombre d'informations supplémentaires nécessaires au système d'exploitation. Il s'agit notamment de l'adresse de la piste, de l'adresse et de la position de chaque enregistrement, de la marque de début de piste (index point) et éventuellement des marques de secteurs.

  La piste contient également des espaces entre-enregistrements (Gap) qui, comme dans le cas des bandes magnétiques influeront fortement sur la capacité réelle d'information de cette piste.

- Chaque enregistrement de données contient sa propre description dans une zone de comptage. Il possède également une marque d'adresse et éventuellement une zone-clé :

  - La marque d'adresse (zone à 2 octets) permet au système de Localiser le début de l'enregistrement.
  - La zone de comptage donne la longueur de l'enregistrement et celle du bloc.
  - La zone-clé permet de reconnaître les données lors d'une seule révolution d'un disque. Elle contient l'indicatif (ou clé) de l'enregistrement. Sa longueur est comprise

DISQUES : NOTION DE CYLINDRE



Un cylindre est constitué par les pistes
ayant le même numéro sur toutes les faces

entre 1 et 255 octets et elle est nulle si les données n'ont pas d'indicatif (dans ce cas, la zone-clé n'existe pas).

- L'accès aux enregistrements est obtenu par un bras de lecture/écriture qui s'insère entre les disques et qui est appelé "peigne".

Pour une position donnée de ce dispositif, on accède simultanément à la quantité d'information contenue dans un CYLINDRE (voir figure 2).

Suivant le volume, le cylindre a un nombre fixe de PISTES (ou TRACK). La piste est l'unité de capacité élémentaire.

3.4. Organisation des fichiers sur le volume.

Sur les mémoires à accès sélectif, on peut avoir différents types d'organisation :

- organisation séquentielle : il s'agit alors d'une simulation de fichier bande,
- organisation partitionnée : le fichier est divisé en sous fichiers appelés membres.

Chaque membre est organisé séquentiellement et possède un nom. Le répertoire des noms des membres et leur adresse est appelé DIRECTORY. Il est placé en début de fichier.

- organisation directe : elle permet d'accéder à un enregistrement sans avoir à lire les enregistrements précédant celui-ci sur le support,
- organisation séquentielle indexée : elle permet la sélection, d'enregistrements d'un fichier grâce à un cheminement à travers le fichier.

Ces différents modes d'organisation seront examinés dans la 2ème partie avec la notion de fichier et les différentes possibilités d'accès à l'information.

Les avantages essentiels des mémoires à accès sélectif sont donc leur très grande capacité et la possibilité d'accéder directement et très rapidement à l'information à traiter. Les inconvénients sont surtout leur coût élevé et les risques de détériorations beaucoup plus grands que pour les bandes, (la sécurité d'un disque nécessite des recopies fréquentes de son contenu sur bande magnétique).

3.5. Utilisation pratique des unités 3330.

En pratique, la constitution physique de l'unité 3330 impose un nombre d'enregistrements par piste qui est fonction de la taille des enregistrements.

On trouvera dans le tableau 3, les possibilités de stockage de l'information sur les unités 3330.
Cette table de capacité permet d'obtenir la taille de l'espace à réserver pour un ensemble de données à écrire sur un disque 3330 (estimation du paramètre SPACE).

Connaissant la longueur de l'enregistrement logique (paramètre LRECL du DCB), on pourra ajuster le meilleur facteur de blocage (paramètre BLKSIZE du DCB) pour occuper au maximum une piste du disque.

La formule utilisée pour créér cette table est :

$$N = \frac{13165}{135 + C + KL + DL}$$

avec :

N = nombre d'enregistrements physiques (ou blocs) par piste

C = 0 si KL = 0

C = 56 si KL $\neq$ Ø

KL = longueur de la clé (organisation séquentielle indexée)

DL = longueur en octets du bloc à écrire.

Exemple : Si les blocs à écrire sont de 1000 octets, sans clés, le nombre maximal de blocs par piste sera :

$$\frac{13165}{1135} = 11,59 \quad \text{soit } 11 \text{ blocs.}$$

TABLEAU 3

TABLE DE CAPACITE DES DISQUES 3330

Nombre d'octets par enregistrement physique (sans clé)	Nombre d'enregistrements par piste	Nombre d'enregistrements par cylindre	Nombre d'octets par enregistrement physique (sans clé)	Nombre d'enregistrements par piste	Nombre d'enregistrements par cylindre
13030	1	19	164	44	836
6447	2	38	157	45	855
4253	3	57	151	46	874
3156	4	76	145	47	893
2438	5	95	139	48	912
2059	6	114	133	49	931
1745	7	133	128	50	950
1510	8	152	123	51	969
1327	9	171	118	52	938
1181	10	190	113	53	1007
1061	11	209	108	54	1026
962	12	228	104	55	1045
877	13	247	100	56	1064
805	14	266	95	57	1083
742	15	285	91	58	1102
687	16	304	88	59	1121
639	17	323	84	60	1140
596	18	342	80	61	1159
557	19	361	77	62	1178
523	20	380	73	63	1197
491	21	399	70	64	1216
463	22	418	67	65	1235
437	23	437	64	66	1254
413	24	456	61	67	1273
391	25	475	58	68	1292
371	26	494	55	69	1311
352	27	513	53	70	1330
335	28	532	50	71	1349
318	29	551	47	72	1368
303	30	570	45	73	1387
289	31	589	42	74	1406
276	32	608	40	75	1425
263	33	627	38	76	1444
252	34	646	35	77	1463
241	35	665	33	78	1482
230	36	684	31	79	1501
220	37	703	29	80	1520
211	38	722	27	81	1539
202	39	741	25	82	1558
194	40	760	23	83	1577
186	41	779	21	84	1596
178	42	798	19	85	1615
171	43	817	18	86	1634

2ème PARTIE : LES FICHIERS ET L'ACCES A L'INFORMATION

1 - NOTION DE FICHIER.

1.1. Généralités.

Un fichier est un ensemble d'informations de même nature rangées sur un support.
Ce support peut-être des cartes perforées, des bandes magnétiques, des disques ...

Un fichier peut avoir une durée de vie très variable, selon qu' il contient des informations de base, des données de caractères périodique ou de simples regroupements transitoires.
On parlera ainsi de fichiers permanents, semi permanents, de fichiers de mouvement ou de fichiers temporaires.

1.2. Les informations du fichier.

- Un fichier est constitué par une série d'enregistrements logiques

- Un enregistrement logique est une unité d'information de base pour un programme. Il est défini par les zones (données) qu'il contient et par son utilisation logique, mais non par la forme physique. Un programme de traitement traite les données sous la forme d'un enregistrement logique.

- Un enregistrement physique est une unité de données relatives à un support et à une organisation de fichier particulière. Un enregistrement physique peut contenir 1 ou plusieurs enregistrements logiques. Quand un enregistrement physique est constitué de plusieurs enregistrements logiques, on dit que ces enregistrements sont "groupés" ou encore "bloqués". Le nombre d'enregistrements logiques contenus dans l'enregistrement physique est appelé "facteur du groupage". Il est déterminé par le programmeur.

  L'enregistrement physique est l'unité technologique de traitement.

- Lorsque l'on créé un fichier, il faut indiquer au système :

  - le type de support et le nom du volume sur lequel, on veut le placer,
  - éventuellement la place à réserver au fichier sur le volume (cas des disques),
  - le nom sous lequel, il doit être identifié,
  - la disposition après création (conservation ou suppression),
  - les attributs du fichier = informations relatives à la taille et à l'organisation des enregistrements logiques et physiques.

Tous ces renseignements sont à indiquer au niveau d'une carte de contrôle appelée carte DD ("DATA DEFINITION") dont la description et l'utilisation seront données dans la 3ème partie de cette note.

1.3. Spécification des fichiers.

- La spécification des fichiers a été standardisée pour la gestion des données. Elle comprend des éléments tels que le format de l'enregistrement logique, la méthode d'accès, l'emplacement et l'utilisation du fichier.

- L'identification du fichier est obtenue par l'emploi de LABELS de volume et de fichier.
  Un LABEL est un enregistrement préformé lisible par la machine et utilisé pour identifier un volume ou un fichier.
  Chaque volume possède un label de volume et peut contenir un ou plusieurs fichiers. Par ailleurs un fichier peut être suffisamment grand pour nécessiter plusieurs volumes afin de contenir tous ses enregistrements.
  Chaque fichier peut donc avoir un ou plusieurs labels.

1.4. Gestion des données.

Au cours du traitement de l'information par un programme les déplacements des données entre la mémoire principale et les unités d'Entrée/Sortie sont effectués au moyen d'un système de gestion des données appelé IØCS (INPUT. ØUTPUT CØNTRØL SYSTEM).

L'IØCS comprend deux parties :

- l'IØCS physique qui est un ensemble de routines d'Entrée/Sortie qui supervisent la lecture et l'écriture des données sans tenir compte de leur contenu logique, ni de leur format.
  L'IØCS physique organise les opérations d'Entrée/Sortie, les contrôle et traite les erreurs.
- l'IØCS logique qui est un ensemble de routines pour le traitement des enregistrements logiques définis dans le programme de l'utilisateur.
  l'IØCS logique demande les opérations d'Entrée/Sortie à l' IØCS physique, traite les conditions EØF (fin de fichier) et EØV (fin de volume), contrôle et écrit les labels, groupe et dégroupe les enregistrements logiques.

1.5. Forme des enregistrements (voir figure 3).

1.5.1. Enregistrements non bloqués la gestion des données définit 3 formats pour les enregistrements logiques :

- Format Fixe (F) : Tous les enregistrements logiques ont le même nombre fixe de caractères (ou même longueur en octets). Il n'y a pas de compteurs.

- Format Variable (V) : Les enregistrements logiques ont un nombre variable de caractères (ou une longueur différente) (cf. figure 4). Un enregistrement de forme variable comporte une zone standard de 4 octets (CE) qui indique la longueur de l'enregistrement logique et une zone standard de 4 octets

O.R.S.T.O.M. Service Hydrologique

date / des.

DIV-261 842

RECAPITULATION DES DIFFERENTES ORGANISATIONS [Formats d'enregistrements]

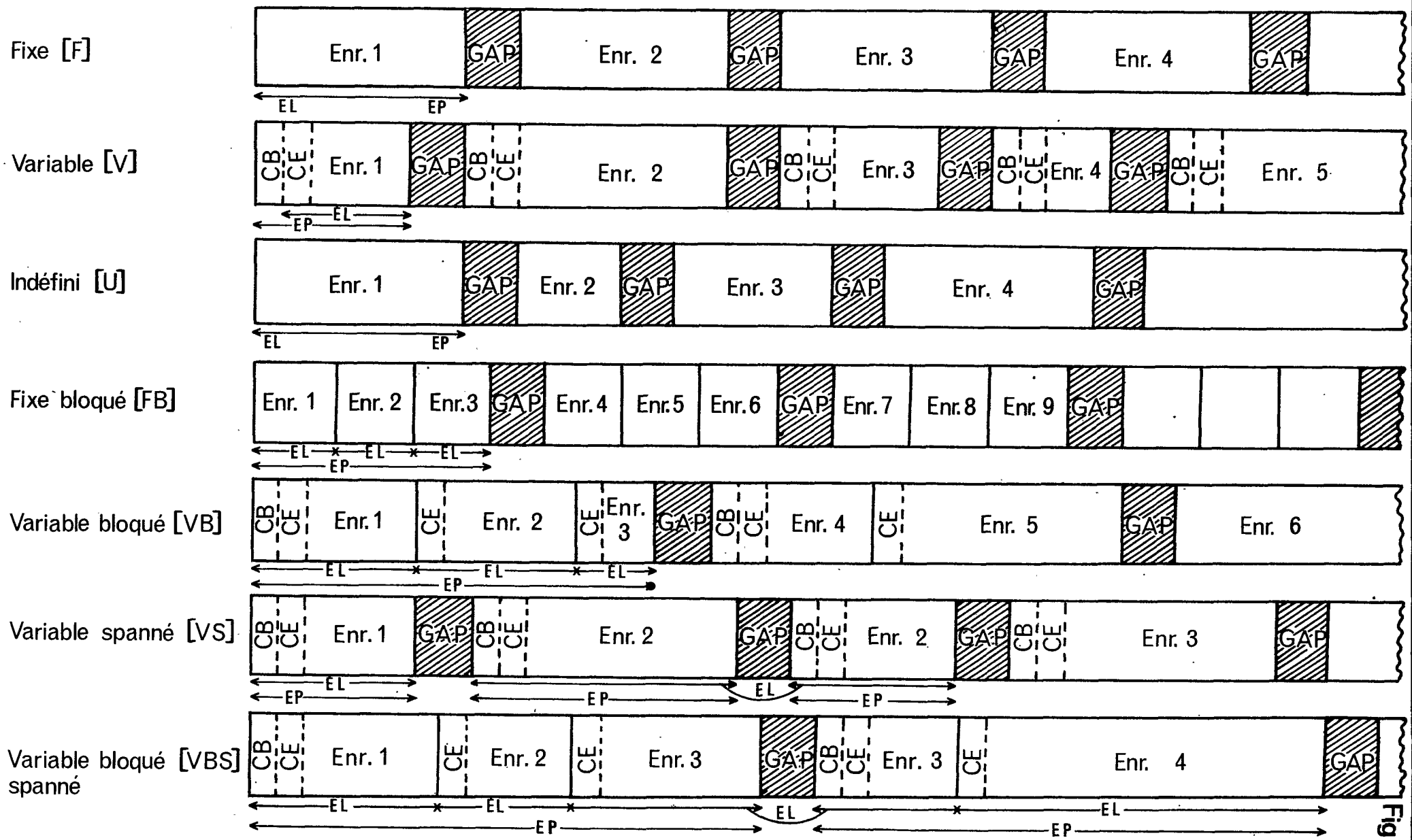
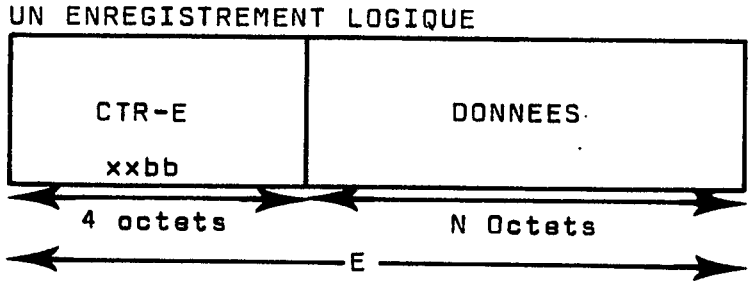
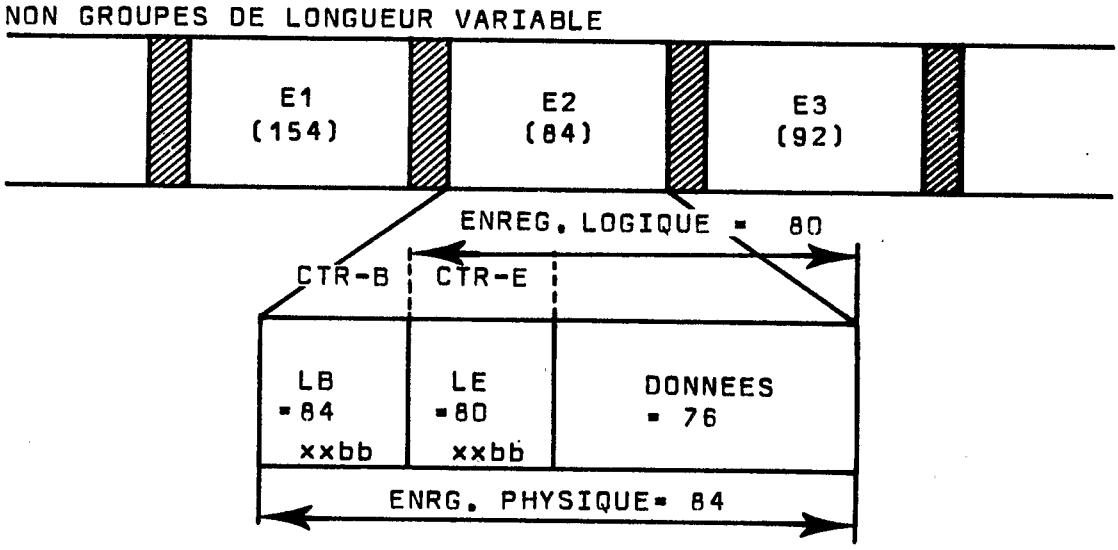


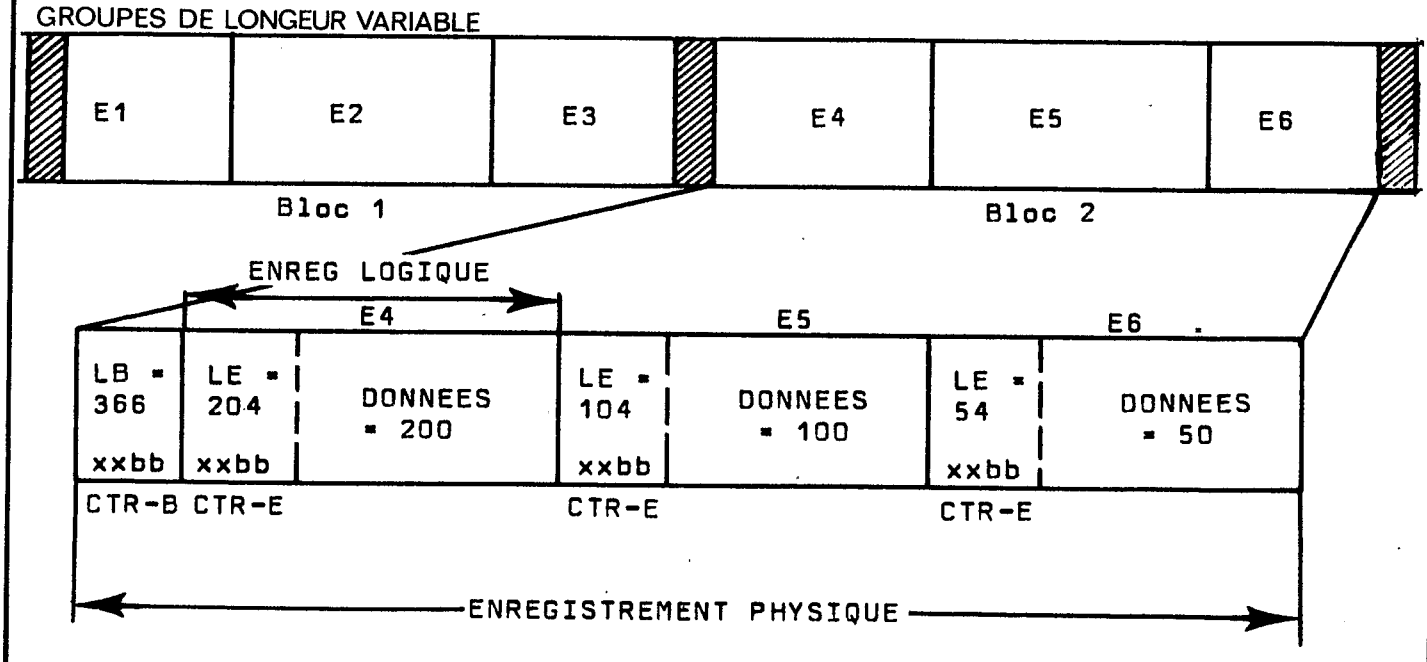
Fig 3

Fig.4

ENREGISTREMENTS DE LONGUEUR VARIABLE







O.R.S.T.O.M. Service Hydrologique

date | des.

DIV-261-851

ORGANISATION DES FICHIERS SUR BANDES MAGNETIQUES

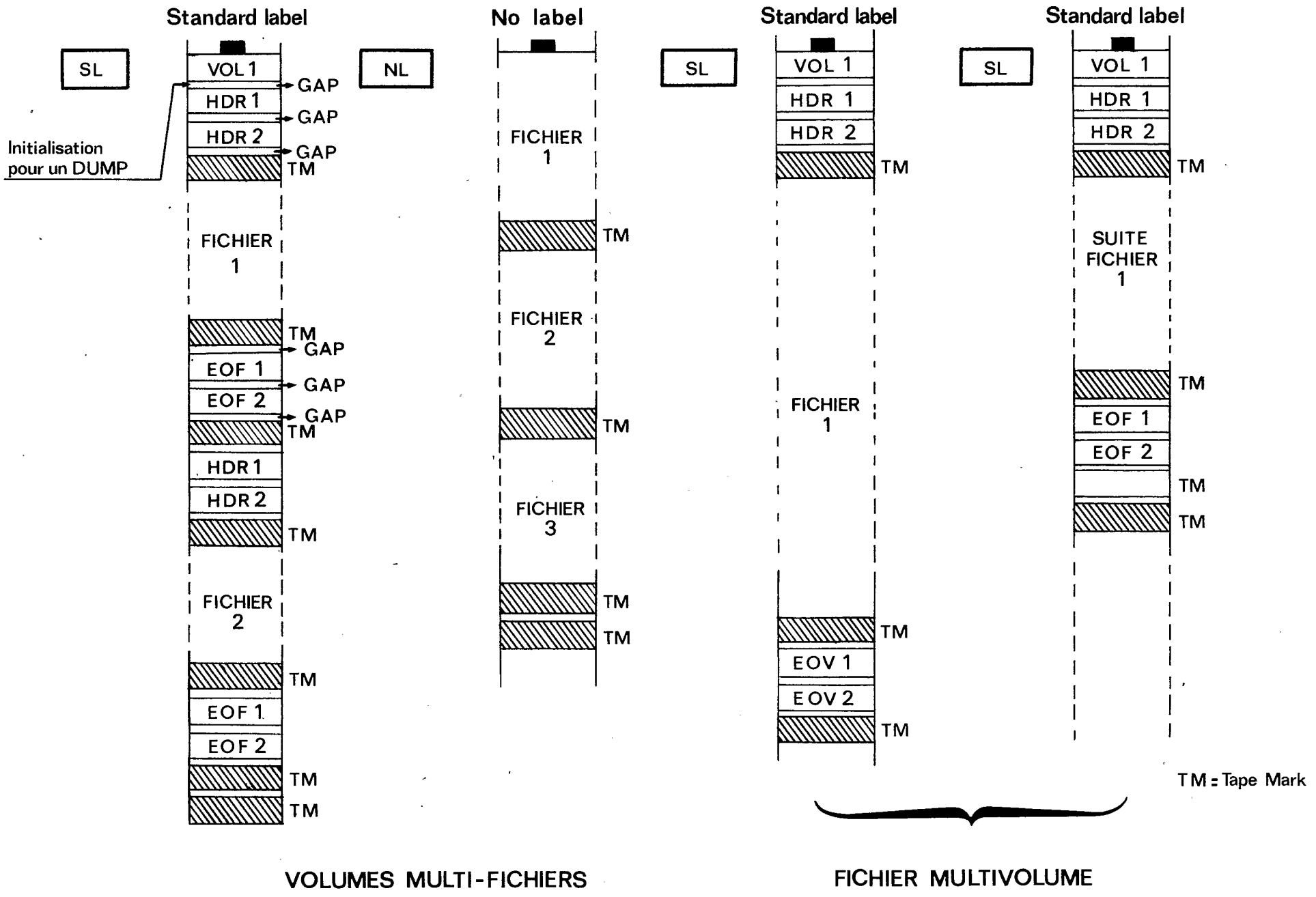


Fig-5

(CB) qui indique la longueur du bloc (ou enregistrement physique) qui dans ce cas est égale à celle de l'enregistrement logique (enregistrements non bloqués).

- Format Indéfini (U) : Les enregistrements logiques ont un nombres variable de caractères qui n'est pas défini.

1.5.2. Enregistrements bloqués. L'enregistrement physique (ou bloc) est constitué de plusieurs enregistrements logiques. On définit alors 3 autres formats :

- Format Fixe Bloqué (FB) : Plusieurs enregistrements logiques de longueur fixe sont placés sur le support de façon contigüe et forment un seul enregistrement physique. Tous les enregistrements physiques ont alors la même longueur.

- Format Variable Bloqué (VB) : Plusieurs enregistrements logiques de longueur variable sont placés sur le support et forment 1 seul enregistrement physique. Les enregistrements physiques n'ont donc pas tous la même longueur (cf figure 4). Une zone standard (CB) est réservée en début de bloc et indique la longueur de l'enregistrement physique en plus de la zone CE réservée en début de chaque enregistrement logique pour en indiquer la longueur.

- Format Indéfini Bloqué (UB) : Les enregistrements logiques de forme U peuvent être groupés, mais la longueur de l'enregistrement étant indéfinie, l'IØCS logique ne réalise par le groupage et le dégroupage. Ces opérations sont alors sous la responsabilité du programmeur.

- Pour les enregistrements de forme V ou U, les enregistrements physiques peuvent ne pas avoir tous la même longueur et le programmeur devra donc spécifier la longueur maximale de telle sorte que la zone réservée dans la mémoire principale soit suffisamment grande pour contenir l'enregistrement le plus long.
Cette spécification sera effectuée à l'aide des paramètres de la carte DD et sera examinée dans la 3ème partie de cette note.

- Dans un fichier, on ne doit pas mélanger des enregistrements de formes différentes.

- Des enregistrements groupés prennent moins de place sur le support et assurent une efficacité accrue au cours des traitements en diminuant les temps de lecture ou écriture. Le groupage peut cependant être limité par l'encombrement accru dans les zones d'Entrée/Sortie (capacité des buffers).

1.5.3. Options supplémentaires. Il existe d'autres formes particulières possibles d'enregistrements :

- Option S ("SPANNED OPTION") Cette option indique que les enregistrements logiques sont étendus c'est-à-dire contenus sur plusieurs enregistrements physiques.
Ceci n'est possible qu'avec des enregistrements de longueur variable et on définit ainsi 2 formats supplémentaires :

  - le Format variable spanné (VS)
  - le Format variable bloqué spanné (VBS)

- Option T ("TRACK OVERFLOW") Cette option indique que les enregistrements peuvent avoir une longueur supérieure à une piste de mémoire à accès sélectif.
En effet sur bande, la longueur maximale autorisée d'un enregistrement physique est de 32760 octets, mais pour les enregistrements sur mémoire à accès sélectif, elle ne peut en principe dépasser la longueur d'une piste de l'unité considérée (soit 13030 octets pour une unité de disque 3330) sauf si on demande le dispositif TRACK OVERFLOW qui permet une longueur maximale de 32760 octets.

- Option A ("ASA") Elle signifie que les enregistrements contiennent un caractère de saut ASA.
Par exemple pour les fichiers en sortie sur imprimante, on aura des enregistrements de format VBA (variable bloqué avec caractère de saut ASA).

2 - ORGANISATION DES FICHIERS ET ACCES A L'INFORMATION.

Les enregistrements logiques sont rangés et retrouvés sur le support suivant une organisation séquentielle ou directe qui dépend du traitement que l'on désire effectuer.

Cette organisation détermine donc le choix du support ainsi que le mode d'accès à l'information :

- accès séquentiel
- accès sélectif (ou direct).

2.1. Organisation et accès séquentiel.

C'est le seul mode d'accès possible pour les fichiers contenus sur bande magnétique (et bien entendu pour les fichiers sur cartes perforées).

Dans cette organisation les enregistrements sont placés les uns derrière les autres et pour lire un enregistrement, il est nécessaire de lire tous ceux qui le précèdent.

2.1.1. Organisation des fichiers sur bande magnétique : Nous avons vu que sur bande magnétique les fichiers sont identifiés par les LABELS de fichier, le volume lui-même (c'est-à-dire la bande magnétique) étant identifiée par le LABEL de volume.

Au cours de la création d'un fichier sur bande magnétique, on peut ou non créér les labels d'identification. Quand ces labels sont définis par la gestion des données, ils sont appelés STANDARD LABELS. Il faut savoir qu'au moment du traitement l' IOCS logique ne traitera que les "standard-labels".

Ces labels sont des enregistrements lisibles par l'ordinateur en plus des fichiers qu'ils servent à identifier.

La figure 5 donne des exemples d'organisation de fichiers sur bande magnétique avec ou sans labels standard dans le cas de volumes multifichiers et avec labels standard dans le cas d'un fichier multivolume :

- Le label de volume (VOL 1), quand il existe, est placé en tête de la bande. Il est écrit au moment où la bande est initialisée par l'opérateur et n'est en principe jamais changé. Donc quand un fichier est changé on ne recouvre pas le label de volume.

- Pour chaque fichier mis sur bande avec des labels standard IBM on a 2 labels début (HDR1, HDR2) placés en tête et 2 labels de fin (EØF1, EØF2) placés en fin de fichier. Ces labels contiennent des informations identiques relatives au fichier lui-même et en particulier son identificateur (DSN).

  Dans le cas d'un fichier multivolume, le label fin de la première bande utilisée (EØV1, EØV2) indique que le fichier continua sur un autre volume.

  Les labels sont séparés des fichiers par des marques de bande individuelles appelées TAPE MARK.

- Lorsque la mise sur bande a été faite sans labels, les fichiers sont directement séparés par une tape-mark.

- La fin des enregistrements sur la bande est également indiquée par une tape-mark qui est l'équivalent de la carte délimiteur (/ *) marquant la fin des fichiers cartes. (On a donc 2 tape-mark en fin du dernier fichier enregistré). Les labels sur bande permettent de nombreux contrôles effectués automatiquement par le système au cours du traitement :

  - vérification du numéro de volume monté par l'opérateur avec le label de volume,
  - vérification de l'identification du fichier (compatibilité avec le DSN de la carte DD),
  - vérification de l'ordre correct de traitement dans le cas de fichiers multivolumes

2.1.2. Opérations d'Entrée/Sortie en accès séquentiel.

- Si les enregistrements ne sont pas groupés, le système de contrôle d'entrée/sortie (IØCS) provoquera à la commande une lecture, et une seule à la fois, vers une zone d'entrée déterminée.
  S'ils sont groupés, l'IØCS provoquera chaque fois la lecture ou l'écriture d'un bloc entier.
  Dans les deux cas grâce à l'IØCS, tout se passe comme si le programme demandait la lecture d'un seul enregistrement logique à la fois.

- En langage FORTRAN, les instructions d'entrée/sortie en accès séquentiel sont les suivantes :

```
READ (n1, nf, ERR = e1, END = e2)liste
WRITE (n1, nf)liste
```

n1 = numéro logique, constante ou variable entière de longueur 4 définissant le fichier sur lequel, on lit ou on écrit. (référence à la carte DD)

nf = numéro de format (facultatif)

e1 = étiquette de transfert en cas d'erreur de lecture (facultatif)

e2 = étiquette de transfert en cas de fin de fichier en lecture (facultatif)

liste = liste de variables simples ou indicées (facultatif)

Rappelons que pour les fichiers cartes, la fin de fichier est indiquée par une carte de contrôle appelée "carte délimiteur" (ou "carte séparateur") qui comporte en principe (1) les caractères / * en colonnes 1 et 2. L'utilisation du paramètre END = e2 dans l'instruction READ est donc conseillée pour éviter un contrôle de fin de données (effectué par exemple sur une carte vierge).

Pour les fichiers sur bande, le contrôle de fin de fichier sera effectué sur la TAPE-MARK de fin de fichier.

- Cas particulier des instructions de lecture ou écriture sans FORMAT :

```
READ (n1) liste
WRITE (n1) liste        (n1 : numéro de l'unité logique)
```

Ces instructions permettent d'échanger des informations entre mémoire centrale et périphériques (en particulier disque et bande magnétique) et cela sans conversion, (transfert direct des bits de la mémoire centrale vers le support et inversement).
La lecture sans format provoque la lecture d'un enregistrement logique dont la longueur doit correspondre à celles des variables de la liste.

Exemple : Si on désire lire sans format sur l'unité 8 un enregistrement logique dont la longueur est de 1600 octets, la liste étant constituée par un tableau B (variables de longueur standard 4 octets) on écrira :

```
DIMENSION B(400)
READ(8) B
```

Attention : La lecture ou l'écriture sans format n'est possible qu'avec des enregistrements dont la forme est en variable spannée (RECFM = VS ou RECFM = VBS)

---

(1) - Les caractères / * de la carte séparateur peuvent être remplacés par un autre délimiteur qui devra alors être spécifié au niveau de la carte DD (voir 3ème partie).

[voir la signification de l'option "spannée" dans le paragraphe 1.5.3. de la 2ème partie de cette note.]

Par défaut le système 370 du CIRCE opère en format VS avec une longueur maximale d'enregistrement de 800 octets
(RECFM = VS, LRECL = 800, BLKSIZE = 800)

- D'autres instructions sont également permises :

READ nf, liste	Cette instruction équivaut à READ (5,nf) liste (lecture de cartes)
PRINT nf, liste	Cette instruction équivaut à WRITE (6,nf) liste (sortie sur imprimante)
PUNCH nf, liste	Cette instruction équivaut à WRITE (7,nf) liste (perforation de cartes en sortie).

- On dispose aussi d'instructions particulières :

END FILE n1

n1 = numéro logique : constante ou variable entière de longueur 4 définissant le fichier (référence à la carte DD).
Cette instruction définit la fin de l'ensemble de données associé à n1 en provoquant l'écriture d'un enregistrement de fin de fichier.

REWIND n1

n1 = numéro logique : constante ou variable entière de longueur 4 définissant le fichier (référence à la carte DD).
Cette instruction entraine le positionnement du dispositif de lecture ou d'écriture sur le premier enregistrement logique du fichier défini par n1.

BACKSPACE n1

n1 : numéro logique : constante ou variable entière de longueur 4 définissant le fichier (référance à la carte DD).
Cette instruction positionne le dispositif de lecture ou d'écriture sur l'enregistrement logique précédent.

Remarques importantes :

Les instructions REWIND et BACKSPACE ne sont utilisables qu'avec des ensembles de données à accès séquentiel sur bande ou sur disque.

En particulier, elles ne sont pas utilisables avec les fichiers sur cartes perforées.

2.2. Organisation sur mémoires à accès sélectif (accès direct)

Les mémoires à accès sélectif (disques magnétiques ...) permettent d'adresser un enregistrement et donc de le retrouver très rapidement sans avoir à lire ceux qui le précèdent sur le support. Cette méthode d'accès à l'information est appelée accès sélectif ou accès direct.

L'adresse peut donner 2 types d'information :

- l'adresse physique : qui décrit l'emplacement de l'enregistrement. Cette information permet l'accès aux enregistrements en ordre physique c'est-à-dire en se basant sur la position de l'enregistrement sur disque (exemple : accès direct en FORTRAN par l'instruction DEFINE FILE),

- l'adresse logique ou clé : affectée par l'utilisateur et qui est en général une information de contrôle faisant partie de la donnée de l'enregistrement.
  Cette clé permet l'accès aux enregistrements en ordre logique (exemple : séquentiel indexé).

Bien entendu l'organisation séquentielle pure permettant l'accès séquentiel (identique au mode d'accès sur bande magnétique) est également possible sur les mémoires à accès sélectif.
On aura également des systèmes d'organisation intermédiaires entre le séquentiel pur et le système par adressage (exemple : organisation partitionnée).

Nous examinerons ces différents types d'organisation et les possibilités d'accès à l'information qui leur correspondent.

2.2.1. Organisation partitionnée

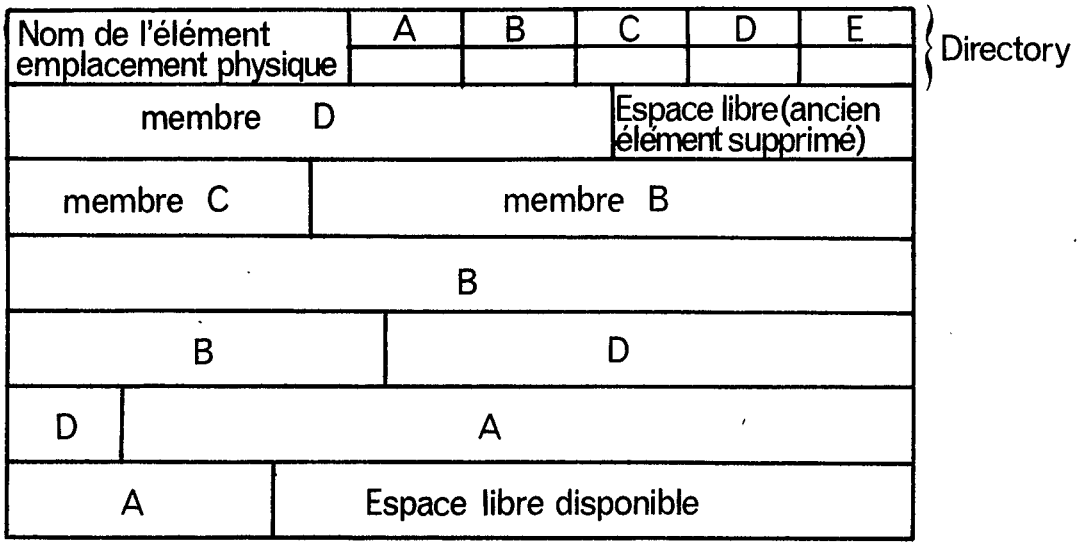
Il s'agit en fait d'une organisation particulière des fichiers (voir figure 6).
Dans cette organisation chaque fichier est divisé en sous-fichiers appelés MEMBRES. Chaque membre a une organisation séquentielle et possède un nom. Le répertoire des noms des membres et leur adresse M.A.S. sont placés en début de fichier et constituent le DIRECTORY (Répertoire).
Ce directory est formé d'enregistrements de longueur fixe (256 octets).

Au cours de la création d'un fichier partitionné, il faut prévoir la place nécessaire au directory qui est codée dans le 4ème sous paramètre du paramètre SPACE de la carte définissant le fichier (voir 3ème partie-instructions de contrôle - carte DD)

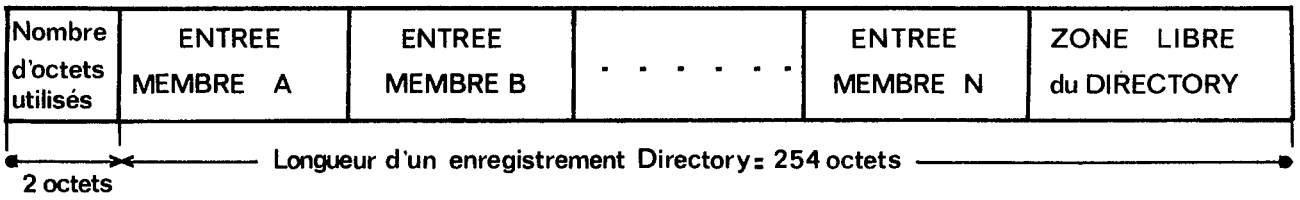
L'organisation partitionnée est obligatoire pour la sauvegarde du résultat du link-edit (load module exécutable d'un programme) que l'on désire conserver sur disque et utiliser ultérieurement. Dans ce cas le nom du membre du fichier partitionné est utilisé dans l'instruction d'appel pour l'exécution (EXEC PGM = MEMBRE).

Dans une organisation partitionnée tous les éléments ont des propriétés analogues (formats d'enregistrement, blocage, options utilisées ...). Chaque élément a un nom simple formé de 1 à 8

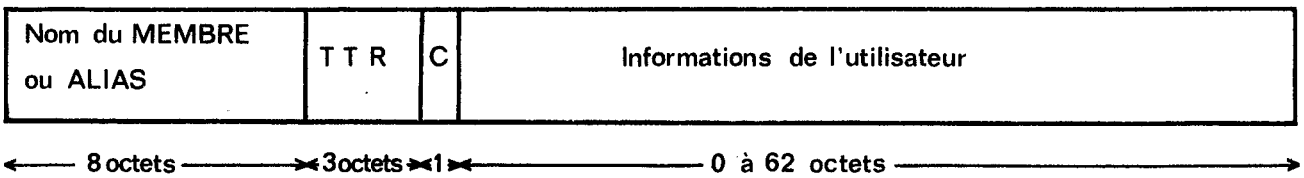
FICHIER PARTITIONNE



6a. Schéma d'un fichier partitionné

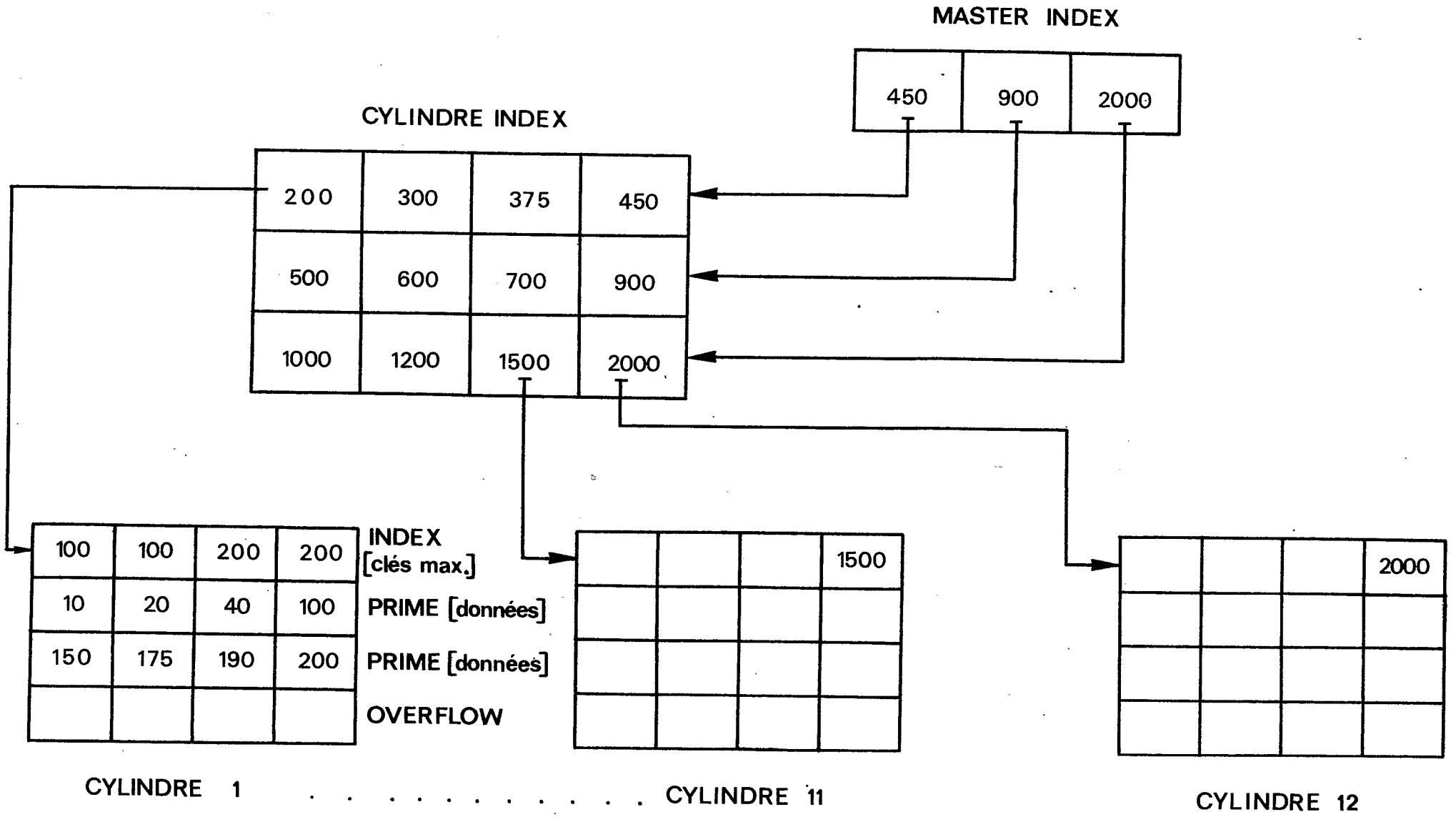


6b. Organisation d'un enregistrement du répertoire (Directory)



6c. Organisation d'une entrée du répertoire: ENTREE MEMBRE N

TTR: Pointeur sur le premier enregistrement du membre
C : Indicateur (si alias, longueur de zone, information, etc...)



EXEMPLE SCHEMATIQUE D'ORGANISATION SEQUENTIELLE INDEXEE.
[ASSEMBLEUR, PL/1, COBOL]

Fig- 7

caractères (voir 3ème partie paramètre DSN de la carte DD).

Une telle organisation permet de modifier un élément du fichier sans se préoccuper des autres.

Remarques :

- Le suppression d'un membre d'un fichier partitionné à l'aide du programme utilitaire IEHPROGM s'effectue par effacement du nom du membre de la directory. Par contre la place de ce membre sur le disque n'est pas restituée.
  On peut cependant récupérer cet espace libre en utilisant la procédure COMPRIME (voir 4ème partie : emploi des programmes utilitaires).
- On ne peut pas créér simultanément plusieurs membres d'un fichier partitionné. On peut par contre lire simultanément plusieurs membres.

2.2.2. Organisation séquentielle indexée (fig. 7) Ce type d'organisation ne peut pas être obtenu en langage FORTRAN.
Dans cette organisation les enregistrements sont rangés séquentiellement. Chaque enregistrement possède une zone de contrôle ou clé (KEY) et les données d'identification des clés sont classées par ordre croissant.

Au cours de la création du fichier, le système créé des index basés sur la clé afin de rendre possible la localisation ultérieure de tout enregistrement.

L'emplacement d'un enregistrement sur la mémoire est donnée au moyen d'une cascade d'index :

- index de cylindre (sur le cylindre-index)
- index de piste (sur chaque cylindre)

- Une entrée dans un index de cylindres contient l'identification d'un cylindre particulier et la reproduction de la clé la plus haute qui se trouve sur ce cylindre.
- Une entrée dans un index de pistes contient l'identification d'une piste particulière et la reproduction de la clé la plus haute qui se trouve sur cette piste.

La recherche d'un enregistrement se fait donc de la façon suivante :

La clé de l'enregistrement étant donnée,

1 - Recherche dans l'index de cylindre de l'entrée qui contient la clé la plus proche de (et supérieure à) la clé fournie.
Le système trouve alors un pointeur qui donne l'emplacement du cylindre qui contient l'enregistrement cherché.

2 - Dans le cylindre ainsi désigné, consultation de l'index de piste.
Le système cherche l'entrée qui contient la clé la plus proche de (et supérieure à) la clé fournie. Il trouve alors un pointeur qui donne l'emplacement de la piste qui contient l'enregistrement cherché.

3 - Sur la piste ainsi désignée recherche de la clé. Lorsqu'elle est trouvée l'enregistrement qui la suit immédiatement est transmis.

Une telle organisation permet donc d'utiliser un fichier :

- en lecture ou écriture d'enregistrements dont les clés sont en ordre ascendant,
- en lecture ou écriture d'enregistrements dont les clés sont données dans un ordre quelconque (utilisation des systèmes d'index),
- en écriture d'enregistrements avec de nouvelles clés. Le système prend en charge le choix de l'emplacement des nouveaux enregistrements et la mise à jour des index.

Pour la procédure à suivre en langage COBOL pour créér et exploiter un fichier organisé en séquentiel indexé, on se reportera à la notice IBM GRF2-4030-0 (Tome 9 ANS COBOL leçon 18 à 21).

2.2.3. Organisation aléatoire Ce type d'organisation ne peut pas être obtenu en langage FORTRAN.

Dans cette organisation, les enregistrements sont rangés "au hasard", cest-à-dire qu'ils sont dispersés à l'intérieur du support, chacun d'eux étant doté d'une adresse (clé) calculéepar l'utilisateur au moment de la création du fichier.

Par exemple si l'enregistrement possède un indicatif (ex : numéro d'identification ...) on passera de cet indicatif à l'adresse par un algorithme de transformation. En effet si en théorie rien ne s'oppose à ce que indicatif et adresse ne fassent qu'un, en pratique, il s'avère que l'indicatif ne peut généralement pas servir directement d'adresse, celle-ci ayant une structure commandée par la nature et la capacité de la mémoire auxiliaire utilisée.

2.2.4. Organisation directe Ce type d'organisation permet l'accès direct à l'aide d'instructionsFORTRAN. (FORTRAN IV H étendu). Cet accès direct a pour énorme avantage d'être beaucoup plus rapide que l'accès séquentiel.

Ce gain de temps dans les opérations d'entrée/sortie en programmation FORTRAN peut encore être amélioré par l'utilisation des instructions READ et WRITE sans FORMAT.

Dans l'organisation directe la forme des enregistrements est obligatoirement fixe. On spécifiera donc RECFM = F dans le paramètre DCB de la carte DD définissant le fichier (voir 3ème partie chapitre 7).
L'espace réservée pour le fichier à accès direct est figé et structuré dès sa création et ne peut donc être étendu.

Dans l'utilisation d'un fichier à accès direct les enregistrements sont repérés par leur adresse physique. La recherche d'un enregistrement se fait par déplacement relatif de l'adresse par incrémentation successive (la longueur de l'enregistrement étant fixe). Chaque enregistrement porte donc un numéro d'enregistrement qui lui est particulier et le programmeur doit spécifier dans les instructions READ ou WRITE ce numéro. Pour cela, il lui est nécessaire d'avoir l'inventaire du fichier qui lui permet de connaître l'emplacement de l'enregistrement à traiter.

En FORTRAN IV (H étendu) on dispose de 4 instructions spécifiques pour la création et l'utilisation de fichiers à accès direct sur disques; il s'agit des instructions suivantes :

DEFINE FILE

READ

WRITE

FIND

Ces instructions dont nous allons préciser la fonction et l'emploi permettent de caractériser un fichier à accès direct et d'effectuer sur ce fichier les opérations d'entrée/sortie et de recherche des enregistrements.

- Signalons au préalable une méthode particulièrement rapide permettant d'organiser en accès direct sur disque un fichier originellement organisé en séquentiel (et dont les enregistrements sont obligatoirement en format FIXE).

  On se contente dans ce cas de recopier le fichier d'origine sur un disque à l'aide du programme utilitaire IEBGENER (voir 4ème partie) en précisant éventuellement dans la carte de définition du fichier de sortie (carte SYSUT2 DD) l'attribut DSØRG = DA du paramètre DCB.

  exemple : A partir du fichier PJIVØ organisé en séquentiel sur la bande 104852, on désire obtenir un fichier ACDIR organisé en accès direct sur le disque SET008 :

```
//RMD80AbJØBbWRR1358,RØCHEMAR
//C1bEXECbPGM=IEBGENER,REGIØN=90K
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbUNIT=BD16,DISP=(OLD,KEEP,KEEP),
//  DSN=PJIVØ,VØL=SER=104852
//SYSUT2bDDbUNIT=3330,DISP=(NEW,KEEP),DSN=ACDIR,
//  VØL=SER=SET008,DCB=(DSØRG=DA,RECFM=F,BLKSIZE=2604),
//  SPACE=(TRK,(656,1),RLSE)
//SYSINbDDbDUMMY
 /*
//
```

Le fichier ainsi obtenu est utilisable en accès direct à condition d'utiliser les instructions spécifiques données ci-dessus (DEFINE FILE ...).

- Instruction FØRTRAN d'entrée/sortie en accès direct :

DEFINE FILE n1(n,l,f,v)

n1 : constante ou variable entière représentant le numéro logique de référence du fichier

n : nombre d'enregistrements maximum prévus

l : longueur de l'enregistrement en octets ou en mots (dépend du paramètre f)

f : mode d'enregistrement (avec ou sans format) :

f = L	avec ou sans format	1 donnée en octets (valeur du paramètre BLKSIZE de la carte DD)
f = E	avec format	1 donnée en octets (valeur du paramètre BLKSIZE de la carte DD)
f = U	sans format	1 donnée en mots (valeur entière de $\frac{BLKSIZE+3}{4}$)

v : nom de la variable associée. A la fin de chaque opération d'entrée/sortie v a une valeur égale au numéro d'ordre de l'enregistrement qui suit celui qui vient d'être transmis. Après l'exécution d'un ordre FIND la variable associée a pour valeur le numéro d'ordre de l'enregistrement sur lequel on veut se positionner.

L'instruction DEFINE FILE caractérise 1 ou plusieurs fichiers mis sur disque et sur lesquels on désire effectuer des opérations de lecture ou d'écriture en accès direct. Elle doit être placée avant toute instruction READ, WRITE ou FIND qui lui fait référence.

Il peut y avoir plusieurs instructions DEFINE FILE dans le même programme, à condition qu'elles se rapportent à des fichiers différents.

C'est le DEFINE FILE qui définit la longueur du buffer et la longueur de l'enregistrement et ceci ne peut être changé par les paramètres du DCB de la carte DD.

exemple : DEFINE FILE 8(40,200,U,IC)

Cet ordre caractérise un fichier de numéro logique 8. Ce fichier aura au maximum 40 enregistrements inférieurs ou égaux à 200 mots, les enregistrements seront lus ou écrits sans format, la variable associée étant IC.

La variable associée peut être utilisée dans des sous programmes.

```
READ (n1' r,nf) liste

WRITE (n1' r,nf) liste
```

n1 : numéro logique de référence du fichier

r : expression arithmétique entière représentant le numéro de l'enregistrement à lire ou à écrire

nf : numéro de format (facultatif)

liste : liste des variables simples ou indicées. La variable associée ne doit pas figurer dans la liste.

Ces instructions permettent de lire ou d'écrire 1 ou plusieurs enregistrements sur un fichier à accès direct ayant été préalablement caractérisé par un ordre DEFINE FILE.
La lecture et l'écriture se font avec ou sans format.

exemple :

```
      DEFINE FILE 8 (500,10,L,ID1), 10 (100,28,L,ID2)
      DIMENSIØN  M (20)
          :
          :
      ID2 = 21
          :
          :
   10 FØRMAT (5I2)
    9 READ   (8'16,10) (M(K),K = 1,10)
          :
          :
   13 WRITE  (10'ID2+5)A,B,C,D,E,F,G
          :
          :
      STØP
      END
```

Les fichiers à accès direct référencés 8 et 10 étant supposés mis sur le disque résident RES302,
Les cartes de contrôle définissant ces fichiers seront du type :

```
//GØ.FT08F001 DD DSN=WRR1328.RØCHEMAR.FICH1,UNIT=3330-1,
//    VØL=SER=RES302,DISP=(OLD,KEEP)
//GØ.FT10F001  DD DSN=WRR1328.RØCHEMAR.FICH2,UNIT=3330-1,
//    VOL=SER=RES302,DISP=(NEW,KEEP),SPACE=(28,(100) ),
//    DCB=(RECFM=F,LRECL=28,BLKSIZE=28)
```

Dans cet exemple, l'ordre 9 READ ... transmet après conversion suivant le format 10, 10 fois 2 caractères, soit 2 enregistrements du fichier référencé 8, dans le tableau M. Et ceci à partir de l'enregistrement numéro 16.
Après la lecture, la variable associée ID1 aura pour valeur 18.

L'instruction 13 WRITE ... transmet, sans conversion (écriture sans format) l'enregistrement numéro 26, soit 7 valeurs (28 octets) sur le fichier référencé 10.
Après l'écriture de cet enregistrement la variable associée ID2 a pour valeur 27.

Si dans un ordre READ ou WRITE la liste dépasse la longueur de l'enregistrement logique (taille maximum de l'enregistrement donnée par le paramètre 1 de l'ordre DEFINE FILE) :

- une erreur est signalée quand on fait une lecture ou écriture avec format
- l'enregistrement est étendu quand on fait une lecture ou écriture sans format, c'est-à-dire qu'un nouveau bloc est lu ou écrit. Si ce dernier est plus court que la longueur de l'enregistrement, il est complété par des zéros.

exemple :

```
DEFINE FILE 8 (400,10,U,IC)
DIMENSION A (4), B(10)
    :
WRITE (8'6) A,B
```

La longueur de l'enregistrement est de 10 mots. L'ordre WRITE écrit l'enregistrement n° 6 qui comprend le tableau A et les 6 premières valeurs du tableau B, et l'enregistrement n° 7 qui comprend les 4 dernières valeurs du tableau B.

FIND (a'r)

a = numéro logique de référence du fichier (constante ou variable entière)

r = expression arithmétique entière dont la valeur indique le numéro de l'enregistrement du fichier sur lequel on désire se positionner.

Cet ordre permet de lancer la recherche d'un enregistrement qui sera lu par la suite par un ordre READ. Ceci permet de faire un traitement en même temps, qu'une recherche donc de diminuer le temps d'exécution. Il n'y a aucun avantage à utiliser un ordre FIND avant un ordre WRITE.

exemple :

```
      DEFINE FILE 8 (500,80,L,IVAR)
   10 FIND  (8'50)
      :
   15 READ  (8'50)A,B
```

L'enregistrement 50 est localisé sur le fichier référencé 8 pendant que les ordres compris entre les étiquettes 10 et 15 s'exécutent. Après l'ordre FIND la variable associée IVAR a pour valeur 50 ; après le READ elle a pour valeur 51.

Exemple général d'utilisation de la méthode d'entrée sortie avec accès direct :

```
      DEFINE FILE 8(1000,72,L,ID8)
      DIMENSION   A(100),B(100),C(100),D(100),E(100),F(100)
   15 FORMAT (6F12.4)
      FIND (8'5)
      DØ 100 I = 1,100
      READ (8'ID8,15)A(I),B(I),C(I),D(I),E(I),F(I)
      FIND (8'ID8 + 4)
       :
  100 CØNTINUE
      DO 200 I = 1,100
  200 WRITE (8',ID8 + 4,15)A(I),B(I),C(I),D(I),E(I),F(I)
       :
      END
```

Cet exemple montre la possibilité de disperser et de regrouper des données.

La première boucle DØ lit les enregistrements n° 5, 10 ... 500 sur le fichier référencé 8.
Le tableau A reçoit la 1ère valeur de ces enregistrements, le tableau B la 2ème, etc ... L'ordre FIND permet de trouver l'enregistrement qui sera lu par la suite pendant l'exécution des ordres de la boucle.

La deuxième boucle DØ permet à partir des tableaux A ... F d'écrire des enregistrements. L'écriture commence à l'enregistrement n° 505, si la variable associée ID8 n'a pas été modifiée après le dernier READ.

Remarques :

- Les ordres END FILE, BACKSPACE, REWIND ne sont pas utilisables pour un fichier avec accès direct.
- Un fichier créé avec accès direct peut être lu en séquentiel dans une autre étape.

3ème PARTIE : LES INSTRUCTIONS DE CONTROLE

1 - GENERALITES.

1.1. Introduction.

Il existe différents types de systèmes d'exploitation pour la série des ordinateurs 370/IBM.

Au CIRCE, le fonctionnement des ordinateurs et le déroulement des travaux sont organisés et supervisés actuellement par :

- l'OS-MVT, version 21-6 (Operating System, Multiprogramming with a Variable number of Tasks)
- et par l'ASP, version 3 (Attached Support Processor).

Il s'agit d'un ensemble de programmes de traitement et de contrôle résidant dans l'ordinateur et facilitant les différentes phases des opérations :

programmes de traitement (compilateurs, utilitaires ...)

programmes de contrôle (gestion du travail et des données, gestion des ordinateurs interconnectés, (158 et 168) exécution simultannée de plusieurs programmes ...).

Ce système d'exploitation nécessite pour effectuer les travaux désirés (compilation, exécution, utilisation et création de fichiers ...) des renseignements complémentaires qui ne figurent pas dans le programme ou les données préparées par l'utilisateur.

Ces informations indispensables constituent "les instructions de contrôle" qui obéissent à des règles d'écriture très précises. (Une faute de syntaxe dans une instruction de contrôle entraîne un message du type "JCL ERROR" et la non exécution du travail).

Ces instructions peuvent être introduites dans l'ordinateur au moyen d'unités diverses. Cependant l'unité la plus couramment utilisée étant le lecteur de cartes, on parlera de "cartes de contrôle".

On a donc un véritable langage de contrôle ou JCL (Job Control Langage) qui permet de fournir des informations à l'ASP et à l'OS.

Les instructions commençant par //* sont analysées par l'ASP et ignorées par l'ØS. Ce sont des "instructions ASP".

Les autres sont des "instructions ØS/ASP".

Certaines collections d'instructions de contrôle fréquemment employées sont incluses dans une bibliothèque accessible en permanence. Elles forment alors une "procédure cataloguée". Il suffit pour la mettre en oeuvre de l'appeler par le nom qui lui est associé.

Les cartes de contrôle et les procédures cataloguées décrites dans cette note sont celles que nécessite l'installation 370/IBM du CIRCE à la date de rédaction.
Cependant nous ne parlerons ni des options prôpres au TSØ (Time Sharing Øption) ni des procédures supplémentaires qu'entraîne le traitement à partir de terminaux.
Nous mettons en garde le lecteur contre les légères mais fréquentes modifications apportées au langage de contrôle par suite de l'évolution constante des installations du Centre de Calcul.

1.2. Notions fondamentales.

L'utilisation du langage de contrôle nécessite au préalable de bien préciser les quelques notions définies ci-après :

- On appelle "ETAPE" (ou "STEP") le chargement et l'exécution de tout programme système ou programme-utilisateur.

  Une étape est définie par une instruction de contrôle EXEC (1).

- L'ensemble des étapes groupées en un même passage machine s'appelle un "TRAVAIL" (ou JØB).

  Un travail peut ne contenir qu'une seule étape.

  Un travail est défini par une instruction de contrôle JØB.

- On appelle "FICHIER" (ou "DATA SET") un ensemble de données d'un programme.

  Un fichier est défini par une instruction de contrôle DD (DATA DEFINITION).

- Un programme codé dans un langage évolué (FORTRAN, COBOL, PL1 ...) n'est pas exploitable tel quel par l'ordinateur. Il doit être traduit en langage machine (COMPILATION), rendu chargeable (EDITEUR DE LIEN ou LINK EDIT) et exécuté (EXECUTION).

  Prenons comme exemple le traitement d'un travail demandant l'exécution d'un programme codé en langage évolué à partir de données d'un fichier. Les différentes étapes sont les suivantes :

  - étape 1 (COMPILATION) : Traduction du "programme SOURCE" (codé en langage évolué) en "module objet" (codé en langage machine) à l'aide du traducteur de langage approprié (compilateur). Le module objet créé constitue un fichier désigné SYSLIN.

---

(1) - Une procédure cataloguée définie par une instruction de contrôle EXEC correspond généralement à plusieurs étapes.

- étape 2 (LINK EDIT) : Réalisation des liens entre le module objet et les sous programmes et fonctions de la bibliothèque des références externes à l'aide de l'éditeur de liens.
  Le résultat de cette opération donne le "module chargeable" qui constitue un fichier désigné SYSLMOD.

- étape 3 (EXECUTION) : Exécution du programme à partir des données par appel des différents fichiers les contenant.

  Signalons que dans le cas d'une exécution avec "CODE AND GO" (procédure WATFIV) il n'y a qu'une seule étape comportant la traduction de langage et l'exécution.

- L'enchaînement des étapes d'un même JOB se fait selon certaines conditions :

  - lorsqu'une étape se termine le système fournit un code (code de retour) qui correspond au résultat de l'exécution de l'étape.
    Ainsi en compilation ou assemblage, on a le code de retour suivant :

    0 aucune erreur n'a été détectée,

    4 il y a des erreurs mineures, l'exécution peut avoir lieu,

    8 il y a des erreurs et l'exécution risque d'être incorrecte, la compilation se poursuit jusqu'au bout. L'exécution n'aura pas lien (sauf demande expresse du programmeur),

    12 il y a des erreurs importantes. L'exécution ne peut avoir lieu,

    16 les erreurs sont telles que le traitement est immédiatement arrêté.

  - au début de chaque étape, figure une instruction de contrôle comportant un paramètre CØND (condition) qui est une expression logique formée d'opérateurs logiques (= , $\geqslant$, > , $\leqslant$, < , $\neq$) et d'opérandes (nombre de 0 à 4095).
    L'évaluation de cette expression donne le détail des valeurs que doivent avoir les codes de retour des étapes précédentes pour que l'étape considérée soit exécutée.
    Si les codes de retour ne satisfont pas aux conditions du paramètre CØND, l'étape n'est pas exécutée.

- Les données nécessaires à l'exécution d'un programme (ainsi que les données résultant de cette exécution) peuvent appartenir à différents fichiers placés sur des supports différents.

  Ainsi par exemple pour un même programme on pourra avoir en entrée :

  - des données d'un fichier f1 sur cartes perforées
  - des données d'un fichier f2 sur cartes perforées

- des données d'un fichier f3 sur une bande magnétique m1
- des données d'un fichier f4 sur un disque magnétique
- des données d'un fichier f5 sur une bande magnétique m2.

en sortie :

- des résultats constituant un fichier f6 sur cartes perforées
- des résultats constituant un fichier f7 sur imprimante
- des résultats constituant un fichier f8 mis sur disque magnétique
- des résultats constituant un fichier f9 sur imprimante.

Nous reviendrons dans cette note sur la procédure à suivre pour l'utilisation de plusieurs fichiers en entrée ou en sortie. Signalons en cependant quelques applications intéressantes à titre d'exemples :

- lecture de données de nature différentes sur cartes perforées : chaque type de données constituera un fichier distinct. On aura alors plusieurs "paquets" de cartes, ce qui évitera de les mélanger ou de les stocker en mémoire après lecture.
- sortie sur cartes perforées de données de nature différente : on créera simultanément autant de fichiers distincts que de type de données, ce qui évitera d'avoir à le séparer manuellement.
- sortie sur imprimante de résultats intermédiaires ou récapitulatifs sans avoir à les stocker en mémoire centrale avant impression. L'impression sera obtenue sur des fichiers distincts.
- séparation des données d'entrée en un fichier de données fixes mises sur un disque et lues à partir de ce disque et un fichier de données variables lues sur cartes (par exemple pour l'exploitation d'un modèle de simulation) etc.

2 - LES CARTES DE CONTROLE.

2.1. Objet des cartes de contrôle.

2.1.1. Les cartes principales Ces cartes ont pour objet :

- Soit de fournir les renseignements généraux et de définir les options principales intéressant l'ensemble du travail :

Ce sont les cartes | // JØB | et | //*MAIN |

- Soit de fournir pour chaque étape, les renseignements qui n'intéressent que cette étape :

carte | // EXEC |

- Soit de fournir, pour chaque fichier utilisé dans une étape, la description de ce fichier et de son support (définition de donnée : "Data Definition)

carte | // DD |

- Soit de poser l'enchaînement de divers travaux dans un ordre prédéterminé. (Réseau de travaux = "net")

carte | //* NET |

- Lorsque des résultats doivent être édités (listés ou perforés) dans des conditions particulières (plusieurs exemplaires, imprimé particulier ...) de donner les directives correspondantes (format d'édition : "format").

carte | //*FORMAT |

2.1.2. Les cartes secondaires. Elles ont pour objet :

- Définir une séquence de cartes de contrôle (ou procédure) qui sera utilisée plusieurs fois dans le même JOB.

cartes | // PRØC | et | // PEND |

- Indiquer les débuts et fin de fichier particuliers fournis sur cartes.

cartes | //*DATASET | et | //*ENDDATASET |

- Introduire, pour plus de clarté, des cartes de commentaires entre les instructions de contrôle

carte | /* |

- Préciser certaines conditions d'exécution et de restitution (priorité, délai d'attente ...)

carte | //*CIRCE |

- Préciser la fin d'un travail.

carte | // |

2.2. Syntaxe des cartes de contrôle.

2.2.1. Recommandations importantes.

- La perforation doit impérativement être effectuée en code EBCDIC (029).
- Existence de blancs significatifs.
- Les paramètres sont séparés par des virgules.
- Certains paramètres ont des sous-paramètres.

. Il y a différents types de paramètres et de sous-paramètres :

- à mot clé : forme PARM = ... (position indifférente)
- positionnel : position impérative.

. Certains paramètres ou sous-paramètres peuvent avoir des valeurs par défaut.

Dans cette note, nous utiliserons les conventions de notation suivantes :

b = au moins un blanc obligatoire

[.....] = paramètre optionnel

paramètre défaut = paramètre optionnel. S'il est omis, la valeur soulignée sera celle donnée automatiquement par le système (valeur dite "par défaut")

{choix 1 / choix 2 / choix 3} = l'un des paramètres (ou l'une des représentations) doit être choisi (e)

Choix1 | choix2 | ..... | choix xn = même signification que la notation précédente.

MAJUSCULES = mot clé dont l'orthographe est impérative.

2.2.2. Format des cartes de contrôle OS/ASP.

De la colonne 1 à la colonne 80, les zones sont réparties de la façon suivante :

//nombopérationbopérandesbcommentaires

On a 1 ou plusieurs opérandes séparées par des virgules.
Ces zones sont positionnelles, c'est-à-dire que c'est le nombre d'alternances zone vierge - zone perforée qui permet de savoir si une zone donnée doit contenir le nom, l'opération, les opérandes ou le commentaire.

exemple :

Si l'on a perforé //XXXbYYYbZZZ

la zone XXX contient le nom
la zone YYY contient l'opération
la zone ZZZ contient les opérandes

Si l'on a perforé //bXXXbYYYbZZZ

la zone XXX contient l'opération (le nom étant omis)
la zone YYY contient les opérandes
la zone ZZZ contient le commentaire.

Le détail des zones est le suivant :

- le nom : identifie l'instruction de contrôle permettant ainsi des références ultérieures. Il comporte 1 à 8 caractères alphanumériques, le 1er étant obligatoirement alphabétique. Cette zone commence obligatoirement en colonne 3 puisque les 2 premières sont occupées par // .

  exemple : //TRI1 ... ou //FUSION .. ou //P1 ....

- l'opération : précise le type d'instruction de contrôle dont il s'agit. C'est obligatoirement l'un des mots conventionnels choisis pour le langage de contrôle de l'Operating System : JOB, EXEC, DD ...
  Cette zone est obligatoirement précédée et suivie d'au moins un blanc.

  exemple : //TRAVAILbbJOB ...
  //bEXEC ...

- le ou les opérandes : séparés par des virgules contiennent des paramètres optionnels ou obligatoires, pour préciser l'opération demandée.
  Cette zone est obligatoirement précédée et suivie d'au moins un blanc. Elle ne doit pas dépasser la colonne 71 d'une carte.

  Si une carte ne suffit pas, une ou plusieurs cartes "suite" peuvent être utilisés en appliquant les règles décrites plus loin.

  exemple : //RMD76bJOBbWRR1358,ROCHEMAR,TIME=(0,10)

- le commentaire : est optionnel et n'est soumis à aucune restriction quant aux caractères utilisés ou à la colonne de début ou de fin.
  Cette zone doit obligatoirement être séparée de la précédente (zone "opérande") par un ou plusieurs blancs.

Seules les colonnes 3 à 71 peuvent être utilisées pour la perforation des noms, opération et opérandes d'une instruction de contrôle OS/ASP.
Si cet espace, n'est pas suffisant on applique les règles suivantes :

1. Un paramètre doit être perforé au complet sur une carte, y compris la virgule qui le suit immédiatement.

2. Si, sur une carte, une virgule est suivie de un ou plusieurs blancs, le reste de la carte est considéré comme un commentaire.

3. Chaque carte "suite" aura la syntaxe suivante :

```
//b P1,P2....
colonnes   paramètres
1 et 2   1 ou plusieurs blancs
```

Les blancs comprennent dans ce cas, au moins la colonne 3, au plus les colonnes 3 à 15.

2.2.3. Format des cartes de contrôle prôpres à ASP.

//*operationbopérandes

//* est perforé de la colonne 1 à la colonne 3. Il n'y a pas de zone "commentaire" autorisée dans ces cartes de contrôle.

Le détail des zones est le suivant :

- l'opération : précise le type d'instruction de contrôle dont il s'agit. C'est obligatoirement l'un des mots conventionnels choisis pour le langage de contrôle ASP : CIRCE, DATASET, FORMAT, MAIN ...

- le ou les opérandes : sont séparés par des virgules et contiennent des paramètres, optionnels ou obligatoires, pour préciser l'opération demandée. Cette zone est obligatoirement précédée et suivie d'un blanc au moins.

  Elle ne doit pas dépasser la zone 71 d'une carte. Si une carte ne suffit pas, une ou plusieurs cartes "suite" peuvent être utilisées en appliquant les règles suivantes :

  1. Perforer un caractère quelconque en colonne 72 de la carte qui doit avoir une carte suite.
  2. Dans la carte suivante, perforer //* en colonne 1 à 3 et la suite du texte à partir de la colonne 4.
  3. Un opérande (et les paramètres qu'il contient) doit être contenu tout entier sur une même carte.

3 - CARTE JOB.

Forme générale :

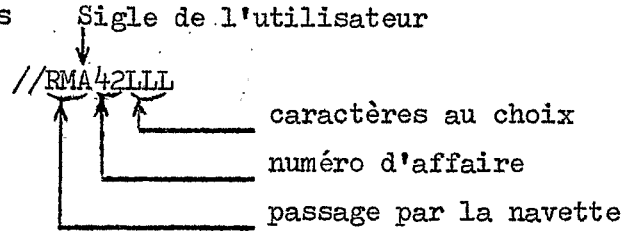
```
//nomdetravailbJØBbsiglenum,utilisateur [,TIME= { (mn,sec) / (,sec) / mn }]
                                                  30 secondes
[,MSGLEVEL=( { 0 / 2,0 / 1 / 1,1 } )] [,REGION=nnnnK]
```

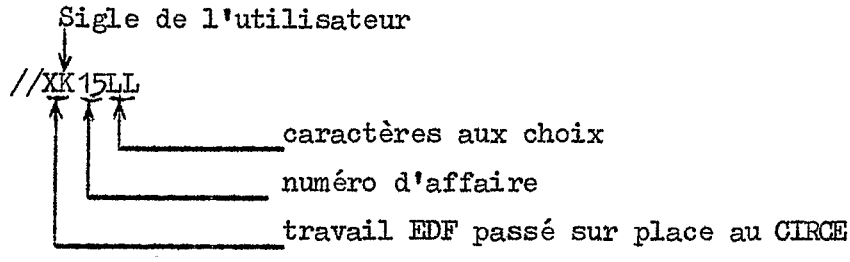
nom de travail : 1 à 8 caractères alphanumériques, le premier étant obligatoirement alphabétique.

Des modalités spéciales doivent être appliquées au Service Hydrologique :

- Pour les passages utilisant la navette de la rue du MAROC les 2 premiers caractères du nom de travail sont obligatoirement RM (colonnes 3 et 4). Ces 2 caractères disparaissent pour un passage sur place au CIRCE.
- Les autres caractères sont utilisés selon certaines règles permettant de faciliter la comptabilité des frais de calcul :
  - Si le travail correspond à une convention EDF : lettre X en colonne 5 (navette de la rue du MAROC) ou en colonne 3 (sur place). Aucune lettre particulière s'il s'agit d'un travail ORSTOM.
  - Ensuite un caractère permettant d'identifier l'utilisateur.
  - 2 chiffres correspondant au numéro d'affaire.
  - Les dernières colonnes disponibles (le total ne devant pas excéder 8) sont laissées au choix de l'utilisateur.

exemples





siglenum : sigle (3 lettres) et numéro de calcul (4 chiffres) attribués par le CIRCE.
Actuellement pour le Service Hydrologique le sigle est WRR
le numéro de calcul est 1358

utilisateur : nom de l'utilisateur 1 à 8 caractères alphanumériques, le 1er étant obligatoirement alphabétique.
Actuellement pour le Service Hydrologique (transmission par la navette de la rue du MAROC) tous les travaux ont pour nom d'utilisateur ROCHEMAR

TIME : Temps de calcul exprimé en minutes et / ou secondes et prévu pour l'ensemble du travail (utilisation de l'unité de calcul : temps CPU pour passage sur le 168 - SY1).
La valeur "sec" doit être inférieure ou égale à 59.
La valeur par défaut est 30 secondes.

Ce temps de calcul prévu doit être estimé par l'utilisateur. Il intervient dans la détermination par le système d'une priorité initiale d'exécution. Cette priorité peut être modifiée de 3 degrés au maximum en plus ou en moins sur demande de l'utilisateur à l'aide d'une carte //*CIRCE.(voir chapitre 10). Une demande de majoration de priorité entraîne une majoration de facturation.

Le tableau 4 donne la priorité initiale en fonction du temps de calcul prévu :

TABLEAU IV

Pour un temps de calcul compris entre	et	La priorité initiale (Pc) vaut
0	10 sec	10
11 sec	20 sec	9
21 sec	40 sec	8
41 sec	1 mn 20 sec	7
1 mn 21 sec	2 mn 40 sec	6
2 mn 41 sec	5 mn	5
5 mn 1 sec	10 mn	4
10 mn 1 sec	20 mn	3
20 mn 1 sec	40 mn	2
40 mn 1 sec	1 h 20 mn	1
1 h 20 mn 1 sec	24 h	0

REMARQUE : La valeur du paramètre TIME de la carte JOB n'intervient que pour l'attribution d'une priorité initiale de passage et non sur la facturation.
Pour celle-ci, c'est le temps réel d'utilisation de la mémoire centrale (temps CPU) qui est pris en compte.

Ce temps CPU est donné sur la feuille comptable à la restitution du travail et le détail pour chaque étape est donné en tête du listing (dans l'état SYSMSG) par des messages de la forme :

IEF374I STEP /FORT /STOP 760541340 CPU 0MIN 01.46SEC MAIN 248K . LCS OK

nom de l'étape	date de la fin d'exécution	temps CPU pour l'étape FØRT (compilation)	encombrement mémoire.

Une demande insuffisante de temps entraîne l'arrêt de l'exécution lorsque le temps demandé est écoulé et l'impression en tête de listing d'un message d'erreur du type :

COMPLETION CODE - SYSTEM = 322 USER = 0000

MSGLEVEL = (instructions, messages)

- instructions : 0 : liste de la carte JOB
  1 : liste de toutes les cartes de contrôle y compris celles des procédures cataloguées. (Valeur prise par défaut)
  2 : liste des cartes de contrôle de l'utilisateur
- messages : 0 : messages d'allocation seulement si fin anormale
  1 : liste des messages quelle que soit la fin. (valeur prise par défaut).

Pour l'utilisation de fichiers sur bandes ou disques et pour la mise au point de programmes, il est conseillé de prendre l'option :

MSGLEVEL = (1,1)

ou ce qui revient au même à supprimer le paramètre MSGLEVEL (option 1,1 prise par défaut).

REGION = encombrement maximal en mémoire centrale nécessaire à chaque étape du travail. Exprimé en K octets (K = 1024).
Il est préférable d'utiliser le paramètre REGION de la carte //bEXEC.

La carte JOB est la 1ère carte de contrôle. Elle se place en tête du JOB. Elle est obligatoire.

Exemple d'utilisation de la carte JOB :

//RMD76DEBbJØBbWRR1358,ROCHEMAR,TIME=(0,20)

4 - CARTE MAIN.

Forme générale :

```
//*MAINb [ LINES = { nnn     } ]   [ ,CARDS = { nnn     } ]
                   { (nnn,D) }                { (nnn,D) }
                   { 2000 lignes }            { 200 cartes }
```

LINES = nombre maximum de lignes à éditer, en milliers pour l'ensemble des fichiers du travail destinés à l'impression (type SYSOUT). Valeur par défaut : 2000 lignes.

CARDS = nombre maximum de cartes à perforer, en centaines pour l'ensemble des fichiers du travail destinés à la perforation (type SYSOUT). Valeur par défaut : 200 cartes.

D = si le nombre maximum de lignes (ou de cartes) prévu est atteint en cours d'exécution, un dump du programme sera tiré et listé. Une carte du type //SYSABENDbDD ou //SYSDUMPbDD doit alors figurer parmi les cartes de contrôle.

Si le travail provoque le dépassement de ces quantités maximales (LINES ou CARDS) il sera interrompu et donnera dans l'état du SYSMSG un message d'erreur du type :

COMPLETION CODE - SYSTEM = 222

Pour estimer le paramètre LINES, il faut se rappeler qu'il est compté une ligne d'impression à l'exécution de chaque instruction WRITE. Il est donc fortement déconseillé d'utiliser dans une instruction FORMAT le caractère de contrôle de saut + (pas d'interligne) car avec cette option la ligne éditée correspond à deux instructions WRITE. Par contre l'option 0 (double interligne) entraîne l'édition d'une ligne blanche (entre 2 lignes imprimées) qui n'est pas comptabilisée.

Notons enfin qu'une page de listing comprend environ 60 lignes d'édition.

La carte MAIN est facultative. Quand elle existe, elle se place derrière la carte JOB.

Exemple d'utilisation de la carte MAIN :

```
//*MAINbLINES=10,CARDS=5
```

Demande de 10000 lignes en impression et de 500 cartes en perforation.

5 - CARTE FØRMAT.

Forme générale

```
//*FORMATb  { PR, texte 1 }
            { PU, texte 2 }
```

PR pour un fichier à lister (PRINT)
PU pour un fichier à perforer sur carte (PUNCH).

Cette carte sert à indiquer la destination des fichiers à éditer de type SYSØUT et à donner les directives correspondantes. (voir signification de SYSØUT chapitre 7, paragraphe 7.1.).

Texte 1 : DDNAME = [ddnom] [,DEST = LØCAL | unité d'impression imprimante associée au lieu d'origine] [,CØPIES = nn 1 exemplaire] [,CØNTRØL = SINGLE | DØUBLE saut programmé] [,FØRMS = type formulaire usuel] [,ØVFL = ØFF saut de page] [,TRAIN = TN / QN]

Texte 2 : DDNAME = [ddnom] [,DEST = ANYLØCAL | unité de perforation perforateur associé au lieu d'origine.] [,CØPIES = nn 1 exemplaire]

La signification des paramètres est la suivante :

- ddnom = nom de la carte DD définissant le fichier auquel se réfère cette carte FØRMAT si nécessaire, ce nom peut être de la forme :

  nom d'étape.nom d'étape de procédure.ddnom

  si DDNAME = n'est suivi d'aucun ddnom, cette carte FØRMAT sera utilisée pour tous les fichiers à éditer qui n'ont pas leur propre carte FØRMAT. Si le niveau de qualification donné pour le ddnom est insuffisant, tous les fichiers portant ce nom seront édités suivant les spécifications de la carte FØRMAT.

- unité d'impression ou de perforation = nom de l'unité sur laquelle on désire éditer le fichier (PR1, PR2, ...)

  LØCAL : unité du site central (CIRCE) ayant les caractéristiques éventuellement exigés.

- nn = nombre d'exemplaires désirés (1 à 99). Attention, la demande de plusieurs exemplaires multiplie le nombre de lignes à éditer (ou de cartes à perforer) et peut donc nécessiter l'utilisation d'une carte MAIN s'il y a dépassement de l'attribution forfaitaire prévue par le système (2000 lignes et 200 cartes).

type : type de papier (ou de cartes) à utiliser. Nom de 1 à 8 caractères fixé par le CIRCE.

- SINGLE : espacement simple interligne
- DØUBLE : espacement double interligne
- ØVFL = ØFF : entraîne : par le saut de page en fin de page, l'option par défaut est : saut en fin de page. S'il n' y a pas de carte FORMAT, il n'y a pas de "saut de page automatique".
- TRAIN : type de chaîne d'impression à utiliser. La chaîne TN contient l'alphabet minuscule et majuscule.

Signalons que l'emploi des paramètres FORMS et TRAIN implique que les opérateurs doivent changer de formulaire (cartes ou papier) et/ou de chaîne d'impression sur l'imprimante. Il est donc conseillé de n'utiliser ces paramètres que de façon exceptionnelle après avoir contacter le service d'exploitation pour connaître les noms des formulaires et les caractéristiques des chaînes d'impression.

Exemples d'utilisation de la carte FØRMAT :

Exemple 1 :

```
//*FØRMATbPR,DDNAME=FT06F001,CØPIES=2,ØVFL=ØN
```

Liste du fichier FT06F001 en 2 exemplaires, avec saut en fin de page.

Exemple 2 :

```
//*FØRMATbPU,DDNAME=SYSPUNCH,CØPIES=2
```

Perforation en 2 exemplaires du programme compilé en binaire (DECK). La sortie du DECK sur cartes perforées nécessite également l'utilisation du paramètre PARM. FORT = (DECK) dans la carte EXEC (voir chapitre 6).

Exemple 3 :

```
//*FØRMATbPU,DDNAME=FT07F001,CØPIES=3
```

Perforation en 3 exemplaires du fichier FT07F001 sur cartes perforées.

Exemple 4 :

```
//*FØRMATbPU,DDNAME=CARTES,DEST=LØCAL
```

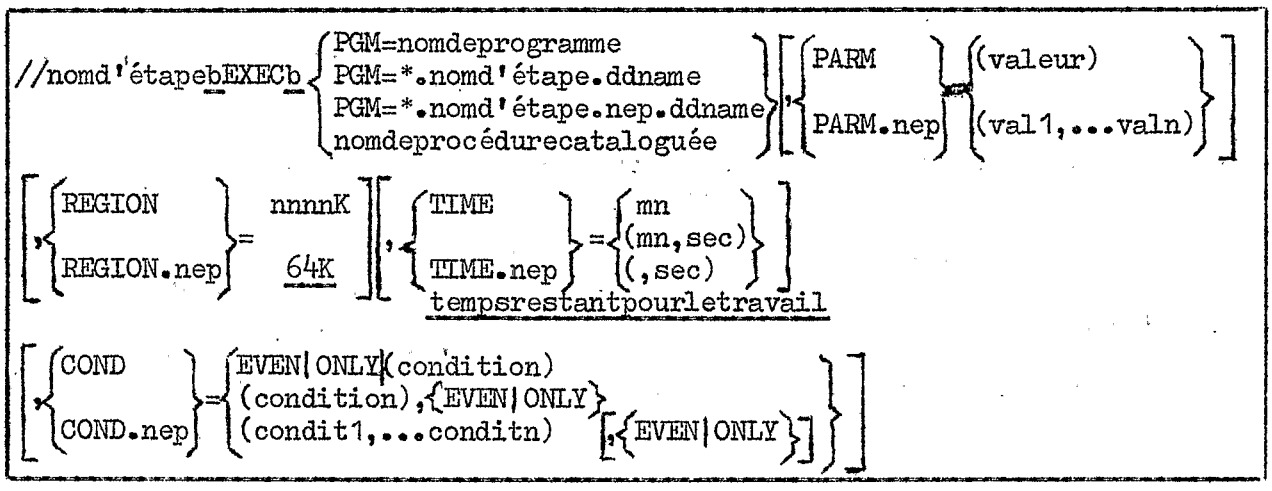
Perfore au CIRCE un fichier créé par un travail qui avait été transmis par un terminal (sans carte FØRMAT, le fichier serait reparti à destination du terminal émetteur).

La carte FORMAT est facultative. Quand elle existe, elle se place derrière la carte MAIN (ou derrière la carte JOB quand il n'y a pas de carte MAIN).

On peut utiliser plusieurs cartes FORMAT dans le même JOB.

6 - CARTE EXEC.

Forme générale



La carte EXEC identifie chaque nouvelle étape d'un travail et désigne selon les cas :

- le module chargeable (programme) à exécuter dans cette étape. (PGM =)
- la procédure cataloguée à utiliser.

L'apparition d'une carte EXEC marque donc la fin d'une étape et le début d'une autre.

nom d'étape : 1 à 8 caractères alphanumériques, le premier étant obligatoirement une lettre. Ce paramètre est optionnel mais recommandé (spécialement lorsque le travail comporte plusieurs étapes). Il permet en particulier de faire référence à l'étape considérée dans la suite du travail.

EXEC mot obligatoire précédé et suivi d'au moins une zone blanche.

PGM = paramètre obligatoire pour un programme exécutable.

nom de programme 1 à 8 caractères alphanumérique, le premier devant être une lettre. Il désigne un programme immédiatement exécutable résidant soit dans la bibliothèque système, soit dans une bibliothèque privée (fichier utilisateur). Dans ce cas, l'utilisateur doit fournir une carte DD définissant cette bibliothèque dont le nom (ddname) sera JOBLIB si la bibliothèque est utilisée par plusieurs étapes du travail et STEPLIB si elle n'est utilisée que dans l'étape en question.

Exemple :

```
.....................
//ET1bEXECbPGM=SØRT
.....................
//ET2bEXECbPGM=P301
//STEPLIBbDDbUNIT=3330-1,DISP=ØLD,VØL=SER=RES302
//bDSN=WRR1358.RØCHEMAR.HYDRØM
.....................
```

Exécution dans la première étape (ET1) du programme utilitaire SØRT (programme de TRI de la bibliothèque système),
exécution dans la deuxième étape (ET2) du programme P301 (module chargeable) appartenant à la bibliothèque HYDROM de l'utilisateur dont le sigle est WRR1358.ROCHEMAR et placée sur le disque RES302.

nom d'étape.ddname : Désigne un programme enregistré dans un fichier partitionné défini par une carte DD de nom "ddname" située dans une étape antérieure "nom d'étape" du travail en cours. Ce fichier est en général temporaire, et souvent le résultat d'une étape d'édition de liens, située dans le même travail.

exemple :

```
.....................
//LKEDbEXECbPGM=IEWL (programme éditeur de liens)
.....................
//SYSLMØDbDDbDSN=&GØSET(MAIN),...(fichier contenant le pro-
                                  gramme exécutable)
.....................
//GØbEXECbPGM=*.LKED.SYSLMØD (étape d'exécution)
```

nep : nom d'étape d'une procédure cataloguée (exemple : FØRT,LKED,GØ...)

nom de procédure cataloguée : 1 à 8 caractères alphanumériques, le premier étant obligatoirement une lettre. Rappelons qu'une procédure cataloguée regroupe plusieurs cartes de contrôle en une seule carte avec des caractères symboliques optionnels permettant le changement de certaines options définies dans la procédure.
Nous examinerons dans le chapitre 15 les procédures cataloguées utilisables sur l'installation 370 IBM du CIRCE.

exemple :

```
//ET1bEXECbFTXCLG
```

exécution dans l'étape "ET1" de la procédure cataloguée FTXCLG : compilation (C) d'un programme source écrit en FORTRAN niveau H étendu (FTX), puis appel de l'éditeur de liens (L) et exécution (G).

PARM : le paramètre PARM permet de préciser les options à appliquer durant l'exécution d'un programme.
Chaque programme système (compilateur, éditeur de liens, chargeur...) comporte un ensemble d'options particulières pour lesquelles des valeurs précises peuvent être choisies par l'utilisateur selon le programme utilisé. Si certaines de ces valeurs ne sont pas spécifiées par l'utilisateur, le programme à exécuter prend les valeurs par défaut.

Le paramètre PARM annule tous les paramètres par défaut de toutes les étapes de la procédure cataloguée et transmet de nouveaux paramètres dans l'étape nommée (ou de la 1ère étape si "nep" n'est pas précisé) les valeurs par défaut des étapes suivantes sont alors celles spécifiées à la génération du système.

On peut mettre les options du paramètre PARM soit entre parenthèses, soit entre apostrophes. Ces options sont des paramètres à mot clé pouvant donc apparaître dans n'importe quel ordre.

Exemple on écrira : PARM = 'LIST,SØURCE,NØDECK'
PARM = (SØURCE,NØDECK,LIST)

Ces 2 formes sont valides et identiques.

Cependant si le paramètre PARM comporte des cartes suite, la totalité des options doit être mise entre parenthèses, chaque option contenant un caractère spécial devant être mis entre apostrophes :

Exemple : 1ère carte PARM = (LIST,'LINECØUNT(50)',SØURCE,
suite 2ème carte NØDECK)

A titre d'exemple nous donnons dans le tableau 5 la liste des principales options modifiables du paramètre PARM pour le compilateur FØRTRAN H étendu. La valeur par défaut prise par le système est la valeur soulignée.

TABLEAU 5

OPTIONS	SIGNIFICATION
NAME=MAIN	nom du programme principal
BCD-EBCDIC	code de perforation du programme
ANSF-NØANSF	définit les bibliothèques FORTRAN utilisées.
MAP-NØMAP	implantation des variables en mémoire - tableau des noms et instructions labellées.
DECK-NØDECK	obtention sur cartes du "module-objet" correspondant au programme source compilé.
ØBJECT-NØØBJECT	sauvegarde sur disque du "module-objet" dans le fichier décrit par la carte DD SYSLIN
LIST-NØLIST	liste en langage assembleur du programme compilé.
SØURCE-NØSØURCE	liste des images de carte du programme-source.
XREF-NØXREF	références croisées : tableau indiquant à quelles instructions des variables et les labels sont employés.
LINECNT=60	nombre maximum de lignes par page pour le listing source
TABSIZE=220K	place maximum à allouer pour la compilation
ØPT=2	niveau d'optimisation
FØRMAT-NØFØRMAT	en plus du listing normal on obtient un listing "structuré".

Exemples d'utilisation du paramètre PARM :

exemple 1. Demande du module-objet ("programme binaire") sur carte :

```
.............
//bEXECbFTXCLG,PARM.FØRT=(DECK),REGIØN GØ=120K
```

exemple 2. Suppression de l'impression du programme source, des tableaux MAP (étape FØRT), LIST et XREF (étape LKED) :

```
//bEXECbFTXCLG,PARM.FØRT=(NØSØURCE,NØMAP),
//      PARM.LKED=(NØLIST,NØXREF)
```

REGION : encombrement maximal en mémoire centrale, exprimé en K octets (1 K = 1024) et prévu pour l'étape (ou toutes les étapes s'il s'agit d'une procédure cataloguée et que "nep" n'est pas précisé).

exemple 1 : //CØPIEbEXECbPGM=IEBGENER,REGIØN=90K

l'étape CØPIE d'exécution du programme utilitaire IEBGENER se verra attribuer une place mémoire de 90K.

exemple 2 : //bEXECbFTXCLG,REGIØN=130K

chacune des étapes de compilation, éditeur de liens et exécution se verra attribuer 130K.

exemple 3 : //bEXECbFTXCLG,REGIØN.GØ=130K

seule l'étape exécution (GØ) se verra attribuer 130K, les étapes de compilation (FØRT) et d'éditeur de liens (LKED) se verront attribuer les valeurs spécifiées par le système. (valeurs par défaut données ci-dessous).

Le paramètre REGIØN demande à être ajusté à sa valeur la plus basse possible, car il intervient lors du calcul de la priorité de prise en compte dans la file d'attente des travaux et surtout dans la facturation du travail (actuellement, pour les travaux passés au CIRCE, l'encombrement mémoire intervient par son carré dans la formule de facturation).

Au CIRCE les valeurs par défaut spécifiées par le système sont les suivantes pour la procédure FTXCLG :

- étape FØRT : 250K
- étape LKED : 142K
- étape GØ : 64K

* L'estimation du paramètre REGIØN peut être faite pour chaque étape d'après l'examen des passages précédents en consultant les fichiers SYSPRINT contenant les listes de compilation et la liste de l'éditeur de liens et le fichier SYSMSG.

Ainsi pour un programme FØRTRAN on a les indications suivantes :

(1) Compilation (FØRT).

- Fichier SYSMSG (messages en tête de listing). L'allocation demandée pour la compilation est donnée par le message IEF374I de l'étape FØRT.

exemple :

IEF374I STEP/ FORT / STOP 76069-1905 CPUOMIN 00.62SEC MAIN248K LCS OK

- Fichier SYSPRINT. Pour chaque programme ou sous programme compilé on a un message donnant la place mémoire non utilisée

exemple :

```
                    MAIN
****** END OF COMPILATION ******        108K BYTES OF CORE NOT USED

                 SOUS-PROGRAMME
****** END OF COMPILATION ******        172K BYTES OF CORE NOT USED
```

Dans cet exemple la place demandée était 248K (attribution par défaut) pour l'étape FORT. Pour le MAIN 108K n'ont pas été utilisés et 172K pour le sous programme. Donc d'une part 248 - 108 = 140K étaient nécessaires et 248 - 172 = 76K d'autre part. La valeur minimale à demander pour cette compilation est donc 140K (+ 10K pour tenir compte des fichiers utilisés pendant la compilation).

(2) Edition de liens (LKED).

- Fichier SYSMSG : l'allocation demandée pour l'éditeur de liens est donnée par le message IEF374I de l'étape LKED

exemple :

```
IEF374I STEP /LKED /STØP 76069.1905 CPU 0MIN00.33SEC MAIN136K LCS OK
```

- Fichier SYSPRINT : il fournit une estimation par défaut de la taille de la région nécessaire exprimée en hexadécimal dans le message "TOTAL LENGTH"

exemple :

```
                TOTAL LENGTH 7030
```

La table de conversion donnée en annexe 1 donne 28720 en décimal soit environ 29 K octets.
A ces 29K, il faut ajouter l'encombrement des buffers pour chaque fichier ouvert.
La région à demander devra être au minimum :

$$\text{REGION} = \text{TOTAL LENGTH}_{10} + 2\,(\text{BLKSIZE (FICH1)} + \ldots \text{BLKSIZE (FICHn)}).$$

2 étant (au CIRCE) la valeur par défaut du nombre de buffers attribués à chaque fichier. BLKSIZE = longueur du bloc (voir parag.7.2.7)

(3) Exécution (GØ).

Le fichier SYSMSG donne la taille de la région réellement utilisée, en tenant compte des fichiers ouverts, dans le message IEF374I de l'étape GØ :

exemple :

```
IEF374I STEP /GØ /STØP 76069.1905 CPU 0MIN00.67SEC MAIN46K LCS OK
```

* Si le programme est exécuté pour la première fois on pourra procéder de la façon suivante :

- Pour les étapes FØRT et LKED, on utilisera les valeurs par défaut

allouées par le système et très largement suffisantes dans la plupart des cas.

- Pour l'étape GØ on fera l'estimation approximative suivante :

  * l'encombrement du programme (MAIN + sous-programmes) sera estimé à partir du décompte des constantes, variables et surtout des tableaux (variables indicées) en tenant compte de leur longueur respective.
  On se rappellera en particulier que la longueur standard d'une variable entière, réelle ou logique est de 4 octets, et d'une variable complexe de 8 octets et que, en FØRTRAN, ces longueurs peuvent être éventuellement modifiées par des instructions de spécification implicite ou explicite (double précision, intégrer 2 ...). (voir Annexe 1 paragraphe 1.6.).

  En pratique, le plus souvent il suffira de faire la somme des variables, chaque tableau représentant un nombre de variables égal au produit des valeurs de ses dimensions, et de multiplier cette somme par 4/1000 (dans le cas de variables de longueur standard). Ce résultat sera par prudence multiplié par 1,5 pour tenir compte de la place nécessaire pour les constantes et les instructions. On obtiendra ainsi une valeur n1 en K octets.

  * A cette valeur n1 il convient d'ajouter l'encombrement des buffers pour les opérations d'entrée/sortie à partir des fichiers utilisés. Au CIRCE l'ouverture d'un fichier (lecture ou écriture) nécessite 2 buffers (valeur par défaut modifiable par le paramètre BUFNØ de la carte DD).
  Il faut donc prévoir pour chaque fichier une place mémoire égale à 2 fois la taille d'un enregistrement physique qui est donnée par le paramètre BLKSIZE de la carte DD correspondante (voir chapitre 7. 2. paramètre DCB de la carte DD)

  Il faut donc prévoir en plus :

  - pour les fichiers cartes (lecture ou perforation): moins de 1K octets
  - pour les fichiersd'impression (avec BLKSIZE = 1922) : 4K octets
  - pour les autres fichiers : $\sim \frac{2 * BLKSIZE}{1000}$ K octets.

  Soit n2 cet encombrement des buffers.

  + le paramètre REGIØN.GØ nécessaire pour ce premier passage sera pris égal à (n1 + n2) K. Par la suite il sera ajusté.

  Une estimation insuffisante du paramètre REGIØN.GØ entraine un arrêt de l'exécution et l'apparition dans le fichier SYSMSG d'un message d'erreur du type :

  CØMPLETIØN CØDE 804 (chargement non effectué)

  ou CØMPLETIØN CØDE 80A (chargement effectué mais ouverture des fichiers impossible).

CØND : Ce paramètre spécifie les conditions sous lesquelles l'étape doit être ou non exécutée, selon le résultat des étapes précédentes (code de retour).

Condition : { code, opérateur
code, opérateur, nom d'étape
code, opérateur, nom d'étape.nep }

avec

Code : nombre entier compris entre 0 et 4095. Il sera comparé au code de retour d'une ou plusieurs étapes précédentes.

Opérateur : {GT|GE|EQ|LT|LE|NE} (opérateur logique)

EVEN : exécution de l'étape même si une ou plusieurs étapes précédentes ont eu une fin anormale (ABEND par exemple)

ØNLY : exécution de l'étape seulement si une ou plusieurs étapes précédentes ont eu une fin anormale (ABEND par exemple)

Le "code" est comparé à l'aide de l' "opérateur" au code de retour issu de chaque étape précédente du travail.
Si "nom d'étape" est précisé, le code ne sera comparé qu'au code de retour de cette étape (cette étape n'appartenant pas à une procédure et n'en faisant pas intervenir).
Si "nom d'étape.nep" est précisé (nep = nom d'étape d'une procédure cataloguée) le code n'est comparé qu'au code de retour de l'étape "nep" de la procédure insérée à l'étape du travail "nom d'étape".

exemple :
//ETAP6bEXECbPGM=UBU,CØND=(EVEN,(4,GT,ETAP3),(12,LT,ETAP5.LKED))

l'étape ETAP6 sera exécutée dans les cas suivants :

- même si une ou plusieurs des étapes précédentes ont eu une fin anormale (ABEND)
- si le code retour de l'étape ETAP3 est > 4,
- si le code retour de l'étape LKED de la procédure insérée à l'étape ETAP5 est < 12.

Remarque importante concernant l'utilisation des paramètres PARM,REGIØN et CØND de la carte EXEC

Si plusieurs étapes sont touchées par les paramètres PARM.nep et/ou REGIØN.nep et/ou CØND.nep, ces paramètres doivent être données dans l'ordre d'occurrence des étapes qu'ils concernent.

exemple :
//ETAP1bEXECbFTXCLG,REGIØN.GO=200K,PARM.FØRT=(DECK)

est invalide car l'étape GØ suit l'étape FØRT. On doit écrire :

//ETAP1bEXECbFTXCLG,PARM.FØRT=(DECK),REGIØN.GO=200K

7 - CARTE DD.

La carte DD ou "DATA DEFINITION" identifie un data-set (fichier) c'est-à-dire un ensemble de données classées selon un ordre prescrit et décrit par des informations de contrôle auxquelles le système peut accéder.

Il doit y avoir une carte DD pour chaque fichier utilisé dans une étape de travail et l'ensemble des cartes DD d'une étape doivent suivre la carte EXEC indiquant le début de cette étape.

7.1. Formes restreintes de la carte DD :

- Nous examinerons tout d'abord 4 formes restreintes de la carte DD :

1. Fichier à éditer :

```
//ddnombDDbSYSØUT= { A }
                   { B }
                   { I }
```

2. Fichier à lire :

```
//ddnombDDb { * [,DLM=delimiteur] }
            { DATA              }
            { /*                }
```

3. Fichier fictif :

```
//ddnombDDbDUMMY
```

4. Fichier ultérieurement défini :

```
//ddnom1bDDbDDNAME=ddnom2 [,DCB=...]
```

ddnom - nom donné à l'instruction DD. 1 à 8 caractères alphanumériques, le premier étant obligatoirement une lettre.

- En CØBØL c'est le nom du fichier qui suit l'expression ASSIGN TØ exemple :

  SELECT FCAR ASSIGN TO UR - S - SYS005

  cette instruction donnera lieu à l'utilisation du ddnom : SYS005

- Pour les programmes de traitement fournis par IBM (compilateurs, linkage editor, utilitaires ...) le ddnom est imposé.
  Ainsi, pour la procédure cataloguée FORTRAN H étendue (FTXCLG) les noms des cartes DD inclues dans la procédure sont :

  lecture des données sur cartes : FT05F001 et SYSIN
  impression des résultats : FT06F001
  perforation des cartes : FT07F001

- Chaque ddnom doit être unique dans une même étape.

- Le ddnom doit être omis si le fichier décrit par cette carte est à "concaténer" au précédent.
La "concaténation" consiste à relier logiquement plusieurs fichiers en entrée, définis par des cartes DD successives. Ils sont alors lus comme s'ils ne constituaient qu'un seul fichier.

- Le ddnom doit être composé en "nom d'étape de procédure.ddnom" pour pouvoir se referer à une instruction DD précise lorsqu'il en existe de même nom dans des étapes différentes :

exemple dans une étape d'exécution d'un programme FØRTRAN :

//GØ.FTxxFyybDDbparamètres

xx : représente le numéro logique du fichier utilisé pour identifier le fichier dans le programme.

yy : représente le numéro de séquence du fichier que l'on veut traiter pour distinguer les utilisations successives en lecture ou écriture.

- Certain ddnom ont une signification particulière :

JØBLIB : définit au niveau du JØB une bibliothèque contenant un programme "load module" à appeler.
La carte //JØBLIBbDD... doit être placée immédiatement après la carte JØB. Elle ne doit pas apparaître dans une procédure cataloguée.

STEPLIB : définit au niveau du STEP (étape) une bibliothèque contenant un programme "load module" à appeler. La carte //STEPLIBbDD ... doit être insérée parmi les cartes DD de l'étape. Elle peut apparaître dans une procédure cataloguée.

SYSABEND : définit un fichier dans lequel sera écrit un dump des zones de mémoire contenant : le noyau du système et la région allouée au programme de l'utilisateur.

SYSUDUMP : définit un fichier dans lequel sera écrit un dump de la région de mémoire allouée au programme de l'utilisateur

exemple : //GØ.SYSUDUMPbDDbSYSØUT=A

SYSØUT : - SYSØUT =A définit un fichier temporaire à imprimer dès la fin du travail.

Au CIRCE les fichiers d'impression définis par le paramètre SYSØUT =A peuvent avoir des enregistrements logiques groupés dans des blocs physiques de longueur inférieure ou égaleà 2020 octets.
On a donc intérêt à augmenter le facteur de blocage des fichiers d'impression afin de diminuer le nombre d'opérations d'écriture effectuées dessus.
On codera par exemple le paramètre DCB (voir paragr.7.2.7.) de la façon suivante :

//FT08FOO1bDDbSYSØUT=A,DCB=(RECFM=VBA,LRECL=137,BLKSIZE=1922)

- SYSØUT = B définit un fichier temporaire à perforer dès la fin du travail.

- SYSØUT = I définit un fichier temporaire à perforer et à interpréter dès la fin du travail. Son utilisation est impossible pour une sortie de cartes "objet" d'une compilation (DECK en binaire).

  Au CIRCE, il n'est pas possible de bloquer les fichiers définis par SYSØUT = B ou SYSØUT = I.

* : Les données qui constituent ce fichier sont sur des cartes perforées qui suivent cette carte DD et cela jusqu'à la prochaine carte de contrôle caractérisée par // en colonnes 1 et 2 jusqu'à une carte délimiteur /*.

exemple : //FØRT.SYSINbDDb*
programme source FØRTRAN à compiler sur cartes.
/*

DATA : est utilisé au lieu de * si les données qui composent ce fichier contiennent des instructions du langage de contrôle.
Les données ne doivent cependant pas contenir les caractères /* en colonnes 1 et 2 (ou le délimiteur indiqué par le paramètre DLM).

Au CIRCE, il n'est pas possible de bloquer les fichiers définis par DATA ou *.

DLM : permet d'indiquer un délimiteur de 2 caractères quelconques indiquant la fin du fichier. Ce délimiteur devra être en colonnes 1 et 2. Si le paramètre DLM est omis, /* sera pris comme délimiteur.

délimiteur : 2 caractères EBCDIC.

exemple : //ENTCARbDDbDATA,DLM=FN
cartes données y compris /*
FN

DUMMY : les opérations d'entrée/sortie, d'allocation d'espace sur mémoire à accès direct, ... portant sur ce fichier seront ignorées.

exemple d'utilisation du paramètre DUMMY :

Un programme FØRTRAN contient des instructions d'écriture sur cartes du type :

```
      WRITE(7,1000) ....
1000 FØRMAT(.....)
```

On supprimera l'effet de ces instructions (perforation de cartes en sortie) en associant une carte DD de la forme :

//GØ.FTO7FOO1bDDbDUMMY

DDNAME = ddnom2 : permet de suspendre la définition du fichier : le système attendra de trouver plus loin dans le flot d'entrée ou la procédure cataloguée utilisée, une carte DD portant le nom ddnom2.
Les caractéristiques du fichier décrit dans ddnom2 seront alors utilisées pour définir le fichier cité dans ddnom1.

7.2. Forme générale de la carte DD :

```
//ddnombDDb [DSNAME=...] [,UNIT=...] [,VØLUME=...] [,LABEL=...]
            [,DISP=... ] [,DCB= ...] [,SPACE= ...]
```

La carte DD peut comporter jusqu'à 7 paramètres à mots-clés.

Parmi ces paramètres 2 peuvent être abrégés : DSNAME en DSN et VØLUME en VØL.

Ces paramètres ne sont pas toujours utilisés. On retiendra les combinaisons générales suivantes :

Bandes magnétiques :

Paramètres	toujours employés	éventuellement employés	jamais employés
création fichier permanent	DSN, UNIT	LABEL, DISP, DCB, VØL	SPACE
utilisation fichier permanent	DSN, UNIT, VØL	LABEL, DISP, DCB	SPACE

- Unités à accès direct (disques)

Paramètres	toujours employés	éventuellement employés	inutiles
création fichier permanent	DSN, UNIT, VØL, SPACE	DISP, DCB, LABEL	
utilisation fichier permanent	DSN, UNIT, VØL	DISP, SPACE, LABEL	DCB
création fichier temporaire	UNIT, SPACE	DISP, DCB, LABEL	VØL, DSN
utilisation fichier temporaire	DSN, UNIT	DISP, SPACE, LABEL	VØL, DCB

7.2.1. DSNAME ou DSN : Ce paramètre spécifie le nom d'un fichier. Sa forme générale est la suivante :

DSNAME DSN	=	nom | nom (nom de membre) | && nom | && nom (nom de membre) *.ddnom | *.nom d'étape.ddnom *.nom d'étape.nom d'étape de procédure.ddnom

nom

- fichier sur bande : 1 à 17 caractères alphanumériques, le premier étant obligatoirement une lettre.
- fichier sur disque : Au CIRCE le nom d'un fichier permanent sur disque doit être de la forme :

SIGLENUM.NOMUTILISATEUR.NOMFICHIER.ect...

- NOMFICHIER.ect... : au choix de l'utilisateur (noms simples facultatifs) ⩽ 8 caract.
- NOMUTILISATEUR : Nom de l'utilisateur ⩽ 8 caract.
- SIGLENUM : Sigle de l'utilisateur ⩽ 8 caract.

Au total le DSN ne doit pas dépasser 44 caractères (y compris les .). Il y a donc 22 niveaux de qualifications possibles.

exemple : DSN = WRR1358.RØCHEMAR.HYDROM.JAUG.ØRSTOM

(WRR1358.RØCHEMAR : obligatoire)

Pour un fichier partitionné, on aura :

DSN = SIGLENUM.NOMUTILISATEUR.NOMFICHIER (nom du membre)

Le nom du membre étant composé de 8 caractères alphanumériques au maximum, le premier étant alphabétique.

exemple : DSN = WRR1358.RØCHEMAR.HYDRØM(P301)

- Si && précède le non, le fichier est temporaire. Si un seul & précède le nom, le nom du fichier est modifiable (paramètre symbolique). S'il n'est pas modifié, le fichier est considéré comme temporaire (nom conservé en fin de JØB)

  exemple : DSN = &PLU

- Si *. précède le nom, on définit le DSN du fichier dans une carte DD d'une autre étape dont on donne la référence en la précisant suffisamment pour qu'elle ne soit pas ambigüe.

  exemple :

```
//ET1bEXECbPGM=PRØG1
//DD1bDDbUNIT=3330,SPACE=(TRK,10),...
/  I2bEXECbPGM=PRØG2
//DD2bDDbDSN=*.ET1.DD1,...
```

7.2.2. UNIT Ce paramètre permet de désigner l'unité physique utilisée (disque, bande ...)

Forme générale :

```
UNIT = { nom générique
       { (nom générique,n)
       { AFF = ddnom
```

nom générique (CIRCE) :

BDE7	pour les bandes 7 canaux
2400-4	pour les bandes 9 canaux avec densité 800 bpi
BD16	pour les bandes 9 canaux avec densité 1600 bpi
BD60	pour les bandes 9 canaux avec densité 6250 bpi
3330	pour les disques 3330 modèle 1
3330-1	pour les disques 3330 modèle 11

Actuellement au CIRCE 6 unités de disques 3330 sont du type 11 : BAN101, PP3330, RES301, RES302, RES303 et RES304. Tous les autres disques sont des 3330 modèle 1.

n : nombre d'unités utilisées pour monter en parallèle les différents volumes d'un même fichier dans le cas d'un fichier - multivolumes.

AFF : demande que l'unité employée pour accéder à ce fichier soit la même que celle utilisée pour le fichier défini dans la carte DD de nom ddnom. Cette carte doit apparaître dans la même étape, et avant la carte DD contenant le sous paramètre AFF.

7.2.3. VØLUME ou VØL Ce paramètre donne les caractéristiques et l'identification du ou des volumes utilisés.

Forme générale :

```
{VØLUME} = { SER = numéro de volume
{VØL   } = { ([,RETAIN] [,rangvol] [,nombrevol] [,SER=(nnm1,...numn)
           {                                     ,REF=dsname
           {                                     ,REF=*.ddnom
           {                                     ,REF=*.nomd'étape.ddnom
           {                                     ,REF=*.nomd'étape.nom
           {                                          d'étape de procé-
           {                                          dure.ddnom
```

SER = numéro de volume (SERIAL NUMBER). Il s'agit du numéro de rangement du volume composé de 6 caractères alphanumériques.

Au CIRCE pour les bandes, il s'agit d'un nombre attribué par le service d'exploitation au laboratoire utilisateur.

exemple : VØL = SER = 103278

103278 est le numéro de rangement attribué par le CIRCE à une bande magnétique.

Pour les disques les paramètres SER des volumes utilisables au CIRCE pour la création de fichier permanents sont les suivants : (voir 1ère partie parag. 3.3.)

disques en SETUP (montage à la demande) : SET001, SET002... ,SET011, SET012
disques privés en ligne (résident) : RES301, RES302, RES303 et RES304
disques privés provisoire : PP3330.

exemple : VØL = SER = RES301

RETAIN : Normalement un volume est démonté après sa dernière utilisation dans l'étape. Si une étape ultérieure du même travail doit utiliser un fichier sur le même volume, il convient d'utiliser l'option RETAIN pour éviter un démontage et un remontage de volume (manipulations qui sont facturées).

Avec l'option RETAIN le volume reste monté en fin d'étape. Il sera démonté en fin d'étape suivante (ou de travail).

exemple : VØL = (,RETAIN, SER = 103278)

Rangvol : rang du volume à monter parmi tous les volumes d'un fichier multi-volumes.

Nombre vol : nombre de volumes utilisés par un fichier multi-volumes.

REF : le nom du volume doit être retrouvé dans une autre carte DD dans la même étape ou dans une étape antérieure.
La référence doit être suffisamment précise pour ne pas être ambiguë.

L'option REF est particulièrement intéressante quand on désire créer plusieurs fichiers mis séquentiellement sur une même bande magnétique dont on ne connaît pas le numéro de volume (numéro que sera attribué par le CIRCE) :

exemple :

```
//ET1bEXEC...
//DD1bDDbDSN=FIC1,UNIT=BD16,VØL=(,RETAIN),LABEL=(1,SL),...
.............
//ET2bEXEC...
//DD2bDDbDSN=FIC2,UNIT=BD16,VØL=REF=*.ET1.DD1,LABEL=(2,SL),...
```

7.2.4. LABEL : Ce paramètre donne le label d'identification du volume et éventuellement les conditions d'accès (lecture et / ou écriture).

Forme générale :

$$\text{LABEL} = \left(\begin{matrix}\text{rang} \\ \underline{1}\end{matrix} \quad \left[\begin{matrix},\text{NL} \\ ,\text{BLP} \\ ,\underline{\text{SL}}\end{matrix}\right] \quad \left[\begin{matrix},\text{IN} \\ ,\text{ØUT} \\ ,\text{INØUT}\end{matrix}\right]\right)$$

La signification des sous-paramètres est la suivante :

rang : spécifie la position relative du fichier sur le volume (à partir du début du volume). Nombre de 1 à 4 chiffres. La valeur par défaut est 1.

NL : (no label) spécifie qu'il y n'y a pas de label sur ce volume.

BLP : (bypasslabel processing) le système ne doit pas traiter le label dont l'examen est à la charge de l'utilisateur.

SL : (standard label IBM) spécifie qu'il y a un label conforme aux normes ØS, traité par le système.

IN : les opérations autorisées sur ce fichier sont uniquement la lecture.

ØUT : les opérations autorisées sur ce fichier sont uniquement l'écriture.

INØUT : les opérations autorisées sur ce fichier sont la lecture et l'écriture.

Remarques importantes pour l'utilisation du paramètre LABEL :

- Tous ces sous-paramètres sont positionnels, s'ils sont omis, on doit donc coder obligatoirement une virgule à leur place.
- Si l'on veut spécifier uniquement le numéro de séquence, on peut omettre les parenthèses :

  exemple : LABEL = 3

- Quand on emploie l'option BLP dans le second sous-paramètre pour ne pas tenir compte des labels, il faudra faire attention à bien calculer le numéro de séquence (rang) car le système considère dans ce cas les labels comme des fichiers compris entre 2 tapemarks et il ne faut pas oublier qu'il existe un label de tête et un label de queue pour chaque fichier de données.

Ainsi pour se positionner au premier fichier, il faudra coder :

LABEL = (2, BLP)

Pour atteindre le second fichier :

LABEL = (5, BLP)

Et pour atteindre le 3ème fichier :

LABEL = (8, BLP) et ainsi de suite.

- Au CIRCE, il est fortement déconseillé de créér une bande no label et pour les disques seule l'option standard label est autorisée.

  D'autre part les sous-paramètres IN,ØUT et INØUT ne doivent être utilisés que pour des fichiers séquentiels ou un membre d'un fichier partitionné.

- La création d'un membre de fichier partitionné avec un programme FØRTRAN exige la spécification du paramètre LABEL = (,,,ØUT) dans la carte DD décrivant le fichier.

  De même la carte DD décrivant le fichier partitionné à relire (avec un programme FØRTRAN) doit obligatoirement contenir le paramètre LABEL = (,,,IN).

  Si on oublie de spécifier le paramètre LABEL = (,,,IN) il y a apparition dans le fichier SYSMSG à l'étape traitant le fichier d 'un message de la forme :

IEC2251 04, NØMJ, GØ, FT10F001, 254, PP3330, WRR1358.RØCHEMAR.FICH

- 04 : code d'erreur
- NØMJ : nom du travail
- GØ : nom d'étape
- FT10F001 : nom de la carte DD
- 254 : adresse physique de l'unité.
- PP3330 : nom du volume sur lequel réside le fichier
- WRR1358.RØCHEMAR.FICH : nom du fichier

7.2.5. DISP. Ce paramètre donne l'état du fichier au début de l'étape et les dispositions à prendre en fin d'étape.

Forme générale :

DISP = ([SHR / NEW / ØLD / MØD] [,DELETE / ,KEEP / ,PASS / ,CATLG / ,UNCATLG] [,DELETE / ,KEEP / ,CATLG / ,UNCATLG])

La fonction des 3 sous paramètres est la suivante :

1er sous paramètre : donne l'état du fichier lorsque l'étape est initialisée

2ème sous paramètre : indique ce que doit devenir le fichier à la fin de l'étape si elle se termine normalement.

3ème sous paramètre : indique ce que doit devenir le fichier à la fin de l'étape si elle se termine anormalement.

La signification des valeurs conventionnelles de ces sous paramètres est la suivante :

NEW : le fichier est créé par l'étape en cours. C'est la valeur par défaut du 1er paramètre.

Lorsque ce paramètre est codé, il entraîne l'exécution des opérations suivantes :

- sur bande : création du label suivant les options données dans le paramètre LABEL.

- sur disque : 1. Vérification dans la VTØC du disque de l'existence ou non du DSN du nouveau fichier.
Si le DSN existe déjà apparition du message suivant :

"DUPLICATE NAME ØN THE VØLUME"

2. Mise à jour de la VTØC avec le nouveau DSN

3. Vérification de la disponibilité d'espace suffisant pour le fichier selon les indications du paramètre SPACE.
Si la place disponible est insuffisante apparition du message suivant :

"INSUFFICIENT SPACE ØN STØRAGE"

SHR : (Shared). Le fichier existait avant l'étape. Il peut être consulté par plusieurs utilisateurs. En particulier les bibliothèques doivent être utilisées en SHR, sauf dans le cas de mise à jour.

ØLD : Le fichier existait avant l'étape. L'utilisation par le JØB de ce fichier est exclusive : un autre utilisateur ne peut avoir accès à ce fichier pendant tout le JØB.

Si ce paramètre est codé le système effectue les opérations suivantes :

sur bande : Si LABEL = SL, l'Ø.S. vérifie le DSN.
Au cas où cette vérification n'est pas satisfaisante, il y a arrêt de l'exécution et apparition du message :

"CØMPLETIØN CØDE 613"

sur disque : l' Ø.S. vérifie que le DSN figure bien dans la VTØC. Si non apparition du message :

"DATA SET NØT FØUND"

L'ØS interdit également l'accès de ce fichier durant toute l'étape à un autre utilisateur en mémoire, en même temps que lui et utilisant ce fichier.

MØD : Le fichier existait déjà avant l'étape au cours de laquelle il sera mis à jour.
Le mécanisme de lecture est alors positionné après le dernier enregistrement du fichier. Le contrôle de ce fichier est exclusif : il ne peut y avoir d'autres utilisateurs.
Si le fichier est vide, le sous paramètre MØD est alors équivalent à NEW.

Sur bande : le fichier sera augmenté d'un certain nombre d'enregistrements.

Sur disque :

- si le fichier est séquentiel, il y a rajout d'enregistrements au fichier existant.
- si le fichier est partitionné le sous paramètre MØD est alors équivalent à ØLD et un membre va être rajouté dans le directory pour inscrire ce nouveau membre. Si cela est impossible pour insuffisance de place apparition du message :

"INSUFFICIENT SPACE IN THE DIRECTØRY"

- dans ces deux cas on peut demander une nouvelle allocation d'espace supplémentaire par le paramètre SPACE.

DELETE : Le fichier sera supprimé en fin d'étape. C'est la valeur par défaut du second paramètre lorsque DISP n'est pas codé.
S'il s'agit d'un fichier sur disque, il y a suppression du nom du fichier de la VTØC et la place utilisée par ce DSN est rendue disponible.

S'il s'agit d'un fichier sur bande, DELETE n'a pour effet que de positionner la bande au point de chargement.

KEEP : Le fichier est conservé pour une utilisation ultérieure, dans un autre JØB.

C'est la valeur par défaut du second paramètre si le fichier existait déjà en début d'étape (c'est-à-dire si le premier paramètre codé est ØLD).

Pour les bandes ce paramètre entraîne le démontage en fin d'étape (sauf si on a codé RETAIN dans le paramètre VØL).

PASS : Le fichier est conservé en fin d'étape pour un usage ultérieur dans une / des étapes du même JØB.

La décision de conservation ou de suppression est donc remise à une étape ultérieure du travail.

Attention l'utilisation du paramètre PASS se fait selon des règles bien précises :

Si un fichier est créé au cours d'une étape d'un travail (avec l'option PASS codé comme 2ème sous paramètre de DISP) et passé à une étape ultérieure de ce même travail, le système ne conserve pas ce fichier si on code les paramètres UNIT et VØL dans la carte DD de l'étape où il est réutilisé (même si la disposition finale est KEEP).

exemple :

```
//carte JØB
//ET1bEXECbPGM=TRUC
//DD1bDDbDSN=WRR1358.RØCHEMAR.FICH1,UNIT=3330-1,
//    VØL=SER=PP3330,SPACE=(TRK,(2,1)),DISP=(NEW,PASS),
//    DCB=(RECFM=FB,BLKSIZE=800,LRECL=80)
.....
//ET2bEXECbPGM=CHØSE
//DD2bDDbDSN=WRR1358.RØCHEMAR.FICH1,UNIT=3330-1,
//    VØL=SER=PP3330,DISP=(OLD,KEEP)
.....
//
```

Le fichier WRR1358.RØCHEMAR.FICH1 n'existe plus à la fin du travail.

Pour le conserver, il faut ne coder que les paramètres DSN et DISP dans les cartes DD réutilisant ce fichier.

Il faut donc écrire :

```
//carte JØB
//ET1bEXECbPGM=TRUC
//DD1bDDbDSN=WRR1358.RØCHEMAR.FICH1,UNIT=3330-1,
//    VØL=SER=PP3330,SPACE=(TRK,(2,1)),DISP=(NEW,PASS)
//    DCB=(RECFM=FB,BLKSIZE=800,LRECL=80)
.....
//ET2bEXECbPGM=CHØSE
//DD2bDDbDSN=WRR1358.RØCHEMAR.FICH1,DISP=(ØLD,KEEP)
.....
//
```

CATLG : Le fichier sera catalogué en fin d'étape, c'est-à-dire que les valeurs des paramètres DSN, UNIT, VØL = SER, seront inscrites dans le catalogue du disque système.
Le catalogue des fichiers utilisateurs du CIRCE réside sur le disque RES300.

Pour l'utilisation d'un fichier catalogué, il suffit dans la carte DD de coder le DSN et DISP = ØLD.

UNCATLG : Le nom du fichier sera enlevé au catalogue du système. Ceci ne veut pas dire qu'il enlève en plus le DSN de la VTØC. Le fichier existe toujours mais son utilisation ultérieure demande tous les paramètres de la carte DD.

Récapitulatif des options du paramètre DISP. (l'option par défaut est soulignée) :

DISP = (NEW,	DELETE,	DELETE)
DISP = (NEW,	PASS,	DELETE)
DISP = (NEW,	KEEP,	KEEP)
DISP = (NEW,	CATLG,	CATLG)
DISP = (ØLD,	DELETE,	DELETE)
DISP = (ØLD,	PASS,	PASS)
DISP = (ØLD,	KEEP,	KEEP)
DISP = (ØLD,	CATLG,	KEEP)

7.2.6. SPACE Ce paramètre permet d'indiquer la quantité et les caractéristiques de l'espace attribué sur mémoire à accès direct.

Forme générale :

SPACE = ({ TRK / CYL / blocs }, (quantité [,incrément] [,directory / ,index]) [,RLSE])

Ce paramètre n'est à utiliser que pour les fichiers sur mémoire à accès direct (disques) à leur création ou à leur extension (DISP=MØD).

La signification des sous paramètres est la suivante :

TRK : track (=piste). L'allocation demandée est spécifiée en nombre de pistes.

CYL : Cylindre. L'allocation demandée est spécifiée en nombre de cylindres.

blocs : longueur du bloc en octets. Il s'agit de la longueur moyenne prévue pour les enregistrements physiques exprimée en octets.

Les performances sont meilleures si on utilise CYL ou TRK. L'allocation par bloc est déconseillée. L'allocation par TRK est fortement recommandée (disponibilité plus grande).

Rappelons que pour une unité de disque 3330 la capacité d'une piste est de 13030 octets et qu'un cylindre comprend 19 pistes. (Voir 1re partie. chapitre 3. parag.5. et tableau 3).

quantité : Ce sous paramètre spécifie l'allocation primaire en nombre d'unités choisies (TRK, CYL ou blocs).

exemple : SPACE = (CYL, 20)

L'utilisateur demande 20 cylindres qui seront réservés avant que s'exécute le programme de création :

Remarque : Lorsqu'on demande une quantité dépassant la capacité d'un disque (par exemple dépassant 404 cylindres pour un disque 3330-1) on provoque une attente du système afin de trouver l'espace demandée et le montage inutile de volumes alors que la requête ne peut être satisfaite.

incrément : Ce sous paramètre spécifie l'allocation secondaire que l'on désire si nécessaire

Ce sous paramètre est positionnel mais optionnel.

Cette allocation secondaire sera allouée dans la mesure de la place disponible et ceci à concurence de 15 fois.

exemple : SPACE = (CYL, (20,2))

L'utilisateur demande 20 cylindres au départ (allocation primaire), auxquels s'ajouteront 2 cylindres en cours d'exécution en cas de saturation des 20 cylindres initiaux, puis encore éventuellement 2 cylindres supplémentaires et ainsi de suite jusqu'à 15 fois.

directory, index : Ce quatrième sous-paramètre donne l'amplitude d'une zone incluse dans l'allocation primaire et particulière soit à l'organisation partitionnée : directory, soit à l'organisation séquentielle indexée : index.

* Fichier partitionné :

La zone directory (ou répertoire) est réservée au début de l'allocation primaire et se compose d'enregistrements d'une longueur fixe de 256 octets. Dans le 4ème sous paramètre du SPACE on indique combien de ces enregistrements sont nécessaires pour le directory. (Voir figure 6).

exemple 1 : SPACE = (CYL, (20, 2, 10))

Le fichier partitionné aura une zone primaire de 20 cylindres plus éventuellement 1 ou plusieurs fois 2 cylindres (jusqu'à 15 fois) et une zone "directory" (ou répertoire) de 10 enregistrements de 256 octets au début du premier des 20 cylindres.

exemple 2 : SPACE = (TRK, (500,,5))

Dans ce cas il n'y a qu'une seule allocation primaire de 500 pistes pour un fichier, et le directory comprendra 5 enregistrements de 256 octets en tête de la région de 500 pistes.

* En séquentiel indexé le 4ème sous paramètre est utilisé pour indiquer le nombre d'unités choisies (dans le 1er sous paramètre) à allouer pour la zone INDEX.

exemple :

```
//DD1bDDbDSN=WRR1358.RØCHEMAR.FICH,UNIT=3330-1,
//   VØL=SER=PP3330,DCB=DSORG=IS,SPACE=(CYL,(20,2,1) ),
//   DISP=(,CATLG)
```

Le fichier défini dans la carte DD1 occupera une place de 20 cylindres (plus éventuellement 1 à 15 fois 2 cylindres) et une zone d'un cylindre réservé pour la zone INDEX.

Remarque importante : l'OS fait la différence entre directory et index au vu du DCB.

Si DSØRG = IS ou ISU il s'agit d'index, si non il s'agit de directory.

RLSE : (Release) Restitution de place allouée. Ce cinquième sous-paramètre indique s'il est codé que l'on restitue la place allouée et non utilisée à la fin de l'étape.

exemple : SPACE = (TRK, (50,2) RLSE)

Il est évident qu'il ne faut pas coder le sous paramètre RLSE lorsque l'on a l'intention d'augmenter la longueur du fichier par la suite dans un autre travail (DISP = MØD)

7.2.7. DCB (Data Control Block) Ce paramètre définit les caractéristiques des enregistrements d'un fichier.

Forme générale :

```
DCB = { (liste d'attributs)
      { ( { dsname                    } [,liste d'attributs] ) }
      { ( { *.nom d'étape.ddnom       }                      ) }
```

Les attributs normaux du DCB sont :

BLKSIZE : longueur maximale d'un bloc
BUFNØ : nombre de buffers à utiliser pour les opérations d'entrée/sortie.
DEN : densité d'enregistrement de l'information sur bande magnétique.

DSØRG : organisation du fichier
LRECL : longueur réelle ou maximale d'un enregistrement logique
RECFM : format et caractéristiques des enregistrements
TRTCH : mode d'écriture sur les bandes 7 pistes.

dsname
*.nom d'étape.ddnom } donne la référence d'un fichier ou d'une carte DD pour lequel les attributs du DCB sont définis et doivent être repris.

Examinons les attributs du paramètre DCB :

BLKSIZE : (Bloc-size) spécifie en octets la longueur maximale du bloc (enregistrement physique).

- La longueur maximale que l'on peut coder est 32760 octets sur bande et 13030 octets sur les disques 3330 sauf si on utilise l'option T dans le paramètre RECFM (auquel cas on peut coder jusqu'à 32760 octets).

- La longueur minimale que l'on peut coder est 18 octets.

- Si RECFM = F le BLKSIZE doit être $\geqslant$ longueur de l'enregistrement logique.

- Si RECFM = FB, le BLKSIZE doit être un multiple exact de la longueur de l'enregistrement logique.

- Si RECFM = V le BLKSIZE doit être $\geqslant$ à la longueur de l'enregistrement logique + 4 octets. (pour le compteur).

- Si RECFM = VB, le BLKSIZE doit être = (n fois le LRECL + 4 octets (pour le compteur)

- Si on code BLKSIZE dans le DCB d'une carte DD pour un fichier déjà existant, ce BLKSIZE recouvre le BLKSIZE écrit dans le label du fichier.

BUFNØ : Spécifie le nombre de buffers à assigner au bloc de contrôle du fichier. Le nombre maximum est 255, la valeur par défaut au CIRCE est 2.

DEN : Spécifie la densité d'enregistrement pour une bande magnétique en nombre de bits par inch (bpi).
DEN prend les valeurs 1, 2, 3 ou 4.

- bande 7 canaux : DEN = 1 pour 556 bpi, 2 pour 800 bpi. par défaut : 2

- bande 9 canaux : DEN = 2 pour 800 bpi, 3 pour 1600 bpi.
DEN = 4 pour 6250 bpi, par défaut DEN = 3

DSØRG : définit l'organisation du fichier. On peut coder les valeurs suivantes :

DA : accès direct
IS : séquentiel indexé,

PØ : fichier partitionné
PS : fichier séquentiel.

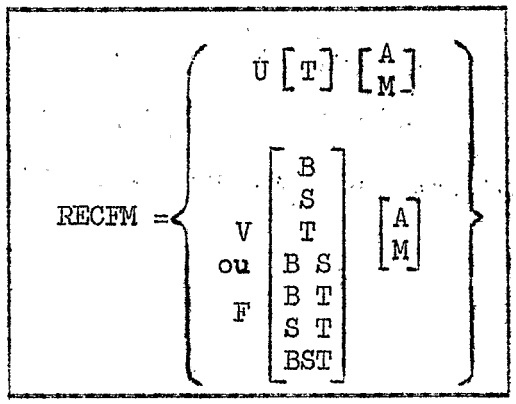
LRECL : longueur réelle ou maximale d'un enregistrement logique, en octets. (y compris le compteur).
Ce sous paramètre est obligatoire pour les enregistrements de longueur fixe et de longueur variable : dans ce cas la longueur indiquée est celle de l'enregistrement le plus grand.
LRECL ne doit pas dépasser la longueur du BLKSIZE sauf lorsqu'il s'agit d'enregistrement en "variable spannée" (RECFM = VS)

Donc :

- Si RECFM = F ou FB, LRECL doit être égal à la longueur de l'enregistrement logique
- Si RECFM = V ou VB LRECL doit être égal à la longueur du plus grand enregistrement + 4 (pour le compteur)
- Si RECFM = U LRECL doit être omis
- Si RECFM = VS ou VBS et si l'enregistrement logique est > 32756 il faut spécifier LRECL = X.

RECFM : spécifie le format et les caractéristiques des enregistrements d'un fichier.
Rappelons que la forme des enregistrements a été examinée dans le paragraphe 1.5. de la deuxième partie.

Les options possibles pour coder le RECFM sont les suivantes : (avec les méthodes d'accès BSAM et QSAM)



Avec :

A : l'enregistrement contient un caractère de saut ASA
exemple : les fichiers de type SYSØUT = A pour l'impression ont un attribut RECFM du paramètre DCB qui est codé :

RECF = VBA

B : les enregistrements sont bloqués

F : les enregistrements ont une longueur fixe

M : les enregistrements contiennent un caractère de saut machine,

S : indique des enregistrements étendus, c'est-à-dire contenus sur plusieurs blocs ("SPANNED"). Ce paramètre ne peut être spécifié que pour des enregistrements de longueur variable.

T : les enregistrements peuvent avoir une longueur supérieure à une piste (dispositif TRACK OVERFLOW) (voir paramètre BLKSIZE)

U : indique des enregistrements de longueur indéfinie.

V : indique des enregistrements de longueur variable.

TRTCH : Indique la technique d'enregistrement pour les bandes 7 pistes.

$$\text{TRTCH} : \left\{ \begin{array}{c} C \\ E \\ T \\ E\ T \end{array} \right\}$$

C : pour l'utilisation du convertisseur

E : indique une parité paire (EVEN parity)

exemple : TRTCH = E

ET : pour l'utilisation du traducteur avec parité paire. Traduction code BCD en EBCDIC ou inversement.

T : pour l'utilisation du traducteur avec parité impaire

On trouvera en annexe 3 un récapitulatif descriptif des principaux paramètres de la carte DD.

8 - CARTES DATASET et ENDDATASET.

Ces cartes permettent de définir le début et la fin d'un fichier qui sera utilisé par une étape du travail.

Forme générale de la carte DATASET :

```
//*DATASETbDDNAME=ddnom  [,J = {YES | NO}]
```

Forme générale de la carte ENDDATASET :

```
//*ENDDATASET
```

La signification des paramètres est la suivante :

ddnom : nom de la carte DD utilisée pour se référer au fichier défini par la carte DATASET.

J : précise comment se termine le fichier qui suit la carte DATASET

J = YES précise que le fichier peut contenir des cartes //...JØB...

Dans ce cas, sa fin doit obligatoirement être indiquée par une carte ENDDATASET.

J = NØ indique que la première carte //...JØB... rencontrée marque la fin du fichier et le début d'un autre travail. Le fichier peut également dans ce cas se terminer par une carte ENDDATASET mais cela n'est pas obligatoire.

Remarque importante :

La carte DD éventuellement employée pour le fichier créé par //*DATASET doit obligatoirement être de la forme :

```
//ddnombDDbUNIT=(CTC,,DEFER),VØL=SER=123456,
//   DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
```

(Il s'agit en effet des références correspondant au lecteur de cartes)

exemple :

```
//*DATASETbDDNAME=INPUT1,J=YES
:  fichier cartes contenant une ou des cartes //...JØB...
.
//*ENDDATASET
//bEXECbPGM=ALPHA
//INPUT1bDDbUNIT=(CTC,,DEFER),VØL=SER=123456,
//  DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
  ..................
/*
```

9 - CARTE NET.

La carte NET est une carte de contrôle ASP qui permet de forcer l'enchaînement de divers travaux dans un ordre prédéterminé (réseau du travaux = "net").

Forme générale :

```
//*NETbID=nom  [,HC=n/0]  [,RL=(travail1,...travailn)]
[,NC={D|F|R}]  [,AB={D|F|R}]  [,NR=(netnom,travail)]
```

La signification des paramètres est la suivante :

ID : nom du réseau ("net") auquel appartient le travail qui contient cette carte //*NET.
Ce nom est obligatoirement de la forme : SiglennnX (8 caractères max) :

Siglennn = sigle et numéro de calcul de la carte //JØB(WRR1358)
X = 1 caractère alphanumérique au choix).

HC=n "Hold Count" n = nombre de travaux "prédécesseurs", c'est-à-dire qui doivent être terminés pour que celui-ci puisse être exécuté.

$$0 \leqslant n \leqslant 32767$$

Si n = 0 ou si le paramètre HC est omis le travail est exécutable sans préalable.

RL : "Release" : donne les noms des travaux "successeurs" (pour l'exécution desquels, il est nécessaire que le travail qui contient cette carte //*NET soit terminé).

NC : "Normal Completion" : ce que l'on doit faire de ce travail lorsque l'un quelconque de ses "prédécesseurs" se termine correctement :

F = supprimer du système ce travail et tous ses successeurs,
R = conserver ce travail dans le système et ne pas décrémenter la valeur de HC,
D = décrémenter de 1 la valeur de HC. Si HC devient zéro, ce travail peut être exécuté.

AB : "Abnormal completion" : ce que l'on doit faire de ce travail lorsque l'un quelconque de ses "prédécesseurs" se termine incorrectement

F, R et D ont la même signification que pour NC.

NR : "Net Release" : permet de désigner un "successeur" qui se trouve dans un autre réseau. On donne alors le nom du réseau (net-nom) et le nom du travail (travail).

Exemple d'utilisation de la carte NET :

exemple 1 :

Soit T1, T2, T3 et T4 des noms schématisés de 4 travaux passant sous le sigle WRR1358 (cartes JØB).

On désire que pour exécuter T2, T1 doit être terminé et pour exécuter T4 que T2 et T3 soient terminés.

On a pour le travail de nom :

```
T1      //*NET   ID=WRR1358Z,RL=(T2)
T3      //*NET   ID=WRR1358Z,RL=(T4)
T2      //*NET   ID=WRR1358Z,RL=(T4),HC=1
T4      //*NET   ID=WRR1358Z,HC=2
```

exemple 2 :

On a 6 travaux désignés schématiquement par A, B, C, D, E, F passant sous le sigle WRR1358.
On désire que si A ou B se termine anormalement D soit supprimé du système, ce qui entraînera la suppression de son successeur F. On peut ainsi corriger A et/ou B; les resoumettre. Lorsqu'ils se termineront correctement, D et F s'exécuteront.

On a pour le travail de nom :

```
A       //*NET   ID=WRR1358Y,RL=(C,D)
B       //*NET   ID=WRR1358Y,RL=(C,D,E)
C       //*NET   ID=WRR1358Y,RL=(F),HC=2
D       //*NET   ID=WRR1358Y,RL=(F),HC=2,AB=F
E       //*NET   ID=WRR1358Y,RL=(F),HC=1
F       //*NET   ID=WRR1358Y,HC=3.
```

Lorsque cette carte existe, elle se place derrière les cartes JØB et MAIN.

10 - CARTE CIRCE.

Cette carte de contrôle ASP, propre au CIRCE, permet de préciser certaines conditions d'exécution et de restitution (priorité, délai d'attente ..)

Forme générale :

```
//*CIRCEb [PASSWØRD=motpasse]  [,HEADER={YES|NØ}][,NEWS={NØ|YES}]
        [,WAIT={min|4S}]     [,PRTY={incr|0}]
```

La signification des paramètres est la suivante :

PASSWØRD = mot passe : mot de passe de 8 caractères hexadécimaux. Obligatoire pour certains numéros de calcul attribués sur demande expresse de l'utilisateur.
Il est alors lié au numéro de calcul et à une orthographe stricte du nom d'utilisateur fourni dans la carte JØB.

HEADER = YES : Une page séparatrice indiquant en caractères géants le nom de l'utilisateur et celui de la carte DD décrivant ce fichier, précèdera chaque fichier listé. L'édition en sera facturée.

HEADER = NØ : (option par défaut). Cette page ne sera éditée qu'au début du 1er fichier listé. La valeur NØ est forcée pour les travaux listés sur un terminal.

NEWS = YES : donne en fin de liste les informations récentes intéressant les utilisateurs du CIRCE;l'édition n'en est pas facturée. (option par défaut).

NEWS = NØ : pas d'édition de ces informations.

WAIT : Ce paramètre n'est considéré que pour un travail lu en "libre-service". Il donne le temps maximum d'attente consenti par l'utilisateur entre la lecture de son travail et le début d'édition de ses résultats sur une imprimante en "libre-service". Il est exprimé en minutes. Si ce délai est dépassé l'impression se fait en salle machine.
Si WAIT est supérieur à un maximum (actuellement fixé à 45 minutes), il est ramené à ce maximum.

PRTY = incrément compris entre - 15 et + 3. : Donne un incrément qui sera ajouté à la priorité calculée par le système.

Un incrément > 0 entraîne une tarification plus élevée.

exemple d'utilisation de la carte CIRCE :

```
//*CIRCEbHEADER=YES,PRTY=+3
```

On demande d'introduire une page séparatrice entre chaque fichier édité et d'augmenter de 2 degrés la priorité du travail.

11 - CARTE *COMMENTAIRE.

Cette carte de contrôle ASP permet d'introduire pour plus de clarté, des cartes de commentaires entre les instructions de contrôle.

Forme générale :

```
//*COMMENTAIRE
```

Commentaire : zone libre contenant des caractères quelconques en colonnes 4 à 80.

exemple :

```
//*CETTE ETAPE LISTE LES FICHIERS CREES PAR LA PRECEDENTE
```

Remarque très importante :

Les cartes de contrôle propres à ASP se caractérisent également par //* en colonnes 1 à 3, suivi d'un mot conventionnel précisant l'opération attendue (par exemple FØRMAT, MAIN ...) il ne faut donc pas utiliser l'un de ces mots à partir de la colonne 4 d'une carte commentaire : elle serait alors prise comme une carte de contrôle ASP.

La carte commentaire peut être placée n'importe où dans le flot des cartes de contrôle JCL d'un travail entre la carte JØB (la première) et la carte "null" // (la dernière du travail). En particulier elle peut être incorporée entre 2 cartes "suite" d'une carte JCL.

On peut coder plusieurs cartes "commentaire" en séquence.

12 - CARTE DELIMITEUR /*.

Cette carte de contrôle OS/ASP sert de marqueur dans la suite des cartes (JCL + données) constituant le "flot d'entrée" du travail.

Forme générale :

/*b commentaire

Commentaire : zone libre précédée d'au moins un blanc.

exemple : /*bFIN DU FICHIER NUMERO 2

Cette carte marque la fin de données fournies à la suite d'une carte //bDDb* ou //bDDbDATA. (voir chapitre 7.1.)

Si l'on a utilisé le paramètre DLM dans la carte DD qui définit ce fichier, les caractères /* de la carte séparateur devront être remplacés par le délimiteur choisi.

Cette carte est obligatoire pour marquer la fin d'un fichier situé dans le flot d'entrée et contenant des cartes "données" de type JCL, que l'on ne veut pas confondre avec les cartes JCL du travail.

Elle est facultative mais recommandée pour les fichiers qui ne contient pas de cartes JCL.

Pour des raisons évidentes, les fichiers de données placés dans le flot d'entrée ne peuvent comporter des cartes commençant par /* en colonnes 1 et 2 (dans ce cas on utilisera le paramètre DLM de la carte DD définissant le fichier pour spécifier 2 autres caractères comme délimiteur).

Une étape d'un travail peut comporter plusieurs fichiers situés dans le flot d'entrée définis par des cartes DD différentes, la fin de chacun pouvant être marquée par une carte /*.

13 - CARTE FIN DE TRAVAIL "NULL" (//).

Cette carte sert à marquer la fin d'un travail.

Forme générale :

//	en colonnes 1 et 2.

Cette carte est placée derrière la dernière carte JCL d'un travail ou la dernière carte des données situées dans le flot d'entrée.

Si elle est omise toutes les cartes JCL qui suivent feront partie du travail et ce jusqu'à la prochaine carte JØB entrée au lecteur de cartes.

14 - CARTES PRØC et PEND.

Ces cartes complémentaires permettent de définir une séquence de cartes de contrôle (ou procédure) qui sera utilisée plusieurs fois dans le même JØB.

Forme générale de la carte PRØC :

```
//nomdeprocédurebPRØCb[param1=[défaut1]] [,param2=[défaut2]]...[,paramn=[défautn]]
```

Cette carte peut être utilisée lorsqu'un travail reprend plusieurs fois un ensemble de cartes de contrôle définissant une ou plusieurs étapes (carte EXEC et cartes DD) sans changements dans le libellé de ces cartes ou avec quelques changements d'une utilisation à l'autre.

Cet ensemble de cartes peut être préparé en 1 exemplaire qui sera ensuite introduit dans le travail dans le flot d'entrée avant les cartes définissant les étapes du travail. Il constitue alors ce que l'on appelle une "procédure in-stream" (procédure utilisateur située dans le flot d'entrée).

Il suffira ensuite de placer à chaque étape où l'on voudrait utiliser cette procédure une carte EXEC nommant cette procédure. Les cartes JCL composant cette procédure seront automatiquement insérées à la place de cette carte EXEC.

Les paramètres symboliques de la procédure pourront se voir attribuer une valeur différente à chaque nouvel emploi de la procédure.

Pour utiliser une procédure "in-stream" il faut inclure la procédure commençant par une carte PRØC et se terminant par une carte PEND avec les cartes de contrôle du travail (après la carte JØB et avant la carte EXEC qui l'appelle).

Tout se passe comme pour les procédures cataloguées mais cette procédure n'est valable que pour le travail dans lequel elle se trouve.

La signification des paramètres de la carte PRØC est la suivante :

nom de procédure : nom de la procédure "in-stream". 1 à 8 caractères alphanumériques, le premier étant obligatoirement une lettre.

param : nom de paramètres symboliques utilisés dans la procédure. Ils seront identifiés par la notation &symbol

défaut : valeur par défaut à donner au paramètre symbolique correspondant. Cette valeur pourra être changée en assignant, dans la carte EXEC qui appelle la procédure, une nouvelle valeur au même paramètre symbolique. Si la valeur par défaut est omise, le paramètre symbolique est annulé.

Forme générale de la carte PEND :

```
//nombPEND
```

Cette carte indique la fin d'une procédure "in-stream".

nom : 1 à 8 caractères alphanumériques, le premier étant obligatoirement une lettre. Ce paramètre est optionnel.

Exemple d'utilisation des cartes PRØC et PEND (procédure in-stream) :

```
// carte JØB
//EFFACE PRØC VØLUME = PP3330
//bEXECbPGM=IEFBR14
//DD1bDDbDSN=&FICHIER,VØL=SER=&VØLUME,DISP=(OLD,DELETE),
//     UNIT=3330-1
//FINbPEND
//A1bEXECbEFFACE,FICHIER='WRR1358.RØCHEMAR.FIC1'
//A2bEXECbEFFACE,FICHIER='WRR1358.RØCHEMAR.FIC2',VOLUME=RES302
/*
//
```

EFFACE est le nom de la procédure créée qui permet la suppression d ' un fichier dont on donne le nom et éventuellement le volume de résidence. (valeur par défaut : PP3330). Cette procédure est appelée 2 fois dans le travail ci-dessus. La première fois pour un fichier sur le volume PP3330, la 2ème fois pour un fichier sur RES302.

Remarque : il ne doit pas y avoir de carte //bDDbDATA ou //bDDb* dans une procédure "in-stream".

- Au CIRCE, il est conseillé d'utiliser la facilité de la procédure "in-stream" lorsque l'usage de la procédure sera momentané. Si la mise en procédure cataloguée se justifie par un emploi fréquent et prolongé, il convient de s'adresser au GROUPE SYSTEME du Centre de Calcul.

15 - LES PROCEDURES CATALOGUEES.

15.1. Généralités.

Une procédure est une instruction symbolique qui regroupe en une seule carte plusieurs instructions de contrôle.

Nous avons vu dans le chapitre précédent que l'utilisateur peut créér des procédures qu'il introduit dans le flot d'entrée du travail ("procédure in stream").

Lorsque de telles procédures ont été testées, elles peuvent être placées dans un fichier spécial (bibliothèque), d'où elles pourront être utilisées par simple appel à l'aide d'une carte EXEC. On parle alors de "procédures cataloguées.

Une méthode simple pour soumettre un programme écrit en langage évolué à la compilation ou à l'assemblage, au chargement ou à l'exécution est de faire appel à une procédure cataloguée.
La présence de paramètres symboliques permet le changement de certaines options définies dans les procédures.

Au CIRCE, il existe des procédures cataloguées spécifiques au système regroupant des instructions de contrôle JCL et permettant l'emploi facile des compilateurs usuels disponibles (ALGOL, ASSEMBLEUR, COBOL, FORTRAN, PL1).
D'autres procédures particulières ont été cataloguées pour faciliter l'emploi de certains programmes utilitaires ou opérer des fonctions de gestion propres au Centre de Calcul.

Nous ne parlerons ici que des procédures cataloguées les plus couramment utilisées pour les besoins actuels du Service Hydrologique de l'ORSTOM.

15.2. Procédures cataloguées des compilateurs usuels.

Au CIRCE ces procédures ont des noms imposés donnés dans le tableau ci-dessous :

Langage	Procédure utilisant le loader		Procédure utilisant le linkage éditor				Région de l'étape compil.
			noms				
	Nom	Alias	compil	compil + link	compil + link+Exec	link + Exec	
COBOL ANS	CBA	COBOL	CBAC	CBACL	CBACLG	CBALG	120K
FORT G1	FTG1	FORTRANG	FTG1C	FTG1CL	FTG1CLG	FTG1LG	110K
FORT H EXT.	FTX	-	FTXC	FTXCL	FTXCLG	FTXLG	250K
FORT WATFIV	FTW	FORTRAN	-	-	-	-	150K

Si on excepte la procédure WATFIV (FØRTRAN), le cheminement d'un programme source jusqu'à son chargement et son exécution peut se faire selon 2 voies différentes :

- Soit en 2 étapes avec compilation et chargeur (loader) puis exécution.
- Soit en 3 étapes avec compilation puis éditeur de liens (link edit) puis exécution.

La figure 8 schématise ces 2 possibilités pour un programme FØRTRAN.

- Le compilateur est un programme qui permet la traduction d'un "programme source" (écrit en langage évolué) en langage machine appelé "module objet".

- Le link edit (ou éditeur de liens) est un programme qui permet de résoudre les références externes c'est-à-dire qu'il permet en particulier de regrouper le programme principal, les sous programmes écrits par l'utilisateur et les sous programmes de la bibliothèque en un seul module chargeable (Load module). Quand on utilise le link edit, le module chargeable est stocké dans un fichier sur disque (carte SYSLMØD du LINK EDIT). Il ne sera chargé en mémoire et exécuté qu'à l'étape d'exécution (GØ).

- Le loader (ou chargeur) est un programme permettant d'effectuer un link edit et une exécution en une seule étape.
  Quand on utilise le loader le résultat de l'édition est directement chargé en mémoire. Cela nécessite de prévoir environ 10 à 15 K supplémentaires pour la place en mémoire de l'étape d'exécution (GØ).

  On utilisera l'éditeur de liens, si le module chargeable doit être mis en librairie ou si il nécessite l'utilisation de paramètres autres que MAP, LET, NCAL, SIZE. En particulier, on utilisera le LINK-EDIT si l'on désire en sortie le DECK (module objet sur cartes perforées).

  Par contre il est préférable d'utiliser le chargeur toutes les fois que le module chargeable à exécuter ne nécessite pas d'être généré avec des paramètres ou des ordres spécifiques à l'éditeur de liens.

  Le chargeur présente l'avantage de réduire le nombre d'étapes du travail, de réduire le temps d'édition de liens et de chargement (environ 2 fois), de réduire le nombre de fichiers temporaires à allouer pour le travail, de réduire le nombre des opérations d'entrée/sortie et le nombre de lignes à éditer pour le travail.

- Pour la mise au point de programmes FØRTRAN on utilisera de préférence les compilateurs WATFIV et FORTRANG1 qui fournissent des diagnostics très précis.
  Signalons en particulier que seul le compilateur FORTRANG1 permet l'utilisation d'instructionsparticulières (du type DEBUG) pour localiser des erreurs éventuelles en faisant appel à la fonction de mise au point. (voir Annexe II).

  Enfin quand on passe à la phase d'exploitation d'un programme FØRTRAN on a intérêt à utiliser le compilateur FØRTRAN H étendu qui augmente les performances à l'exécution.

Lorsque l'on utilise une procédure cataloguée, le système génère les instructions de contrôle contenues implicitement dans cette procédure. L'utilisateur peut cependant opèrer des remplacements sur ces instructions à l'aide de cartes de contrôle.
En particulier, il peut modifier les paramètres d'une carte DD générée par l'appel de la procédure. Il peut également opérer des additions de cartes de contrôle DD pour définir des fichiers supplémentaires. Ces modifications doivent alors être effectuées en tenant compte des règles suivantes :

- Tous les paramètres de la carte DD générés par la procédure et que l'on n'aura pas modifiés seront conservés.

- Les modifications doivent être effectuées dans l'ordre dans lequel elles apparaissent dans la procédure cataloguée et en particulier dans l'ordre d'enchaînement des étapes (l'étape devant être précisée dans la carte DD modifiée).
  Dans le cas contraire une erreur de JCL est diagnostiquée.
  De même des paramètres se référant à une étape de procédure cataloguée n'existant pas entraîne une erreur de JCL.

- Les cartes DD d'addition doivent suivre les cartes DD de remplacement. (les cartes //FØRT.SYSIN et //GØ.SYSIN sont des cartes d'addition).

- Au CIRCE les noms des cartes DD apparaissant dans les procédures cataloguées sont systématiquement ordonnés alphabétiquement. Ainsi par exemple on aura :

```
//LKED EXEC ...
//SYSLIB DD ...
//SYSLIN DD ...
//SYSLMØD DD ...
//SYSPRINT DD ...
//SYSUT1 DD ...
```

- exemple de modification d'une carte DD d'une procédure cataloguée :

On désire conserver sur bande le résultat d'une compilation faite en employant la procédure FTXC ; on aura :

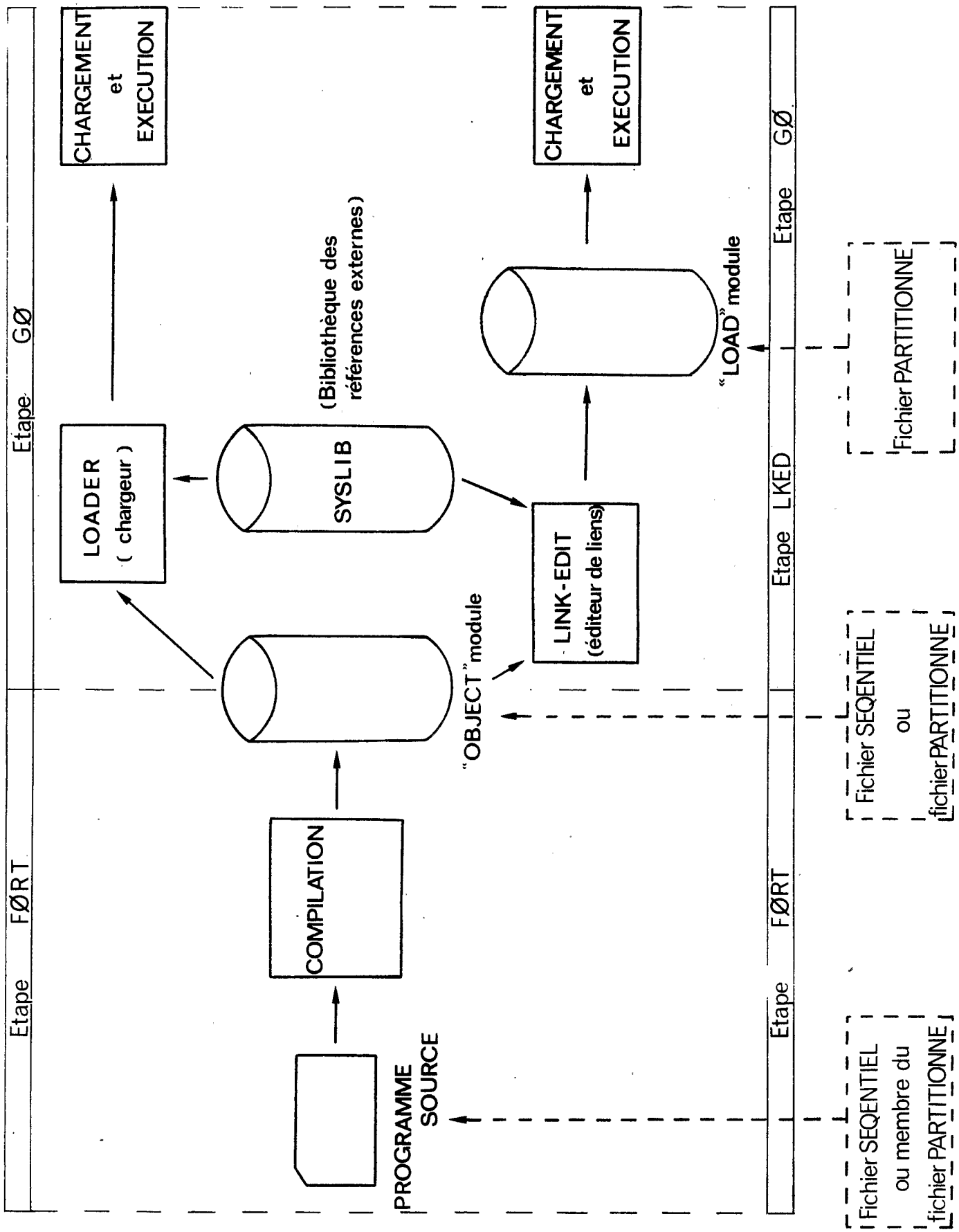
```
//carte JOB
//E1bEXECbFTXC
//FØRT.SYSLINbDDbDSN=COMPIL,UNIT=BD16,VOL=...
//    DISP=(NEW,KEEP)
//FØRT.SYSINbDDb*
   programme à compiler (FORTRAN)
/*
//
```

- exemple d'addition d'une carte DD dans une procédure cataloguée :

On désire créér en sortie sur imprimante un autre fichier que le

Fig_8

CHEMINEMENT D'UN PROGRAMME SOURCE FORTRAN



fichier affecté à l'unité 6 (valeur par défaut de l'unité correspondant à l'imprimante).

Soit 8 ce numéro d'unité pour laquelle on aura programmé des instructions du type WRITE (8,...) ; il faudra donc ajouter à la procédure une carte DD définissant l'unité d'écriture 8.
Cette carte d'addition sera placée après les cartes normales de la procédure :

exemple :

```
// carte JØB
//E1bEXECbFTXCLG
//FØRT.SYSINbDDb*

     programme FORTRAN

 /*
//GØ.SYSINbDDb*

     données sur cartes

 /*
//GØ.FT08F001bDDbSYSØUT=A,
//    DCB=(RECFM=VBA,LRECL=137,BLKSIZE=1922)
//
```

Dans cet exemple les cartes //GØ.SYSIN et //GØ.FT08F001 étant toutes les deux des cartes d'addition, leur ordre est indifférent.

15.3. Utilisation des procédures cataloguées FØRTRAN.

15.3.1. Compilateur WATFIV : Il s'agit d'un compilateur de mise au point très rapide en compilation et qui détecte les erreurs en fournissant des diagnostics précis.

Cette procédure (dite "CODE AND GØ") effectue la traduction et l'exécution d'un programme FORTRAN en une seule étape.

Le nom de cette étape unique est GØ.

L'appel de la procédure se fait par une carte EXEC du type :

```
//nom d'étapebEXECbFTW [,REGIØN=nnnK]
```

L'utilisation de ce compilateur nécessite en plus des cartes de contrôle OS, 3 cartes propres à WATFIV :

```
$JØB
$ENTRY
$STØP
```

Pour ces 3 cartes, le signe $ est perforé en colonne 1.

Un travail utilisant la procédure WATFIV se présentera donc de la façon suivante :

```
// carte JØB usuelle
// nom d'étape bEXECbFTW,REGION=nnnK
//GØ.SYSINbDDb*
$JØB                    IDENT1,PARAMETRES
.............           ↑colonne 16
    Travail (programme et sous-programme)
$ENTRY
    données
$STØP
/*
//
```

Remarques importantes pour l'utilisation du compilateur WATFIV.

- le paramètre REGIØN de la carte EXEC aura pour valeur la région nécessaire pour l'exécution du programme augmentée de 95K pour la place du compilateur (la valeur par défaut est 150 K).
- $ENTRY est obligatoire même s'il n'y a pas de données.
- On peut exécuter plusieurs travaux dans la même étape O.S. (c'est-à-dire correspondant à une carte EXEC FTW unique). Dans ce cas chaque travail comportera une carte $JOB et une carte $ENTRY et l'ensemble de ces travaux se terminera par une seule carte $STØP.

Les paramètres de la carte JOB sont les suivantes : (valeur par défaut soulignées)

KP = | 29 / 26 Code BCD (26) ou EBCDIC (29) dans lequel est perforé le programme FORTRAN

TIME = | 1 / M nombre entier de minutes

PAGES = | 100 / N nombre entier maximum de pages permises en édition.

LINES = | 60 / K nombre de lignes par page.

RUN = | CHECK — CHECK contrôle l'usage des variables non définies.
| NØCHECK — NOCHECK ne le contrôle pas.
| FREE — FREE permet l'exécution malgré des erreurs éventuelles à la compilation.

LIST	listing du programme source
NØLIST	suppression de ce listing.
LIBLIST	listing des programmes recueillis dans des bibliothèques
NØLIBLIST	suppression de ce listing.
WARN	Edition des erreurs dites "WARNING" et "EXTENSION"
NØWARN	Suppression de cette édition.
EXT	Edition des messages dits "EXTENSION"
NØEXT	Suppression de cette édition.

Tous ces paramètres sont optionnels et à mot-clé, c'est-à-dire que leur ordre est indifférent.

Le compilateur WATFIV donne 3 sortes de diagnostics à la compilation :

*** ERROR ***	erreur grave qui empêche l'exécution du programme.
** WARNING **	erreur que le compilateur a pris sur lui de corriger raisonnablement.
* EXTENSION *	syntaxe permise propre à WATFIV. Cette syntaxe n'est pas autorisée par les autres compilateurs FORTRAN (FTG ou FTX).

Le compilateur WATFIV permet un certain nombre d'extensions de langage par rapport au langage FORTRAN IV.
Par contre, il comporte quelques restrictions et incompatibilités par rapport au FORTRAN IV.
Pour plus de détails concernant l'utilisation de ce compilateur, on se reportera à la notice "USER'S GUIDE WATFIV" publiée par le CIRCE.

15.3.2. Compilateur FØRTRANG1 et FØRTRANH étendu.

Selon les fonctions à exécuter, les procédures utilisant ces compilateurs comportent une, deux ou trois étapes.

Ces étapes ont des noms imposés :

- étape FØRT : compilation
- étape LKED : éditeur de liens
- étape GØ : chargement et exécution.

L'appel de ces procédures se fait à l'aide d'une seule carte EXEC.

- Procédures FTG1 et FTX. Ces procédures permettent de compiler, charger et exécuter un programme FØRTRAN.

Elles ne comportent que 2 étapes car elles utilisent le loader (chargeur) :

compilation (FØRT)
chargement et exécution (GØ).

L'emploi du loader nécessite d'augmenter d'environ 10K la place mémoire réservée pour l'étape GØ.

Les noms des cartes DD (valeurspar défaut) inclusdans ces procédures pour les fichiers usuels sont :

lecture des données sur cartes : FT05F001 et SYSIN (1)
impression des résultats : FT06F001
perforation des cartes : FT07F001

Les autres fichiers doivent être définis par des cartes DD.
Exemple d'utilisation : compilation et exécution d'un programme source FORTRAN avec données sur cartes :

```
// carte JØB
//bEXECbFTX,REGION.GØ=150K
//FØRT.SYSINbDDb*
|programme source FORTRAN sur cartes
/*
//GØ.SYSINbDDb*
|données sur cartes
/*
//
```

Remarque : Au CIRCE la procédure FTG1 peut également être appelée FORTRANG. L'appel se fera donc soit par :

```
//bEXECbFTG1,
```

soit par //bEXECbFORTRANG

- Procédures FTG1C et FTXC. Ces procédures permettent de compiler un programme FØRTRAN.
On a donc une seule étape (FØRT).

---

(1) - Remarque : Les procédures cataloguées ne peuvent pas contenir de cartes //DDb* ou //DDbDATA. Lors de l'appel de ces procédures on rajoute une carte DD remplaçant la carte DD de la procédure et définissant le flot d'entrée.

exemple : Si dans une procédure, on a la carte.

```
//FT05F001bDDbDDNAME=SYSIN
```

pour lire les données lors de l'exécution on écrira :

```
//SYSINbDDb*
```

exemple : compilation simple d'un programme source FØRTRAN à l'aide du compilateur H étendu avec demande en sortie du DECK (module objet = programme compilé mis sur cartes perforées)

```
// carte JØB
//bEXECbFTXC,PARM.FØRT='DECK'
//FØRT.SYSINbDDb*
    |programme FØRTRAN sur cartes
/*
//
```

- Procédures FTG1CL et FTXCL : Ces procédures permettent de compiler et effectuer le link-edit (éditeur de liens) d'un programme FØRTRAN en 2 étapes :

  - compilation (FØRT)
  - éditeur de liens (LKED)

exemple : compilation et link-edit d'un programme source FORTRAN à l'aide du compilateur G1 :

```
// carte JØB
//bEXECbFTG1CL
//FØRT.SYSINbDDb*
    |programme FORTRAN sur cartes
/*
//
```

- Procédures FTG1CLG et FTXCLG : Ces procédures permettent de compiler, effectuer le link edit et exécuter un programme source FØRTRAN.

On a 3 étapes :

  - compilation (FØRT)
  - éditeur de liens (LKED)
  - exécution (GØ)

les noms des cartes DD inclues dans ces procédures pour les fichiers usuels sont :

lecteur des données : FT05F001 et SYSIN
impression des résultats : FT06F001
perforation des cartes : FT07F001

Nous donnons en annexe 4 la liste des instructions de contrôle générées par le système à l'appel de la procédure FTXCLG

exemple d'utilisation : compilation, link edit et exécution d'un programme source FORTRAN à l'aide du compilateur H étendu

et avec données sur cartes :

```
// carte JØB
//bEXECbFTXCLG,REGIØN.GØ=200K
//FØRT.SYSINbDDb*
    |programme source FØRTRAN sur cartes.
/*
//GØ.SYSINbDDb*
    |données sur cartes
/*
//
```

- Procédures FTG1LG et FTXLG : Ces procédures permettent d'effectuer le link edit et l'exécution d'un programme déjà compilé (c'est-à-dire sous forme de "module objet"). On n'a donc que 2 étapes :
  - éditeur de liens (LKED)
  - exécution (GØ).

  exemple d'utilisation : link edit et exécution d'un programme compilé en binaire (DECK) et mis sur cartes avec données également sur cartes : (dans cet exemple la compilation est supposée avoir été effectuée à l'aide du compilateur H étendu) :

```
// carte JØB
//bEXECbFTXLG,REGIØN.GØ=100K
//LKED.SYSINbDDb*
    |programme binaire sur cartes (DECK)
/*
//GØ.SYSINbDDb*
    | données sur cartes (fichier SYSIN)
/*
//
```

15 . 4 . Utilisation des procédures cataloguées CØBØL :

Au CIRCE ces procédures utilisent le compilateur CØBØL ANS.

Comme pour les procédures FØRTRAN, elles comportent soit une, soit deux, soit trois étapes selon les fonctions à éxécuter et l'emploi ou non du loader.

Ces étapes ont des noms imposés :

- étape CØB : compilation
- étape LKED : éditeur de liens
- étape GØ : chargement et exécution.

L'appel de ces procédures se fait à l'aide d'une seule carte EXEC.

- Procédure CBA (alias CØBØL) : Cette procédure utilise le loader et ne comporte donc que les étapes de compilation (CØB) et de chargement et exécution (GØ).

Les noms des cartes DD inclus dans la procédure et correspondant à des fichiers usuels sont :

lecture de cartes : SYSIN
impression de résultats : SYSØUT

Pour les autres fichiers le DDNAME est le nom du fichier qui suit l'expression ASSIGN TØ dans le programme CØBØL (exemple : SYS005, SYS006, ...)

exemple d'utilisation : programme source CØBØL et données sur cartes.

```
// carte JØB
//bEXECbCBAb,REGIØN.GØ=200K
//CØB.SYSINbDDb*
    | programme CØBØL sur cartes
/*
//GØ.SYSINbDDb*
    | données sur cartes
/*
//
```

- Procédure CBAC : Cette procédure permet la simple compilation d'un programme CØBØL et comporte donc une seule étape (CØB).

exemple d'utilisation :

```
// carte JØB
//bEXECbCBAC
//CØB.SYSINbDDb*
    | programme CØBØL sur cartes
/*
//
```

- Procédure CBACL : Cette procédure permet de compiler et d'effectuer le link edit d'un programme source CØBØL.

  2 étapes : CØB (compilation) et LKED (link-edit).

  exemple d'utilisation : programme source COBOL et données sur cartes

```
// carte JØB
//bEXECbCBACL
//CØB.SYSINbDDb*
    | programme CØBØL sur cartes
/*
//
```

- Procédure CBACLG : Cette procédure permet de compiler, d'effectuer le link-edit et l'exécution d'un programme source CØBØL en 3 étapes (CØB, LKED, GØ).

  Les noms des cartes DD inclusdans la procédure et correspondant à des fichiers usuels sont :

  lecture de cartes : SYSIN
  impression de résultats : SYSØUT

  Pour les autres fichiers le DDNAME est le nom du fichier qui suit l'expression ASSIGN TØ dans le programme CØBØL (exemple : SYS004, SYS005, SYS007 ...)

  exemple d'utilisation : programme source CØBØL et données sur cartes : (fichiers SYSIN, SYS006, SYS007)

```
// carte JØB
//bEXECbCBACLG,REGIØN=90K
//CØB.SYSINbDDb*
    | programme CØBØL sur cartes
/*
//GØ.SYSINbDDb*
    | données sur cartes (fichier SYSIN)
/*
//GØ.SYS006bDDb*
    | données sur cartes (fichier SYS006)
/*
//GØ.SYS007bDDb*
    | données sur cartes (fichier SYS007)
/*
//
```

- Procédure CBALG. Cette procédure permet d'effectuer le link-edit et l'exécution d'un "module objet" obtenu par compilation d'un programme source CØBØL à l'aide du compilateur CØBØL ANS. Cette procédure ne comporte donc que 2 étapes :

  - étape LKED (link-edit)
  - étape GØ (exécution).

  exemple d'utilisation : utilisation d'un DECK COBOL (module objet-binaire) :

```
// carte JØB
//bEXECbCBALG
//LKED.SYSINbDDb*
    | programme binaire sur cartes (DECK)
/*
//GØ.SYS008DDb*
    | données sur cartes (fichier SYS008)
/*
//
```

On trouvera en annexe des exemples variés d'utilisation de l'ensemble de ces procédures cataloguées. (En particulier avec des données constituant des fichiers sur bandes magnétiques ou sur disques).

15.5. Autres procédures cataloguées : Il existe de nombreuses procédures cataloguées autres que celles liées aux compilateurs et décrites dans les paragraphes précédents.

Parmi ces procédures mises à la disposition des utilisateurs, certaines sont spécifiques au système IBM 370 et d'autres spécifiques au Centre de Calcul (procédures CIRCE).

Nous n'examinerons que les plus utiles à la gestion des fichiers qui sont en fait des "procédures utilitaires" et qui seront donc décrites dans la 4ème partie de cette note avec les programmes utilitaires.

Il s'agit essentiellement des procédures suivantes :

SØRT/MERGE : permet le tri et la fusion d'enregistrements (procédure IBM).

CØMPRIME : permet la compression d'un fichier partitionné.

DUDUMP : appel du programme d'analyse de bande DUDUMP.

DUMPBAND : appel du programme d'analyse de bande DUMPBAND.

EFFACE : suppression d'un fichier résidant sur disque magnétique.

RESER : réservation de la place d'un fichier sur un disque magnétique.

4ème PARTIE : LES PROGRAMMES UTILITAIRES

Nous avons rassemblé sous le terme de "programmes utilitaires" un certain nombre de procédures et de programmes de servitude dont l'emploi particulièrement facile permet d'effectuer très rapidement la plupart des opérations courantes en gestion de fichiers (tri, fusion, copie, analyse des fichiers, suppression des fichiers ...).

Parmi ces "utilitaires" nous avons retenu les plus couramment employés dont nous donnons la liste ci-dessous en distinguant d'une part les procédures cataloguées (bibliothèque IBM ou CIRCE) et d'autre part les programmes de servitude de l'Operating System (OS) propres à l'ordinateur 370 IBM.

- Procédures cataloguées "utilitaires". Elles sont appelées par une seule carte EXEC.

  SØRT-MERGE : permet le tri ou la fusion d'enregistrements (IBM)

  RESER : permet la réservation d'une espace sur disque pour un fichier (CIRCE)

  CØMPRIME : permet la compression d'un fichier partitionné (CIRCE)

  EFFACE : permet la suppression d'un fichier résidant sur disque (CIRCE)

  DUDUMP : permet l'analyse des caractéristiques des fichiers contenus sur une bande magnétique (CIRCE)

  DUMPBAND : permet l'analyse et le listage des enregistrements contenus sur une bande magnétique (CIRCE)

  BØTUTI : permet d'obtenir la table des matières (VTØC) d'un disque ou les caractéristiques d'un fichier particulier , ou la liste d'un fichier séquentiel ou de tous les membres d'un fichier partitionné.

- Programmes utilitaires "SYSTEM". Ces programmes sont utilisés pour la maintenance et l'exploitation du système.

  IEHLIST : permet le listage de la VTØC, du DIRECTØRY ou du CATALØGUE d'une mémoire à accès sélectif (M.A.S.)

  IEHPRØGM : permet de modifier ou d'effacer un fichier séquentiel ou partitionné ou un membre d'un fichier partitionné

  IEHMØVE : permet la sauvegarde d'un fichier partitionné sur bande et son contraire.

- Programmes utilitaires "DONNEES". Ces programmes servent à créér, comparer, changer, dupliquer des fichiers.

IEBGENER : permet la copie d'un fichier séquentiel sur n'importe quel support avec ou sans changement d'organisation de tout ou partie du fichier.

IEBPTPCH : permet le listage sur imprimante ou la perforation sur cartes d'un fichier séquentiel ou partitionné avec ou sans changement d'organisation de tout ou partie du fichier.

IEBCØPY : permet la copie de tout ou partie d'un fichier partitionné.

IEBUPDTE : permet de créér et de mettre à jour des fichiers séquentiels ou partitionnés.

IEFBR14 : permet de réserver de la place sur un disque pour un fichier séquentiel. Permet également d'effacer un fichier sur disque.

Dans cette note, nous nous limiterons à la description sommaire et à l'emploi pratique des "utilitaires" qui présentent le plus d'intérêt pour les besoins des utilisateurs du Service Hydrologique.

1. TRI-FUSIØN (SØRT-MERGE)

1.1. Fonction.

Il s'agit de la procédure cataloguée SØRT mise au point par IBM permettant l'utilisation du "programme P.S.M." (programme Sort Merge) dont les principales fonctions sont les suivantes :

- trier des enregistrements (SØRT) c'est-à-dire les ranger selon un ordre donné

- fusionner de 2 à 16 séquences d'enregistrements (MERGE). Les séquences à traiter doivent avoir été triées au préalable dans le même ordre que celui désiré pour le fichier final de sortie et selon des zones de contrôle identiques.

Les ordres de tri possibles sont l'ordre ascendant et l'ordre descendant

exemple ordre ascendant en EBCDIC :

blanc .(+ & ø *) - / , = A B C ... X Y Z 0 1 2 3 4 5 6 7 8 9

1.2. Contraintes.

1.2.1. TRI Le fichier d'entrée doit être séquentiel, bloqué ou non, composé d'enregistrements de longueur fixe ou variable, situé sur n'importe quelle unité.
S'il y a plusieurs fichiers à traiter en entrée, ils peuvent être concaténés.

La longueur d'un enregistrement ne doit pas dépasser une piste (les fichiers de travail étant situés sur des unités à accès direct).

1.2.2. FUSIØN Les fichiers d'entrée peuvent être au nombre de 2 à 16. Ils doivent être organisés séquentiellement et contenir des enregistrements de longueur fixe ou variable.

Les enregistrements des fichiers d'entrée doivent déjà être triés selon l'ordre désiré pour le fichier de sortie final.

Tous les enregistrements doivent être du même format ; les facteurs de blocage peuvent être différents d'un fichier à l'autre mais dans ce cas le plus grand doit correspondre au fichier nommé le premier.

1.3. Zones de contrôle et séquence de classement.

Les enregistrements sont classés selon une ou plusieurs "zone de contrôle" spécifiées par l'utilisateur.
La première zone spécifiée est appelée la zone majeure ;
Les autres zones sont appelées zones mineures.

Il peut y avoir un maximum de 64 zones de contrôle. Elles peuvent se chevaucher ou être contenues à l'intérieur d'autres zones de contrôle. Il n'est pas nécessaire qu'elles soient contiguës mais elles doivent être situées dans les 4092 premiers octets de l'enregistrement.

Chaque zone peut atteindre 256 octets de long (mot de contrôle).

La figure 9 donne l'exemple d'un mot de contrôle constitué de 4 zones de contrôle.

Les mots de contrôle pour chaque enregistrement sont traités selon la séquence EBCDIC.

1.4. Mise en oeuvre du Programme SØRT - MERGE (P.S.M.).

Pour exécuter le P.S.M. l'utilisateur doit écrire 2 sortes de cartes :

- des cartes directives
- des cartes J.C.L.

Les cartes directives sont traitées par le P.S.M.
Les cartes J.C.L. sont traitées par l' Operating System (O.S.).

En général l'appel au P.S.M. se fait à l'aide de la procédure SØRT. D'autres possibilités d'appel sont possibles mais, elles ne seront pas décrites dans cette note.

1.4.1. Les cartes directives.

1.4.1.1. Forme générale des cartes directives.

Les directives sont des informations qui permettent

- de décrire le fichier d'entrée
- de fournir des renseignements concernant les zones de contrôle sur lesquelles le tri ou la fusion seront réalisés.

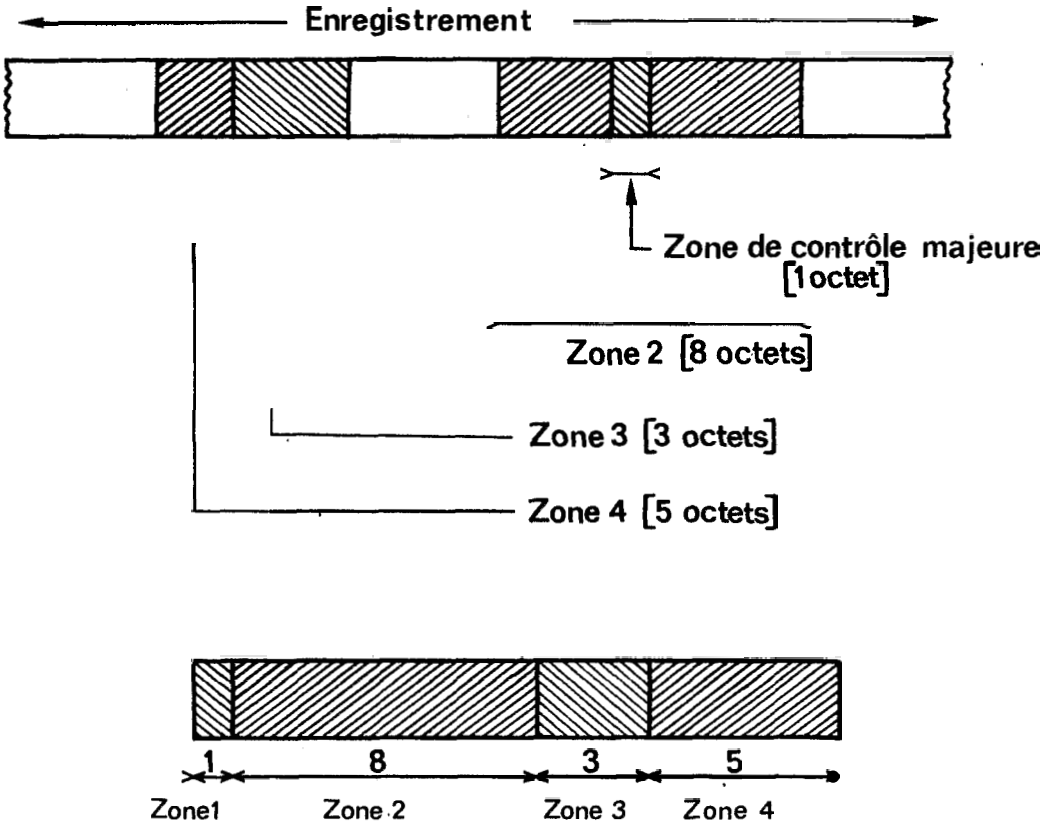
Nous n'examinerons ici que les 3 directives les plus importantes :

SØRT : fournit des informations concernant la zone de contrôle et la taille du fichier à traiter. Cette directive est utilisée pour un tri (SØRT) ; elle ne doit pas être utilisée pour une fusion.

MERGE : fournit des informations au sujet de la zone de contrôle et de la taille du fichier à traiter. Cette directive sert pour réaliser une fusion (MERGE) ; elle ne doit pas être utilisée pour un tri.

END : signale la fin d'une série de directives.

EXEMPLE DE MOT DE CONTROLE CONSTITUE DE 4 ZONES DE CONTROLES [TRI/FUSION]



La forme générale d'une carte directive est la suivante :

b zone d'écriture du nom de la directive	b opérande (≤ colonne 71)	b Commentaires
↑ Colonne 1 (blanc obligatoire)		↑ Colonne 72 (colonne suite)

- la carte commence obligatoirement par un blanc

- la zone d'écriture du nom de la directive doit être la première d'une carte directive. Elle ne peut dépasser la colonne 71. Elle contient l'un des mots SØRT, MERGE ou END.

- la zone des opérandes est la seconde de la carte directive. Elle comprend 1 ou plusieurs opérandes séparés par une virgule.
  Si l'écriture dépasse la colonne 71, la zone des opérandes doit débuter sur la carte où est écrit le nom de la directive.
  Chaque opérande est défini par un mot-clé

  mot-clé = (1re valeur, 2ème valeur, ... $n^{ème}$ valeur).

  exemples: FIELDS = (1, 5, FL, A)
  FORMAT = CH

- la zone commentaire est facultative et peut contenir toute information désirée.

- la colonne 72 est réservée à la demande éventuelle de cartes suites. On y perfore alors un caractère quelconque.

  Une carte suite comporte 1 blanc en colonne 1. L'écriture du paramètre ou du commentaire en cours doit débuter dans les colonnes 2 à 16.
  Le nombre maximal de cartes suites pour les directives SØRT et MERGE est de 19.

- colonnes 73 à 80 zone identificatrice au choix de l'utilisateur.

Restrictions importantes :

L'inclusion de blancs est interdite dans les opérandes,
les valeurs ne peuvent pas dépasser 8 caractères,
les virgules et blancs servent de délimiteurs,
chaque type de directive ne doit apparaitre qu'une fois dans une exécution de SORT-MERGE,
le nombre de cartes directives (y compris les cartes suites) ne doit pas dépasser 33.

1.4.1.2. Directives SØRT ou MERGE :

La forme simplifiée la plus courante est :

```
bSØRT  }b{FIELDS = (p1,m1,f1,s1,p2,m2,f2,s2,...p64,m64,f64,s64)                    }
  ou   }b{FIELDS = (p1,m1,s1,p2,m2,s2,...p64,m64,s64), FORMAT = f                  }
bMERGE }
                  [,SIZE = y]   [,SKIPREC(1) = z]
```

(1) - Le paramètre SKIPREC ne peut être utilisé que pour un tri (SØRT)

Signification et codage des paramètres :

FIELDS : description des zones de contrôle (1 à 64). Les zones doivent être décrites en ordre décroissant d'importance (majeures, mineures ...)

Quatre informations doivent être fournies pour décrire une zone de contrôle des enregistrements d'entrée :

p : début de la zone de contrôle de l'enregistrement. Il s'agit de la position de la zone à l'intérieur de l'enregistrement logique (en octets et en tenant compte éventuellement du compteur d'enregistrement).

m : longueur de la zone de contrôle (en octets). $\leqslant 256$ octets.

f : format des enregistrements dans la zone de contrôle. Ce paramètre est codé de la façon suivante :

CH : caractères EBCDIC non signés
FL : flottant, signé
FI : entier, signé

s : séquence désirée :

A : séquence ascendante
D : séquence descendante

FØRMAT = f Ce paramètre optionnel peut être utilisé si toutes les zones de contrôle ont le même format.
f est alors codé comme dans le paramètre FIELDS (voir ci-dessus)

SIZE = y - Pour un tri (SØRT) nombre d'enregistrements du fichier à trier
- Pour une fusion (MERGE) nombre total d'enregistrements pour tous les fichiers d'entrée à fusionner.

Si y est un nombre estimé, la valeur doit être précédée du caractère E (SIZE = Ey).

SKIPREC = z Ce paramètre ne peut être utilisé que pour un tri. Il donne le nombre z d'enregistrements du fichier d' entrée que l'utilisateur veut ignorer avant de commencer le traitement.

Exemples d'écriture :

b SØRTbFIELDS = (73,6,A,79,2,A), FORMAT = CH, SIZE = 270

ou

b SØRTbFIELDS = (73,6,CH,A,79,2,CH,A), SIZE = 270

Directive SØRT (il s'agit d'un tri)
Opérandes FIELDS et SIZE :

Pour ces 2 cartes, on a 2 zones de contrôle décrites : la première ou zone majeure commence au 73ème octet de l'enregistrement et s'étend sur 6 octets ; elle contient descaractères EBCDIC. La deuxième ou zone mineure commence au 79ème octet de l'enregistrement, s'étend sur 2 octets, contient des caractères EBCDIC.
Le tri sera effectué en ordre ascendant. Le nombre d'enregistrements du fichier d'entrée est de 270.

b MERGEbFIELDS = (73,8,CH,A), SIZE = E80

Directive MERGE (il s'agit d'une fusion)
Opérandes FIELDS et SIZE :

Une zone de contrôle est décrite ; elle commence au 73ème octet des enregistrements et s'étend sur 8 octets, contient des caractères EBCDIC. La fusion est à réaliser en ordre ascendant.
Le nombre d'enregistrements des fichiers d'entrée est estimé à 80.

1.4.1.3. Directive END. Cette directive doit être utilisée quand les programmes utilisateurs sont dans le flot d'entrée.
Elle ne contient aucun opérande et doit être placée après toutes les autres cartes directives :

bEND

1.4.2. Les cartes JCL.

Une étape faisant appel au P.S.M. contient généralement une carte EXEC et des cartes DD relatives aux fichiers utilisés.

1.4.2.1. Carte EXEC. Elle fait l'appel au P.S.M. et est de la forme :

//nomd'étapebEXECbSØRT

1.4.2.2. Cartes DD. Les cartes DD suivantes sont à fournir

- carte //SØRT.SØRTIN ...

Cette carte définit le fichier d'entrée pour un tri. Elle n'est pas nécessaire pour une fusion seule.

Si le fichier d'entrée est le résultat d'une concaténation, l'utilisateur doit appliquer les règles suivantes :

- RECFM doit être le même pour tous les fichiers concaténés
- BLKSIZE peut varier, mais le fichier ayant le plus grand paramètre de blocage (BLKSIZE) doit être nommé dans la première carte DD.

Les performances du tri sont augmentées si les enregistrements sont bloqués.

exemples d'écriture de la carte SØRTIN :

```
// SØRT.SØRTINbDDb*
```

Le fichier d'entrée est sur cartes et situé dans le flot d'entrée.

```
//SØRT.SØRTINbDDbDSN=WRR1358.RØCHEMAR.FICH1,DISP=ØLD,
//        UNIT = 3330-1, VOL = SER = PP3330
```

Le fichier d'entrée est sur disque 3330

- Cartes SØRT.SØRTINnn.

Ces cartes décrivent les caractéristiques de fichiers d'entrée pour une fusion. Elles ne sont pas utiles pour un tri.

nn : nombre compris entre 01 et 16. Ces numéros doivent être donnés en ordre croissant. On ne peut pas sauter de numéro.

Le fichier ayant le plus grand blocksize doit être défini par SØRTINO1.
Le format d'enregistrement doit être le même pour tous les fichiers d'entrée. La longueur de l'enregistrement logique doit également être la même pour tous les fichiers d'entrée.
Si les enregistrements sont de longueur variable, la taille la plus grande doit être affectée au fichier SØRTINO1.

exemple de cartes SØRT.SØRTINnn :

```
//SØRT.SØRTINO1bDDb*
  | cartes données 1er fichier
/*
//SØRT.SØRTINO2bDDb*
  | cartes données 2ème fichier
/*
//SØRT.SØRTINO3bDDb*
  | cartes données 3ème fichier
/*
```

Dans cet exemple, il y a 3 fichiers d'entrée sur cartes à fusionner.

- Cartes SØRT.SØRTWKnn

Ces cartes décrivent les caractéristiques des fichiers utilisés pour le stockage intermédiaire des enregistrements à trier. Ces fichiers sont inutiles pour une fusion seule.
Ces fichiers sont séquentiels et situés sur disque.
Nous examinerons dans le paragraphe 1.4.3. la façon de calculer l'encombrement de ces fichiers de travail (calcul du paramètre SPACE).

nn : nombre compris entre 01 et 32. L'ordre des cartes doit être tel que les numéros soient croissants. Il ne faut pas sauter de numéros.

exemple de cartes SØRT.SØRTWKnn :

```
//SØRT.SØRTWKO1bDDbUNIT=3330,SPACE=(TRK,(5,1) )
//SØRT.SØRTWKO2bDDbUNIT=3330,SPACE=(TRK,(5,1) )
//SØRT.SØRTWKO3bDDbUNIT=3330,SPACE=(TRK,(5,1) )
```

- Carte SØRT.SØRTØUT.

Cette carte décrit les caractéristiques du fichier dans lequel les enregistrements triés ou fusionnés sont placés.

Les performances du tri sont augmentées si les enregistrements du fichier de sortie sont bloqués.

exemples d'écriture de carte SØRT.SØRTØUT :

```
//SØRT.SØRTØUTbDDbSYSØUT=A
```

dans cet exemple le fichier de sortie est imprimé.

```
//SØRT.SØRTØUTbDDbSYSØUT=B
```

dans cet exemple le fichier de sortie est sur cartes perforées

```
//SØRT.SØRTØUTbDDbUNIT=3330,DSN=&&F1,DISP=(NEW,PASS),
//     SPACE=(TRK, (10,5) )
```

dans cet exemple le fichier de sortie est sur disque ; il est temporaire et passé à l'étape suivante.

- Carte SØRT.SYSIN.

Cette carte sert à introduire les cartes directives du P.S.M. dans le flot d'entrée du travail

exemple :

```
//SØRT.SYSINbDDb*
bSØRTbFIELDS=(1,13,CH,A), SIZE = E2000
bEND
/*
```

1.4.3. Calcul de l'encombrement des fichiers de travail. (tri)

Le nombre de pistes requises pour effectuer le tri des enregistrements d'un fichier est donné par la formule suivante :

$$T = \frac{S.N}{K\ (N - 1)} + 2N$$

où : T = nombre de pistes requises pour l'ensemble des fichiers de travail.
N = nombre de fichiers de travail ; ce nombre doit être compris entre 3 et 6.
S = nombre exact ou approché d'enregistrements du fichier d'entrée.
$K = \frac{B}{L}$ avec :

B = 12000 pour les disques 3330
L = longueur en octets de chaque enregistrement du fichier d'entrée. Pour des enregistrements de longueur variable, L est la longueur maximale.

On ne conserve que la partie entière de K ; si le calcul donne K = 0, on prendra K = 1.

L'utilisateur peut demander jusqu'à 6 fichiers de travail. Le nombre T de pistes calculé doit être réparti d'une manière égale entre les fichiers.

1.5. Exemples d'utilisation du programme de TRI-FUSION.

1.5.1. exemple d'utilisation pour un tri.

Soit à trier des enregistrements de précipitations journalières sur cartes (modèle ØRSTØM - CØH 101) avant traitement par un programme dans une étape ultérieure.

- calcul de l'encombrement des fichiers de travail :

supposons que le nombre estimé de cartes à trier soit de 2000. On désire utiliser 3 fichiers de travail.
La formule du paragraphe 1.4.3. donne comme nombre de pistes requises T pour les fichiers de travail.

$$T = \frac{2000 \times 3}{150 \times 2} + 6 = 26$$

On prendra donc 9 pistes pour chaque fichier de travail.

- zones de contrôle.

On désire effectuer le tri sur les 13 premières colonnes (numéro de station, année, mois, quinzaine) en ordre ascendant.

On aura donc :

```
// carte JØB
//TR1bEXECbSØRT
//SØRT.SØRTINbDDb*
{.....
cartes CØH 101 à trier
.....
/*
//SØRT.SØRTØUTbDDbUNIT=3330,DSN=&PJ,DISP=(NEW,PASS),
//   SPACE=(CYL,(1,1),RLSE),DCB=(RECFM=FB,LRECL=80,BLKSIZE=12800)
//SØRT.SØRTWKO1bDDbUNIT=3330,SPACE=(TRK,(9,1) )
//SØRT.SØRTWKO2bDDbUNIT=3330,SPACE=(TRK,(9,1) )
//SØRT.SØRTWKO3bDDbUNIT=3330,SPACE=(TRK,(9,1) )
//SØRT.SYSINbDDb*
bSØRTbFIELDS=(1,13,CH,A),SIZE=E2000
bEND
/*
.
.
.
étapes suivantes
```

Dans cet exemple, les cartes triées constituent le fichier temporaire &PJ qui est mis sur disque avec un facteur de blocage de 160 et qui sera repris dans une étape ultérieure du JØB.

En sortie sur imprimante si l'exécution du tri a été correcte, le système répète les spécifications des cartes directives et donne les nombres exacts d'enregistrements du fichier d'entrée (IN) et du fichier de sortie (OUT)

Les messages donnant ces indications sont de la forme :

```
.....
IGH070I - FILE SIZE      =      2000 SPECIFIED
IGH045I - END  SØRT PH
IGH049I - SKIP MERGE PH
IGH054I - RCD  IN  1981,OUT              1981
IGH052I - END  OF  SORT/MERGE
```

Dans cet exemple le nombre estimé d'enregistrements à trier est de 2000 et le nombre exact est de 1981.

1.5.2. exemple d'utilisation pour une fusion.

Supposons que pour effectuer une mise à jour, on désire fusionner le

fichier &PJ de l'exemple précédent et un fichier identique de pluviométries journalières déjà créé dans un JOB précédent et mis sur bande magnétique et dont le DSN est PJFICH1.

Il est donc nécessaire que le fichier PJFICH1 contienne des enregistrements identiques à ceux de &PJ et triés selon le même ordre.
Supposons que le fichier PJFICH1 soit constitué d'environ 40000 enregistrements et que ses caractéristiques soient les suivantes :

```
DSN      = PJFICH1
RECFM    = FB
LRECL    = 80
BLKSIZE  = 8000
```

numéro de la bande support : SER = 100000. On suppose que ce fichier est le 3ème de la bande qui possède des labels standards IBM.

On aura en partant du fichier d'origine :

```
//  carte JØB
//TR1bEXECbSØRT
.
.   1re étape (voir exemple précédent)
.
/*
//FUS1bEXECbSØRT
//SØRT.SØRTINO1bDDbUNIT=3330,DISP=ØLD,DSN=&PJ
//SØRT.SØRTINO2bDDbUNIT=BD16,DISP=(ØLD,KEEP),
//    VØL=SER=100000,LABEL=(3,SL),DSN=PJFICH1,
//    DCB=(RECFM=FB,LRECL=80,BLKSIZE=8000)
//SØRT.SØRTØUTbDDbUNIT=BD16,DISP=(NEW,KEEP,KEEP),
//    DSN=PJFICH2,LABEL=(1,SL),
//    DCB=(RECFM=FB,LRECL=80,BLKSIZE=12800)
//SØRT.SYSINbDDb*
bMERGEbFIELDS=(1,13,CH,A),SIZE=E42000
bEND
/*
//
```

Dans cet exemple le fichier résultant de la fusion a pour DSN : PJFICH2 et il constitue le 1er fichier d'une bande qui sera attribuée à l'utilisateur. Ce nouveau fichier mis à jour a un facteur de blocage de 160 au lieu de 100 pour le fichier PJFICH1.

On remarquera dans cet exemple que le premier des 2 fichiers à fusionner nommé par SØRTINO1 est celui qui a le plus grand blocksize c'est-à-dire le fichier &PJ (BLKSIZE = 12800).

En sortie sur imprimante si l'exécution de la fusion a été correcte, le système répète les spécifications des cartes directives et donne le nombre exact d'enregistrements du fichier de sortie. Les messages donnant ces indications sont de la forme :

```
IGH054I - RCD IN       ,OUT 43628
IGH052I - END OF SORT/MERGE
```

2. Réservation, Compression, Suppression d'espace sur disque

Ces opérations peuvent être réalisées à l'aide des procédures cataloguées RESER, COMPRIME et EFFACE prôpres au CIRCE et des programmes utilitaires IEFBR14 et IEHPRØGM de la bibliothèque IBM.

2.1. Réservation de place sur disque.

2.1.1. Fichier partitionné. Il s'agit par exemple de créér une bibliothèque de programmes mis sur disques et constituant un fichier partitionné. On utilise alors la procédure RESER.

2.1.1.1. Procédure RESER : Cette procédure réserve sur le volume désiré la place du fichier partitionné nécessaire au stockage ultérieur des membres de ce fichier.

Son emploi est très simple puisqu'il suffit d'une seule carte EXEC :

```
//R1bEXECbRESER,FICHIER='par1',UNIT='par2',VØLUME=par3,
//   PLACE=par4
```

avec : par 1 = nom du fichier partitionné selon les conventions CIRCE (voir 3ème partie chapitre 7 paragraphe 7.2.1)

par 2 = type d'unité : 3330 pour les unités 3330-1
3330-1 pour les unités 3330-11

par 3 = nom du volume choisi (ex : RES301 ...)

par 4 = nombre de pistes à réserver (allocation primaire)

l'allocation secondaire sera 15 fois 1 piste (cette allocation se fera lorsque toute l'allocation primaire sera épuisée dans la mesure de la place encore disponible sur le volume). En plus 16 emplacements de 256 octets seront réservés pour le DIRECTORY permettant le stockage d'environ 128 noms de membres.

exemple : On désire réserver 10 pistes sur le volume RES304 pour le fichier partitionné dont le DSN est WRR1358.RØCHEMAR.FICH1

```
// carte JØB
//E1bEXECbRESER,FICHIER='WRR1358.RØCHEMAR.FICH1',
//  VØLUME=RES304,UNIT='3330-1',PLACE=10
//
```

Dans cet exemple, on réserve 10 pistes d'allocation primaire, 1 piste d'allocation secondaire et 16 blocs de 256 octets pour le directory.

On obtient sur le fichier d'édition SYSPRINT le message suivant :

```
SYSTEM UTILITIES ..... IEHPROGM
UTILITY END
```

En cas d'anomalie, on obtient dans le SYSMSG :

- soit : SPACE NOT AVAILABLE : il n'y a pas assez de place disponible sur le volume choisi.

- soit : DUPLICATED NAME ON THE VOLUME : le nom choisi pour le fichier partitionné existe déjà sur le volume considéré.

2.1.1.2. Création d'une bibliothèque de programmes.

Supposons que l'on désire créér une bibliothèque de programmes FORTRAN ou COBOL sur disque.
On peut effectuer la compilation et le link-edit par utilisation des procédures cataloguées FTXCL (programmes FORTRAN) ou CBACL (programmes COBOL). Ces procédures donnent le "load module" (programme compilé et linké) qui constitue un fichier défini par la carte //LKED.SYSLMØD DD.

Ensuite chaque load module ainsi sauvegardé constituera un MEMBRE d'un fichier partitionné créé auparavant à l'aide de la procédure RESER.
Ce fichier partitionné constitue alors une bibliothèque de programmes directement exécutables (c'est-à-dire au niveau de l'étape GØ).

Soit par exemple WRR1358.RØCHEMAR.FICH1 le DSN du fichier partitionné créé par la procédure RESER dans un JØB précédent (voir parag. 2.1.1. 1.) et devant constituerla bibliothèque de programmes (sur disque RES304). Soit P301 le nom de membre que l'on attribue au load module du programme source FORTRAN PØH301 que l'on désire placer dans cette bibliothèque. On procédera de la façon suivante :

```
// carte JØB
//E1bEXECbFTXCL
//FØRT.SYSINbDDb*
     ..............
     programme source PØH 301 (FØRTRAN) sur cartes
     ..............
/*
//LKED.SYSLMØDbDDbUNIT=3330-1,VØL=SER=RES304,
//  DISP=ØLD,DSN=WRR1358.RØCHEMAR.FICH1(P301)
//
```

L'utilisation ultérieure de ce programme se fera par le paramètre PGM de la carte EXEC et référence à la bibliothèque à l'aide d'une carte

```
//STEPLIBbDD  .....  si définition au niveau de l'étape
ou
//JØBLIB bDD  .....  si définition au niveau du JØB.
```

exemple : utilisation du programme PØH301 mis sur disque avec données sur cartes.

```
// carte JØB
//E1bEXECbPGM=P301,REGION=90K
//STEPLIBbDDbUNIT=3330-1,VOL=SER=RES304,DISP=OLD,
//    DSN=WRR1358.RØCHEMAR.FICH1
//FT05F001bDDb*
      | données sur cartes
/*
//FT06F001bDDbSYSØUT=A,DCB=(RECFM=VBA,LRECL=137,BLKSIZE=1922)
//FT07F001bDDbSYSØUT=B
/*
//
```

Notons que l'appel du programme par le paramètre PGM de la carte EXEC nécessite de préciser les fichiers 5 (lecture de cartes), 6 (sortie sur imprimante) et 7 (sortie de cartes perforées). Les cartes DD correspondantes ne sont pas, en effet, générées par le système comme c'est le cas lorsque l'on utilise une procédure cataloguée du type FTXCLG.

2.1.2. Fichier séquentiel : on utilise le programme utilitaire IEFBR14.

Le programme IEFBR14 est un programme de la bibliothèque du système. Pour l'utiliser on demande son chargement en mémoire par le paramètre PGM de la carte EXEC.

Il suffira de préciser dans une carte DD les caractéristiques du fichier pour lequel, on désire réserver de la place et le nom du volume sur lequel se fera cette réservation.

exemple : On désire réserver 5 pistes sur le disque RES304 pour y placer le fichier séquentiel dont le DSN est WRR1358.RØCHEMAR.FICH2

On procédera de la façon suivante :

```
// carte JØB
//E1bEXECbPGM=IEFBR14
//DD1bDDbUNIT=3330-1,DISP=(,KEEP,DELETE),
//  VØL=SER=RES304,DSN=WRR1358.RØCHEMAR.FICH2,
// SPACE=(TRK,(5,1) )
//
```

Dans ce cas, on réservera 5 pistes d'allocation primaire et une piste d'allocation secondaire.

2.2. Compression d'un fichier partitionné.

Au cours de remplacements successifs de membres existants d'un fichier partitionné le système perd la place des anciens membres sur disque. On est alors amené à comprimer le fichier pour récupérer toute la place inutilisée.

Pour réaliser cette opération, particulièrement intéressante pour la gestion de bibliothèques de programmes, on utilise la procédure cataloguée CØMPRIME (procédure CIRCE).

L'appel à cette procédure se fait par une seule carte EXEC :

```
//C1bEXECbCØMPRIME,FICHIER='par1',UNIT='par2',VØLUME=par3
```

avec par 1 = nom du fichier partitionné

par 2 = type d'unité : 3330 pour les unités 3330-1
3330-1 pour les unités 3330-11

par 3 = nom du volume sur lequel réside le fichier partitionné

exemple : On désire comprimer la bibliothèque WRR1358.RØCHEMAR.HYDRØM résidant sur RES302,
On procède de la façon suivante :

```
// carte JØB
//C1bEXECbCØMPRIME,FICHIER='WRR1358.RØCHEMAR.HYDRØM',
//   UNIT='3330-1',VØLUME=RES302
//
```

2.3. Suppression de tout ou partie d'un fichier.

2.3.1. Suppression d'un fichier séquentiel ou partitionné résidant sur disque.

Si on désire supprimer un fichier séquentiel ou partitionné du disque où il réside on peut utiliser la procédure EFFACE (procédure CIRCE)

L'appel se fait par une carte EXEC du type :

```
//F1bEXECbEFFACE,FICHIER='par1',UNIT='par2',VØLUME=par3
```

avec par 1 = nom du fichier à supprimer

par 2 = type d'unité : 3330 pour les unités 3330-1
3330-1 pour les unités 3330-11

par 3 = nom du volume sur lequel réside le fichier. La valeur par défaut de ce paramètre est PP3330

exemple : On désire supprimer le fichier WRR1358.RØCHEMAR.FICH1 résidant sur RES304.

On procède de la façon suivante :

```
// carte JØB
//GØMMEbEXECbEFFACE,FICHIER='WRR1358.RØCHEMAR.FICH1',
//     VOLUME=RES304,UNIT='3330-1'
//
```

On obtient dans la liste éditée du fichier SYSMSG :

```
WRR1358.RØCHEMAR.FICH1 ..................... DELETED
```

2.3.2. Suppression d'un membre d'un fichier partitionné.

Si on désire supprimer un membre d'un fichier partitionné, on utilise le programme utilitaire IEHPRØGM.

Ce programme a plusieurs fonctions, il permet de :

- Supprimer un fichier séquentiel ou partitionné ou un membre d'un fichier partitionné.
- Renommer un fichier séquentiel ou partitionné ou un membre d'un fichier partitionné.
- Mettre dans le catalogue le nom d'un fichier séquentiel ou partitionné.
- Réserver de la place sur M.A.S. pour un fichier séquentiel ou partitionné.

L'emploi de ce programme nécessite 2 sortes d'instructions :

a) instructions du langage de contrôle du système d'exploitation (J.C.L.)

- carte JØB usuelle
- carte EXEC :

```
//bEXECbPGM=IEHPRØGM
```

- carte définissant le fichier des messages du programme IEHPRØGM :

```
//SYSPRINTbDDbSYSØUT=A
```

- carte DD définissant le type d'unité et le nom du volume sur lequel IEHPRØGM va travailler :

```
//DD1bDDbUNIT=xxxx,VØL=SER=xxxxxx,DISP=ØLD
```

b) instructions de contrôle spécifiques à l'utilitaire IEHPRØGM :

Ces instructions sont annoncées par la carte //SYSINbDDb*

Elles sont donc perforées sur cartes

Pour la suppression d'un membre d'un fichier partitionné (ou d'un

fichier séquentiel) l'instruction de contrôle est la suivante :

b SCRATCHbDSNAME=par1,VØL=par2=par3,MEMBER=par4

avec par 1 = nom du fichier partitionné

par 2 = type d'unité { 3330 pour les unités 3330-1
3330-1 pour les unités 3330-11

par 3 = nom du volume sur lequel réside le fichier.

par 4 = nom du membre dans le cas d'un fichier partitionné.

exemple :
On désire effacer le membre SP1 du fichier WRR1358.RØCHEMAR.FICH1 résidant sur SET002. On procède de la façon suivante :

```
// carte JØB
//GbEXECbPGM=IEHPRØGM
//SYSPRINTbDDbSYSØUT=A
//DD1bDDbUNIT=3330,VØL=SER=SET002,DISP=ØLD
//SYSINbDDb*
bSCRATCHbDSNAME=WRR1358.RØCHEMAR.FICH1,VØL=3330=SET002,MEMBER=SP1
/*
//
```

On obtient sur le fichier SYSPRINT le résultat suivant :

```
SYSTEM SUPPØRT UTILITIES ..... IEHPRØGM
SCRATCH DSNAME=WRR1358.RØCHEMAR.FICH1,VØL=3330=SET002,MEMBER=SP1
NØRMAL END ØF TASK RETURNED FRØM SCRATCH
UTILITY END
```

Remarque : Pour effacer un fichier en fin de JØB, il est préférable d'utiliser le paramètre DISP = (OLD, DELETE) dans la carte DD qui définit le fichier.

Le scratch prendra effet à la fin de l'étape contenant la carte DD.

3. Analyse du contenu des supports magnétiques.

Il est très important de pouvoir analyser le contenu d'un support magnétique pour connaître le nom, les caractéristiques et l'organisation des fichiers qu'il contient et éventuellement de pouvoir lister sur imprimante les enregistrements.

Pour ces opérations, l'utilisateur du CIRCE dispose des procédures cataloguées DUDUMP et DUMPBAND (CIRCE) pour l'analyse des bandes magnétiques et de la procédure BØTUTI et du programme utilitaire IEBPTPCH pour les disques.

A ces utilitaires, il convient d'ajouter le programme IEBGENER qui permet la copie simple sur imprimante de tout fichier séquentiel ou d'un membre d'un fichier partitionné. Ce programme IEBGENER sera décrit dans le chapitre 4 relatif aux copies de fichiers.

3.1. Analyse d'une bande magnétique.

3.1.1. Caractéristiques et organisation des fichiers contenus sur la bande (DUDUMP).

Les caractéristiques et l'organisation des fichiers contenus sur une bande sont obtenues à l'aide de la procédure DUDUMP mise au point par le CIRCE.

La mise en oeuvre de cette procédure se fait de la façon suivante :

cas général : appel par une seule carte EXEC

```
// carte JØB usuelle
//D1bEXECbDUDUMP,ID=nnnnnn
//
```

avec nnnnnn = numéro de la bande à analyser.

Dans le cas particulier où l'on ne connaît pas le numéro de la bande (par exemple lorsque celle ci est attribuée dans le même JOB au cours d'une étape précédente) on procède alors de la façon suivante :

On ajoute une carte DD complémentaire GØ.DDTAPE DD ...
Dans laquelle, on utilise le paramètre REF pour faire référence à la carte DD qui définit le fichier qui a donné lieu à l'attribution de la bande que l'on désire analyser

exemple :

```
// carte JØB usuelle
//E1bEXECbFTXCLG
     ..........
     étape E1
     ..........
//GØ.FTO8FOO1bDDbUNIT=BD16,DISP=(NEW,KEEP,KEEP),
//   VØL=(,RETAIN),DSN=FICH1,DCB= ....
/*
//D1bEXECbDUDUMP
//GØ.DDTAPEbDDbVØL=REF=*.E1.GØ.FTO8FOO1
/*
//
```

Cette procédure donne en sortie sur imprimante les renseignements suivants :

en tête, le message :

*** EXAMEN DE LA BANDE nnnnnn ***

nnnnnn : numéro de la bande

ensuite si la bande contient des labels standard IBM :

- le label de volume c'est-à-dire le numéro de la bande

VØL=SER=nnnnnn

Pour chaque fichier contenu sur la bande :

- les labels de début du fichier avec les valeurs des paramètres suivants :
  - DSN
  - RECFM
  - LRECL
  - BLKSIZE
  - DEN
- une Tape Mark
- le contenu du fichier avec :
  - le nombre de blocs
  - la longueur du bloc le plus grand (en octets)
  - la longueur du bloc le plus petit (en octets)
- une Tape Mark

- les labels de fin de fichier avec :

  le nombre de blocs
  la date de création

- une Tape Mark

En fin d'analyse (dernier fichier) on a reproduction de la Tape Mark de fin d'enregistrement et impression du message suivant :

*** FIN DE L'EXAMEN DE LA BANDE nnnnnn ***

Remarque importante :

Lorsque l'on utilise la procédure DUDUMP, la bande n'est pas rebobinée en fin d'exécution. Il est donc conseillé d'effectuer cette opération en dernière étape d'un JØB.

3.1.2. Impression des enregistrements contenus sur une bande (DUMPBAND)

Cette impression peut être obtenue rapidement et simplement en utilisant la procédure DUMPBAND (procédure CIRCE).

La mise en oeuvre de cette procédure se fait par appel à l'aide d'une carte EXEC et avec la possibilité d'avoir différentes options :

```
// carte JØB usuelle
//D1bEXECbDUMPBAND,UNIT=par1,ØPT='par2',ID=par3
                                        + autres paramètres éventuels
//
```

avec par 1 = type d'unité bande (par défaut BD16)

par 2 = options :

ØPT = 'N,HEXA,I,BAD,S'

N : nombre inférieur ou égal à 999999 indiquant le nombre d'enregistrements physiques (blocs) désirés. Option par défaut : tout le fichier.

HEXA : indique que la représentation sera hexadécimale. Option par défaut : caractères.

I : nombre inférieur ou égal à 999 indiquant le nombre maximum d'enregistrements incorrects consécutifs que l'on tolère avant l'arrêt du programme.
Option par défaut : 10

BAD : on liste les seuls enregistrements incorrects. Option par défaut : tous les enregistrements corrects ou incorrects.

S : nombre inférieur ou égal à 999999 indiquant le bloc à partir duquel commence le listage.
Option par défaut 1.

Attention : - les 2 labels de début et les 2 labels de fin d'un fichier sont considérés chacun comme un bloc.
Par contre le label de volume n'est pas pris en considération.

- Tous les paramètres contenus dans ØPT sont repérés par leur position, donc toute opération par défaut doit être mentionnée par la virgule de séparation.

  • par 3 = numéro de la bande. Lorsque l'on ne connait pas ce numéro (fichier créé dans une étape précédente et mis sur une bande nouvellement attribuée) on peut opérer comme pour la procédure DUDUMP (parag.3.1.1.) en utilisant une carte complémentaire GØ.DDTAPE et en faisant référence à la carte DD correspondant au fichier ayant donné lieu à l'attribution de la bande à examiner.

On peut avoir d'autres sous-paramètres au niveau du paramètre ID. En particulier, on peut donner des précisions sur le DCB (conversion, parité, densité ...). Dans ce cas ces paramètres sont mis entre quotes.

exemples d'utilisation.

exemple 1.

```
// carte JØB usuelle
//D1bEXECbDUMPBAND,ID=103345,ØPT='100,HEXA,,,10'
//
```

On analyse la bande 103345 en imprimant en sortie 100 blocs à partir du 11ème et sous forme hexadécimale.
Tous les enregistrements sont imprimés sauf dans le cas, où il y aurait plus de 10 enregistrements incorrects consécutifs.

exemple 2.

```
// carte JØB usuelle
//D1bEXECbDUMPBAND,UNIT=BDE7,ØPT=',,,,2999!,ID='009761,
//      DCB=(TRTCH=ET,DEN=2)
//
```

On analyse la bande 009761 (bande à 7 canaux) on imprime ce qu'il y a sur cette bande en représentation EBCDIC à partir du 3000ème enregistrement physique.

Exemple 3.

```
// carte JØB usuelle
//D1bEXECbDUMPBAND,ØPT=',HEXA,,BAD',IB=103345
//
```

On imprime en hexadécimal, les enregistrements physiquement incorrects

3.2. Analyse du contenu d'un disque.

Un disque, à la différence d'une bande, est en général utilisé par plusieurs utilisateurs.

Il est donc intéressant de pouvoir imprimer la table des matières (VTØC) d'un disque et également les listes des fichiers qu'il contient.

La liste VTØC est réalisée à l'aide de la procédure CIRCE : BØTUTI, les listes des fichiers sont réalisées par le programme utilitaire standard IBM : IEBPTPCH

3.2.1. Liste de la VTØC (table des matières) d'un disque.

On utilise la procédure BØTUTI qui est appelée à l'aide d'une carte EXEC.

```
// carte JØB  usuelle
//bEXECbBØTUTI,PARM=par1,V='SER=par2',U='par3'  ,D='par4'
//
```

par 1 : mot clef définissant la fonction demandée :

PARM = VTØCS donne la liste simple c'est-à-dire sans la liste des membres des fichiers partitionnés.
Pour chaque fichier du volume, on obtient en clair :

- le DSØRG
- le RECFM
- le BLKSIZE
- le LRECL
- le nombre d'EXTENTS (extensions-allocation secondaire)
- la date de création
- l'emplacement physique du fichier sur le volume
- l'allocation faite pour ce fichier
- la place réellement utilisée.

PARM = VTØCD donne la même liste avec en plus pour les fichiers partitionnés le nom des membres appartenant à ce fichier.

par 2 : nom du volume pour lequel, on demande la liste VTØC

exemple : V = 'SER = SET002'

par 3 : type d'unité sur laquelle est monté le volume :

3330 unités 3330-1
3330-1 unités 3330-11

par 4 : nom du fichier pour lequel, on demande la liste VTØC

Attention : Si ce paramètre est omis toute la VTØC est imprimée

3.2.2. Liste hexadécimale d'un fichier séquentiel ou partitionné.

La représentation hexadécimale est surtout utile pour lister des données écrites en binaire.

Cette liste peut être obtenue à l'aide de la procédure BØTUTI. Elle se présente en sortie sous la forme de 120 caractères par ligne avec 2 blancs insérés tous les 8 caractères (il s'agit en fait de la forme standard des sorties du programme IEBPTPCH).

Pour obtenir cette liste, on appelle la procédure BØTUTI de la façon suivante :

```
// carte JØB usuelle
//bEXECbBØTUTI  ,PARM=par1  ,V='SER=par2',U='par3',D='par4'
//
```

par 1 : mot clef définissant la fonction demandée.

PARM = FICPH donne la liste hexadécimale du fichier partitionné.

Option par défaut : liste hexadécimale du fichier séquentiel (pas de paramètre PARM).

par 2 : nom du volume sur lequel réside le fichier.

par 3 : type d'unité sur laquelle est monté le volume :

3330 pour les unités 3330-1
3330-1 pour les unités 3330-11

par 4 : nom du fichier.

3.2.3. Liste en EBCDIC d'un fichier séquentiel ou partitionné.

Pour obtenir cette liste, on utilise également la procédure BØTUTI.

Seuls les 80 premiers caractères de chaque enregistrement logique sont imprimés en EBCDIC. La présentation est de 80 caractères par ligne.

L'appel de la procédure se fait de la façon suivante :

```
// carte JØB usuelle
//bEXECbBØTUTI,PARM=par1,V='SER=par2',U='par3',D='par4'
//
```

par 1 : mot clef définissant la fonction demandée.

PARM = FICSE donne la liste du fichier séquentiel en EBCDIC
PARM = FICPE donne la liste du fichier partitionné en EBCDIC.

par 2 : nom du volume sur lequel réside le fichier.

par 3 : type d'unité sur laquelle est monté le volume :

3330 pour les unités 3330-1
3330-1 pour les unités 3330-11.

par 4 : nom du fichier.

3.2.4. Exemple d'utilisation de la procédure BØTUTI

```
// carte JØB usuelle
//E1bEXECbBØTUTI,PARM=VTØCS,V='SER=RES302',U='3330-1'
//E2bEXECbBØTUTI,PARM=VTØCD,V='SER=RES302',U='3330-1',
//   D='WRR1358.RØCHEMAR.HYDROM'
//E3bEXECbBØTUTI,V='SER=PP3330',U='3330-1',
//   D='WRR1358.RØCHEMAR.FICH1'
//E4bEXECbBØTUTI,PARM=FICSE,V='SER=PP3330',U='3330-1',
//   D='WRR1358.RØCHEMAR.FICH1'
//E5bEXECbBØTUTI,PARM=FICPE,V='SER=RES302',U='3330-1',
//   D='WRR1358.RØCHEMAR.HYDRØM'
```

Dans cet exemple

E1 donne la liste VTØC simple du disque RES302.

E2 donne la liste VTØC avec le nom des membres du fichier partitionné WRR1358.RØCHEMAR.HYDROM.

E3 donne la liste hexadécimale du fichier séquentiel WRR1358.RØCHEMAR.FICH1 résidant sur PP3330

E4 donne la liste EBCDIC du fichier séquentiel WRR1358.RØCHEMAR.FICH1 résidant sur PP3330

E5 donne la liste en EBCDIC (80 caractères de chaque enregistrement logique) du fichier partitionné WRR1358.RØCHEMAR.HYDRØM.

3.2.5. Utilisation du programme utilitaire IEBPTPCH

3.2.5.1. Généralités

Ce programme utilitaire IBM est employé pour imprimer ou perforer le contenu :

- d'un fichier séquentiel ou partitionné,
- d'un ou plusieurs membres d'un fichier partitionné,
- de certains enregistrements d'un fichier séquentiel ou partitionné,
- du directory (répertoire du disque),

Remarque : chaque enregistrement est imprimé par groupe de 8 caractères séparés par 2 blancs ; au maximum on a 12 groupes (soit 96 caractères) par ligne. Cette option standard d'impression peut être modifiée par emploi de l'option RECØRD.
La fin d'un enregistrement logique est signalée à l'impression par *.
La fin d'un enregistrement physique est signalée par **.
La perforation standard est sur 80 colonnes.

3.2.5.2. Structure générale de l'IEBPTPCH

Elle est de la forme :

```
// carte JØB usuelle
//bEXECbPGM=IEBPTPCH
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDb ....
//SYSUT2bDDb ....
//SYSINbDDb*
    | instructions de contrôle spécifiques à l'utilitaire IEBPTPCH
/*
//
```

Où - SYSPRINT définit le fichier des messages générés par IEBPTPCH

- SYSUT1 définit le fichier à imprimer ou à perforer (fichier sur disque ou sur bande).
- SYSUT2 définit soit l'imprimante soit la perforatrice de cartes

  pour lister sur imprimante : SYSUT2bDDbSYSØUT=A
  pour perforer des cartes : SYSUT2bDDbSYSØUT=B

Remarques : - en entrée (SYSUT1) on peut avoir des enregistrements de longueur fixe, variable, indéfinie ou des enregistrements spannés.

- en sortie (SYSUT2) les enregistrements ne peuvent pas être bloqués.

3.2.5.3. Instructions de contrôle spécifiques à IEBPTPCH

a) Paramètre PRINT et PUNCH

PRINT indique que l'on désire imprimer le fichier

PUNCH indique que l'on désire perforer le fichier

- Ces paramètres sont placés en tête et sont précédés d'au moins un blanc et suivis ou non d'opérandes qui sont des sous-paramètres à option.

Le format est le suivant :

ou

bPRINTb opérandes
bPUNCHb opérandes

Paramètres	Opérandes = Sous-Paramètres à option	
ou PRINT ou PUNCH	[TYPØRG=yy], [TØTCØNV=zz] , [CNTRL=n], [STØPAFT=n], [STRTAFT=n], [SKIP=n], [MAXNAME=n], [MAXFLDS=n], [MAXGPS=n] , [MAXLITS=n],	applicables à PRINT ou à PUNCH
	[INITPG=n] , [MAXLINE=n],	applicables à PRINT seulement.
	[CDSEQ=n], [CDINCR=n] ,	applicables à PUNCH seulement

) Sous-Paramètres applicables à PRINT ou à PUNCH

- TYPØRG=yy

  yy=PS (option prise par défaut) indique que le fichier d'entrée est en organisation séquentielle.

  yy=PØ indique que le fichier d'entrée est en organisation partitionnée.

- TØTCØNV=zz

  zz=XE indique que l'on désire une sortie en hexadécimal.

  zz=PZ indique que l'on désire une sortie en décimal étendu à partir de données "packées".
  Option par défaut = représentation alphanumérique.

- CNTRL=n indique l'espacement (ou le casier du perforateur) pour la sortie :

  n = 1 : espacement simple (impression) ou casier 1 (perforation) c'est l'option par défaut.
  n = 2 : espacement double (impression) ou casier 2 (perforation)
  n = 3 : espacement triple (uniquement à l'impression).

- STØPAFT=n indique le nombre d'enregistrements logiques devant être imprimés ou perforés.
  Option par défaut : tout le fichier.

- STRTAFT=n indique le nombre d'enregistrements logiques à sauter avant impression ou perforation.
  En format VS et VBS, il s'agit du nombre de blocs à sauter.

- SKIP=n indique une impression ou perforation sélective. Un seul enregistrement sera traité tous les n enregistrements.
  Par défaut : pas d'échantillonnage.

- MAXNAME=n indique le nombre de cartes MEMBER spécifiées dans le cas de fichiers partitionnés.

- MAXFLDS=n indique le nombre total de paramètres FIELD dans toutes les cartes RECØRD.
  La valeur par défaut est zéro.

- MAXGPS=n indique le nombre de paramètres IDENT (dans des cartes RECØRD successives).

- MAXLITS=n indique le nombre de caractères des zones spécifiées dans les paramètres IDENT des cartes RECØRD.

) Sous-paramètres valables avec PRINT seulement.

- INITPG=n indique la numérotation des pages, n = valeur initiale du compteur. Option par défaut n = 1.

- MAXLINE=n indique le nombre maximum de lignes à imprimer par page. Option par défaut n = 60.

) Sous-paramètres valables avec PUNCH seulement.

- CDSEQ=n indique la numérotation des cartes des colonnes 73 à 80, n est la valeur initiale du compteur; la valeur par défaut est n = 00000000.

- CDINCR=n incrément pour la numérotation des cartes; option par défaut n = 1.

b) Paramètre TITLE.

Ce paramètre sert à l'impression ou à la perforation d'un titre. Il est optionnel, mais quand la carte TITLE existe, elle suit obligatoirement la carte PRINT (ou PUNCH).

Le format est le suivant :

```
bTITLEbITEM = ('titre' [,s] )
```

avec : titre = titre à imprimer (ou perforer) $\leqslant$ 40 octets.

s = rang du 1er octet du titre à éditer. Option par défaut 1.

c) Paramètre MEMBER.

Ce paramètre sert à sélectionner un membre d'un fichier partitionné, on peut mettre plusieurs cartes MEMBER si l'on veut sélectionner plusieurs membres.
Cette carte est facultative. Le format est le suivant :

```
bMEMBERbNAME = par 1
```

Où par 1 est le nom du membre à considérer.

d) Paramètre RECØRD.

Ce paramètre sert à définir un groupe d'enregistrements (ou un enregistrement) qui doit être imprimé (ou perforé) suivant les spécifi-

cations de l'utilisateur. Le format est le suivant :

bRECØRDb[IDENT = (l1,'valeur',P1)] [,FIELD=(l2,P2,C,P3)]

IDENT : identifie le dernier enregistrement à éditer. Il ne doit y avoir qu'un seul IDENT par carte RECØRD.
En l'absence d' IDENT tout le fichier sera édité.

l1 = longueur en octets de l'identificateur ($\leq$ 8 caractères)

'valeur' = nom de l'identificateur

P1 = rang de l'octet où commence l'identificateur dans l'enregistrement.

FIELD : définit les zones à sélecter dans l'enregistrement et les formats de sortie.

l2 = longueur de l'enregistrement à éditer (en octets $\leq$ 120)

P2 = rang du 1er octet à éditer (en entrée).
Option par défaut : 1

C = conversion éventuelle :

XE alphanumérique ⟶ hexadécimale
PZ décimal packé ⟶ décimal étendu

Option par défaut : pas de conversion (copie de l'entrée sur la sortie).

P3 = rang du 1er octet à éditer dans l'enregistrement de sortie.

3.2.5.4. Exemples d'utilisation de l' IEBPTPCH.

exemple 1. impression du fichier WRR1358.RØCHEMAR.FICH1 résidant sur le disque RES302 et dont le DCB est :
DCB = (RECFM=FB,LRECL=80,BLKSIZE=800).

```
// carte JØB usuelle
//bEXECbPGM=IEBPTPCH
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDSN=WRR1358.RØCHEMAR.FICH1,VOL=SER=RES302,
//   UNIT=3330-1,DISP=ØLD
//SYSUT2bDDbSYSØUT=A
//SYSINbDDb*
bPRINT
/*
//
```

exemple 2. impression du même fichier avec représentation hexadécimale des caractères :

```
// carte JØB usuelle
//bEXECbPGM=IEBPTPCH
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDSN=WRR1358.RØCHEMAR.FICH1,VØL=SER=RES302,
//   UNIT=3330-1,DISP=ØLD
//SYSUT2bDDbSYSØUT=A
//SYSINbDDb*
bPRINTbTØTCØNV=XE
/*
//
```

exemple 3. impression du même fichier avec édition s s les 2 blancs de séparation :

```
// carte JØB usuelle
//bEXECbPGM=IEBPTPCH
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDSN=WR1358.RØCHEMAR.FICH1,VØL=SER=RES302,
//   UNIT=3330-1,DISP=ØLD
//SYSUT2bDDbSYSØUT=A
//SYSINbDDb*
bPRINTbMAXFLDS=1
bRECØRDbFIELD=(80)
/*
//
```

On a indiqué MAXFLDS=1 car il existe 1 seul sous paramètre FIELD.
Ce dernier vaut 80 (LRECL = 80 dans le DCB de SYSUT1).

exemple 4. perforation sur cartes du même fichier avec séquentiation des cartes à partir de la carte 1000 avec une incrémentation de 100.

```
// carte JØB usuelle
//bEXECbPGM=IEBPTPCH
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDSN=WRR1358.RØCHEMAR.FICH1,VØL=SER=RES302,
//   UNIT=3330-1,DISP=ØLD
//SYSUT2bDDbSYSØUT=B
```

```
//SYSINbDDb*
bPUNCHbCDSEQ=1000,CDINCR=100
/*
//
```

exemple 5. impression standard des membres SS1 et SS2 du fichier partitionné WRR1358.RØCHEMAR.FIC.PART résidant sur SET002 et de DCB = (RECFM=FB,LRECL=80,BLKSIZE=8000).

```
//  carte JØB usuelle
//bEXECbPGM=IEBPTPCH
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDSN=WRR1358.RØCHEMAR.FIC.PART,VØL=SER=SET002,
//    UNIT=3330,DISP=ØLD
//SYSUT2bDDbSYSØUT=A
//SYSINbDDb*
bPRINTbTYPØRG=PØ,MAXNAME=2
bMEMBERbNAME=SS1
bMEMBERbNAME=SS2
/*
//
```

exemple 6. impression avec édition sans les 2 blancs de séparation des membres SS1 et SS2 du fichier partitionné précédent.

```
// carte JØB usuelle
//bEXECbPGM=IEBPTPCH
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDSN=WRR1358.RØCHEMAR.FIC.PART,VØL=SER=SET002,
//    UNIT=3330,DISP=ØLD
//SYSUT2bDDbSYSØUT=A
//SYSINbDDb*
bPRINTbTYPØRG=PØ,MAXNAME=2,MAXFLDS=2
bMEMBERbNAME=SS1
bRECØRDbFIELD=(80)
bMEMBERbNAME=SS2
bRECØRDbFIELD=(80)
/*
//
```

3.2.5.5. Remarque :

Si au cours d'un même JØB, on demande 2 étapes de IEBPTPCH, l'une pour la liste sur imprimante, l'autre pour la perforation de cartes, on doit ajouter les cartes FØRMAT suivantes :

//*FØRMATbPRbDDNAME=SYSUT2

avant l'étape d'impression de la liste et

//* FØRMATbPUbDDNAME=SYSUT2

avant l'étape de perforation.

4. Copie des Fichiers.

La copie d'un fichier sur un autre support peut être facilement obtenue à l'aide du programme utilitaire IBM IEBGENER.

Pour les fichiers résidant sur bande magnétique, il existe en outre une procédure CIRCE très simple permettant leur copie sur une autre bande. Il s'agit de la procédure CØPBDE

4.1. Le programme utilitaire IEBGENER.

4.1.1. Généralités. Ce programme permet les opérations suivantes :

- copier un fichier séquentiel ou un membre d'un fichier partitionné
- copier avec changement de blocage des enregistrements des fichiers créésen format V ou F.
- copier avec édition de tout ou d'une partie seulement d'un fichier, cette copie se faisant éventuellement pour chaque enregistrement :

  avec ou sans suppression d'une partie de l'enregistrement
  avec ou sans organisation des octets
  avec ou sans conversion des caractères
  avec ou sans rajout de caractères

- copier avec changement d'organisation : à partir d'un fichier séquentiel création de 1 ou plusieurs membres d'un fichier partitionné.

4.1.2. Langage de contrôle nécessaire. Le programme IEBGENER est utilisé avec des cartes de contrôle système et des cartes paramètres qui lui sont spécifiques.
La structure générale est de la forme suivante :

```
//  carte JØB usuelle
//bEXECbPGM=IEBGENER
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDb .........
//SYSUT2bDDb .........
//SYSINbDDb*
    | instructions de contrôle spécifiques à l'utilitaire
    | IEBGENER
/*
//
```

où
- SYSPRINT définit le fichier des messages de IEBGENER
- SYSUT1 définit le fichier d'entrée à recopier
- SYSUT2 définit le fichier de sortie sur lequel sera copié le fichier d'entrée.

Remarque : Pour la copie simple et la copie avec changement de blocage des enregistrements, il n'y a pas besoin d'instructions de contrôle spécifiques.
Dans ce cas, on a donc

```
//SYSINbDDbDUMMY
```

4.1.3. Mise en oeuvre du programme.

4.1.3.1. Copie simple d'un fichier. Dans ce cas aucune instruction de contrôle spécifique n'est nécessaire; IEBGENER permet alors très facilement un changement de support.

exemple 1 : Copie sur imprimante d'un paquet de cartes

```
//  carte JØB  usuelle
//*FØRMATbPR,DDNAME=SYSUT2,CØNTRØL=SINGLE
//bEXECbPGM=IEBGENER
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDATA
    |cartes à lister pouvant contenir des cartes // en colonne 1 et 2
/*
//SYSUT2bDDbSYSØUT=A,DCB=BLKSIZE=80
//SYSINbDDbDUMMY
//
```

Dans cet exemple, on utilise une carte FØRMAT avec le paramètre CØNTRØL = SINGLE pour éviter d'avoir le saut programmé et pour obtenir un espacement simple interligne en sortie.

exemple 2. : Copie sur perforateur d'un paquet de cartes :

```
// carte JØB  usuelle
//bEXECbPGM=IEBGENER
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDATA
    |cartes à dupliquer pouvant contenir des cartes // en colonnes
    |1 et 2
```

```
/*
//SYSUT2bDDbSYSØUT=B,DCB=BLKSIZE=80
//SYSINbDDbDUMMY
//
```

exemple 3. : Copie sur le disque PP3330 du paquet de cartes précédent et création d'un fichier séquentiel WRR1358.RØCHEMAR.FICH1.

```
// carte JØB  usuelle
//bEXECbPGM=IEBGENER
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDATA
    | cartes à écrire sur le disque pouvant comporter des cartes // en colonnes
    | 1 et 2
/*
//SYSUT2bDDbUNIT=3330-1,VØL=SER=PP3330,DISP=(,KEEP,DELETE),
//    DSN=WRR1358.RØCHEMAR.FICH1,SPACE=(TRK,(1,1) ),
//    DCB=(BLKSIZE=6400,LRECL=80,RECFM=FB)
//SYSINbDDbDUMMY
//
```

exemple 4. : Copie sur le disque SET002 du paquet de cartes précédent et création du membre SP1 du fichier partitionné WRR1358.RØCHEMAR.FICH2.PART créé sur ce disque.

```
// carte JØB usuelle
//bEXECbPGM=IEBGENER
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDATA
    | cartes pouvant comporter des cartes // en colonnes 1 et 2
/*
//SYSUT2bDDbUNIT=3330,VØL=SER=SET002,DISP=(,KEEP,DELETE),
//  DSN=WRR1358.RØCHEMAR.FICH2.PART(SP1),SPACE=(TRK,(1,1,1) ),
//  DCB=(RECFM=FB,LRECL=80,BLKSIZE=6400)
//SYSINbDDbDUMMY
//
```

exemple 5. : Copie du 3ème fichier (label 3) dont le DSN est FICH3 et résidant sur la bande n° 102808 sur une nouvelle bande (numéro non connu

de l'utilisateur) en position de 1er fichier (label 1). L'opération sera faite sans modification du blocage.

```
// carte JØB usuelle
//bEXECbPGM=IEBGENER,REGIØN=90K
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbUNIT=BD16,VØL=SER=102808,DISP=(ØLD,KEEP),
//    DSN=FICH3,LABEL=(3,SL),DCB=(RECFM=FB,LRECL=80,BLKSIZE=12800)
//SYSUT2bDDbUNIT=BD16,DISP=(NEW,KEEP,DELETE),DSN=NFIC1,
//    LABEL=(1,SL),DCB=(RECFM=FB,LRECL=80,BLKSIZE=12800)
//SYSINbDDbDUMMY
//
```

Remarques : - Pour cet exemple de copie de bande à bande, il est plus simple d'utiliser la procédure CØPBDE qui sera décrite dans le paragraphe 4.2.

- Dans cet exemple, il est nécessaire de demander 90K pour le paramètre REGION compte tenu des forts blocages (13K) des fichiers SYSUT1 et SYSUT2. On a environ 2X2X13 = 52K nécessaires pour les entrées/ sorties.

4.1.3.2. Copie avec changement de blocage.

Cette opération n'est valide que pour des fichiers créés en format F ou V.

Ici encore aucune instruction spécifique à l'IEBGENER n'est nécessaire, on a donc //SYSINbDDbDUMMY

Exemple 1 : Copie du fichier WRR1358.RØCHEMAR.FICH1 résidant sur SET002 avec DCB = (BLKSIZE=800,LRECL=80,RECFM=FB) sur WRR1358.RØCHEMAR.ØPT résidant sur le même disque et avec DCB = (BLKSIZE = 6400,LRECL = 80, RECFM = FB).

```
//  carte JØB usuelle
//bEXECbPGM=IEBGENER
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbUNIT=3330,VØL=SER=SET002,DISP=ØLD,
//      DSN=WRR1358.RØCHEMAR.FICH1
//SYSUT2bDDbUNIT=3330,VØL=SER=SET002,DISP=(,KEEP,DELETE),
//     DSN=WRR1358.RØCHEMAR.ØPT,SPACE=(TRK,(3,1) ),
//     DCB=(BLKSIZE=6400,LRECL=80,RECFM=FB)
//SYSINbDDbDUMMY
//
```

exemple 2 : Copie avec changement de blocage de 2 fichiers séquentiels (FICH1 et FICH2) résidant sur la bande 103871, sur une nouvelle bande dont le numéro n'est pas connu de l'utilisateur.

```
// carte JØB usuelle
//CØP1bEXECbPGM=IEBGENER,REGIØN=90K
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbUNIT=BD16,VØL=(,RETAIN,SER=103871),DISP=ØLD,
//     DSN=FICH1,LABEL=(1,SL),
//     DCB=(RECFM=FB,LRECL=80,BLKSIZE=1920)
//SYSUT2bDDbUNIT=BD16,VØL=(,RETAIN),DISP=(NEW,KEEP,DELETE),
//     DSN=FICH1,LABEL=(1,SL),
//     DCB=(RECFM=FB,LRECL=80,BLKSIZE=12800)
//SYSINbDDbDUMMY
/*
//CØP2bEXECbPGM=IEBGENER,REGIØN=90K
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbUNIT=BD16,VØL=SER=103871,DISP=ØLD,
//     DSN=FICH2,LABEL=(2,SL),
//     DCB=(RECFM=FB,LRECL=80,BLKSIZE=1920)
//SYSUT2bDDbUNIT=BD16,VØL=REF=*.CØP1.SYSUT2,
//     DSN=FICH2,LABEL=(2,SL),
//     DCB=(RECFM=FB,LRECL=80,BLKSIZE=12800)
//
```

4.1.3.3. Copie avec édition.

Le programme IEBGENER permet de copier tout ou partie d'un fichier séquentiel avec ou sans modification des enregistrements :

- copie de tout/partie de chaque enregistrement
- copie avec/sans réorganisation des octets
- copie avec/sans conversion des caractères
- copie avec/sans rajout de caractères

Dans ce cas, on utilise les instructions spécifiques de IEBGENER : ces instructions sont constituées de 3 cartes paramètres qui doivent être mises (si nécessaire) dans l'ordre suivant :

- GENERATE
- MEMBER
- RECØRD

* ordre GENERATE : Forme générale :

bGENERATEb[MAXNAME=n1] [,MAXFLDS=n2] [,MAXGPS=n3] [,MAXLITS=n4]

MAXNAME=n1 : nombre de cartes MEMBER spécifiées (copie avec changement d'organisation voir paragraphe 4.1.3.4.)

MAXFLDS=n2 : nombre total de paramètre FIELD rencontrés dans les cartes RECØRD.

MAXGPS=n3 : nombre de paramètres IDENT rencontrés dans les cartes RECØRD.

MAXLITS=n4 : nombre de caractères littéraux contenus dans les paramètres FIELD.

L'ordre GENERATE est unique et le premier ordre d' IEBGENER.

* ordre MEMBER (voir paragraphe 4.1.3.4.) inutile dans le cas d'une copie avec édition.

* ordre RECØRD la carte RECØRD définit les informations d'édition. Forme générale :

bRECØRDb[IDENT=(par1,'par2',par3)] [,FIELD={(par1,par2,par3,par4) / (par5,par6,par7,par8)}]

IDENT : identifie le dernier enregistrement à éditer. En l'absence d'IDENT tout le fichier sera édité.

par 1 = longueur (en octets) de l'identificateur ($\leqslant 8$ octets)

'par2' = identificateur lui même

par 3 = rang de l'octet où commence l'identificateur dans l'enregistrement.

FIELD : Ce paramètre a 2 fonctions différentes donc 2 écritures possibles :

- 1re fonction : copie de tout ou partie de l'enregistrement avec/sans réorganisation de l'enregistrement à éditer et avec/sans conversion de caractères :

  par 1 : longueur de l'enregistrement à éditer. Valeur par défaut 80

  par 2 : rang de l'octet du début de l'enregistrement à éditer (en entrée) valeur par défaut 1

  par 3 : conversion demandée :

  par 3 = HE les caractères BCD sont traduits en EBCDIC

par 3 = PZ les caractères UNPACK sont traduits en PACK

par 3 = ZP les caractères PACK sont traduits en UNPACK

valeur par défaut : pas de conversion.

par 4 : rang du 1'octet de l'enregistrement à éditer (en sortie)
valeur par défaut 1

- 2ème fonction : rajout de caractères dans un enregistrement avec/sans conversion de caractères :

par 5 : nombre de caractères à rajouter dans chaque enregistrement à éditer (maximum = 40)

'par6' : les caractères eux mêmes à rajouter (maximum 40)

par 7 : conversion des caractères (identique à par 3 de la 1re fonction de FIELD)

par 8 : rang de l'octet où on désire insérer les caractères.

exemple : Copie de tout le fichier ; pour chaque enregistrement :

- copie des 72 premiers octets et conversion BCD ⟶ EBCDIC
- rajout du mot MAIN, octet 73 à 76.

```
// carte JØB usuelle
//*FØRMATbPRbDDNAME=SYSUT2,CØNTRØL=SINGLE
//bEXECbPGM=IEBGENER
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDATA
    | Cartes à éditer (code BCD) pouvant comporter des cartes // en colonnes
    | 1 et 2
/*
//SYSUT2bDDbSYSØUT=A,DCB=BLKSIZE=80
//SYSINbDDb*
bGENERATEbMAXFLDS=2,MAXLITS=4
bRECØRDbFIELD=(72,,HE,),FIELD=(4,'MAIN',,73)
/*
//
```

On obtient sur la SYSPRINT le résultat suivant:(en plus de la liste demandée)

```
DATA SET UTILITY - GENERATE
     GENERATE MAXFLDS=2,MAXLITS=4
     RECORD FIELD = (72,,HE,),FIELD = (4,'MAIN',,73)
PROCESSING ENDED AT EOD
```

4.1.3.4. Copie avec changement d'organisation.

IEBGENER permet à partir d'un fichier séquentiel de créér un ou plusieurs membre d'un fichier partitionné.

Les instructions de contrôle spécifiques sont dans ce cas :

bGENERATEbMAXNAME=n1 [,MAXGPS=n3]
bMEMBERbNAME=par1
bRECØRDb[IDENT= (par1,'par2',par3)]
b[MEMBER] ....
b[RECØRD] ....

- ordre GENERATE : 1er ordre d' IEBGENER et doit être unique

  MAXNAME=n1 nombre de cartes MEMBER spécifiées= nombre de membres à créér.

  MAXGPS=n3 nombre de paramètres IDENT de la carte RECØRD en l'absence de MAXGPS tout le fichier séquentiel formera un membre d'un fichier partitionné.

- ordre MEMBER : annonce le nom du membre à créér.

  Forme générale de la carte MEMBER :

bMEMBERbNAME = par 1

  par 1 = nom du membre à créér.

- ordre RECØRD : annonce l'identification du dernier enregistrement à considérer pour la création du membre dont on vient de donner le nom.

  Le paramètre IDENT de l'ordre RECORD a la même signification que dans le cas de la copie avec édition (voir paragraphe 4.1.3.3.) En l'absence de IDENT tout le fichier est pris en compte.

- le nouvel ordre MEMBER : si il existe annonce la constitution d'un deuxième membre dont le premier enregistrement sera celui qui suit immédiatement le dernier enregistrement qui a servi à créér le premier membre

  exemple : On désire créér les 2 membres SS1 et SS2 à partir d'un fichier séquentiel sur cartes comprenant deux sous-programmes FØRTRAN.

  SS1 sera composé du 1er sous-programme et se terminera par la carte END (END perforé en colonnes 7 à 9)

  SS2 sera le 2ème sous-programme.

Soit WRR1358.RØCHEMAR.FICH.PART le fichier choisi déjà créé résidant sur le disque SET002. On aura :

```
// carte JØB usuelle
//bEXECbPGM=IEBGENER
//SYSPRINTbDDbSYSØUT=A
//SYSUT1bDDbDATA
      |sous-programme SS1 sur cartes
/*
//SYSUT2bDDbUNIT=3330,VØL=SER=SET002,DISP=(MØD,KEEP),
//   DSN=WRR1358.RØCHEMAR.FICH.PART
//SYSINbDDb*
bGENERATEbMAXGPS=1,MAXNAME=2
bMEMBERbNAME=SS1
bRECØRDbIDENT=(3,'END',7)
bMEMBERbNAME=SS2
/*
//
```

On obtient dans la SYSPRINT le résultat suivant :

```
DATA SET UTILITY - GENERATE
GENERATE MAXGPS=1, MAXNAME=2
     MEMBER NAME=SS1
RECORD IDENT = (3, 'END', 7)
     MEMBER NAME=SS2
PROCESSING ENDED AT EOD
```

4.2. La procédure CØPBDE.

Cette procédure mise au point par le CIRCE permet la copie d'un fichier séquentiel résidant sur bande sur une autre bande. Elle n'est pas utilisable pour les autres supports.

- Son emploi est très simple puisqu'elle ne nécessite qu'une carte de contrôle EXEC :

```
// carte JØB  usuelle
//bEXECbCØPBDE,VØL1=nnnnnn,VØL2=nnnnnn,N=1,DSN=d
//
```

avec :

VØL1 = nnnnnn : numéro de la bande sur laquelle réside le fichier à copier

VØL2 = nnnnnn : numéro de la bande sur laquelle on désire copier le fichier

N = 1 : label du fichier (rang sur la bande).

DSN = d : nom du fichier (paramètre DSN de la carte DD)

les fichiers à copier doivent être précédés de standard-labels IBM.

exemple :

Copie de 3 fichiers IDMAL, ICMAL, PJMAL (placés en séquence sur la bande 108024), sur la bande 110325 :

```
// carte JØB  usuelle
//E1bEXECbCØPBDE,VØL1=108024,VØL2=110325,N=1,DSN=IDMAL
/*
//E2bEXECbCØPBDE,VØL1=108024,VØL2=110325,N=2,DSN=ICMAL
/*
//E3bEXECbCØPBDE,VØL1=108024,VØL2=110325,N=3,DSN=PJMAL
//
```

ANNEXE - 1 -
RAPPEL DE QUELQUES NOTIONS DE BASE ET DEFINITIONS

1 - REPRESENTATION INTERNE DE L'INFORMATION

1.1. Numération binaire - bit - octet.

Les ordinateurs utilisent la numération binaire ou numération à base 2.

On a donc 2 chiffres 0 ou 1 correspondant à 2 situations : oui ou non, le courant passe ou ne passe pas, ...

L'élément constitutif d'une donnée pouvant représenter l'une ou l'autre de 2 valeurs ou états distincts est appelé "bit"

Le bit (abréviation de binary digit) est le chiffre représentant l'un ou l'autre des nombres entiers zéro et un en numération binaire.

1 bit permet de représenter 2 nombres (2^1)

4 bits permettent de représenter 16 nombres (2^4)

8 bits permettent de représenter 256 nombres (2^8) ect ...

Un ensemble de bits consécutifs traités comme un tout constitue un multiplet (ou byte).

L'octet est un multiplet de 8 bits significatifs, chacun prenant la valeur 0 ou 1, ce qui donne 256 combinaisons possibles.

exemple : 0 1 0 1 0 0 1 1

On emploie le terme byte comme abréviation de "8 bit byte"

Le byte (ou octet) est la plus petite partie adressable de la mémoire (l'adresse est un indicatif désignant l'emplacement qu ' occupe une information mise en mémoire).

1.2. Codage d'un nombre décimal.

- Décimal Codé Binaire (DCB) ou Binary Coded Decimal (BCD). Il s'agit d'une méthode de codage consistant à transcrire un nombre décimal non point en bloc (comme on le fait en vrai binaire) mais chiffre par chiffre, chaque chiffre étant représenté par 4 bits.

On garde donc la structure du nombre décimal mais on code individuellement chacun de ses chiffres par son équivalent binaire. On utilise 4 bits afin de pouvoir coder les chiffres de 0 à 9.

exemple : Soit à coder le nombre décimal 19 :

On code 1 par 0001
et 9 par 1001

On aura donc en DCB : 19_{10} = 00011001

On aurait en binaire : $19_{10} = 10011_2$

Le code DCB permet également de représenter l'alphabet et d'autres signes conventionnels grâce à une table de correspondance :

Ainsi par exemple : A = 1101
B = 1110
ect ...

Pour avoir une représentation plus complète, on utilise un DCB à 6 bits qui permet de représenter 64 symboles.
Cependant si on désire une représentation des minuscules et des majuscules, on doit utiliser un code à 8 bits.

- Code EBCDIC ("Extended Binary Coded Decimal Interchange Code) Il s'agit d'un code à 8 bits de principe analogue au code DCB mais permettant 256 combinaisons. Ce code comporte 64 caractères. Le 370 IBM utilise comme code de perforation et code interne l' EBCDIC.

1.3. Représentation Hexadécimale.

La numération hexadécimale est une numération à base 16.

Pour exprimer les valeurs 10 à 15 on utilise les symboles A à F :

0 1 2 3 4 5 6 7 8 9 A B C D E F

Cette représentation facilite la lecture ou l'écriture de l'octet qui est alors représenté par 2 chiffres hexadécimaux.

On aura donc la correspondance suivante :

TABLEAU A1 - 1

Hexadécimal	Binaire
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Octet	Hexadécimal
00111010	3 A
11110010	F 2
10000101	8 5

En résumé :

1 octet = 2 fois 4 bits
4 bits = 16 combinaisons possibles
1 octet = 8 positions binaires ou 2 chiffres hexadécimaux.

La conversion Hexadécimal - Décimal est donnée dans la table A1 - 2

TABLEAU A1 - 2

TABLE DE CONVERSION HEXADECIMAL-DECIMAL											
COLONNES - HEXADECIMAL											
6		5		4		3		2		1	
HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC
0	0	0	0	0	0	0	0	0	0	0	0
1	1048576	1	65536	1	4096	1	256	1	16	1	1
2	2097152	2	131072	2	8192	2	512	2	32	2	2
3	3145728	3	196608	3	12288	3	768	3	48	3	3
4	4194304	4	262144	4	16384	4	1024	4	64	4	4
5	5242880	5	327680	5	20480	5	1280	5	80	5	5
6	6291456	6	393216	6	24576	6	1536	6	96	6	6
7	7340032	7	458752	7	28672	7	1792	7	112	7	7
8	8388608	8	524288	8	32768	8	2048	8	128	8	8
9	9437184	9	589824	9	36864	9	2304	9	144	9	9
A	10485760	A	655360	A	40960	A	2560	A	160	A	10
B	11534336	B	720896	B	45056	B	2816	B	176	B	11
C	12582912	C	786432	C	49152	C	3072	C	192	C	12
D	13631488	D	851968	D	53248	D	3328	D	208	D	13
E	14680064	E	917504	E	57344	E	3584	E	224	E	14
F	15728640	F	983040	F	61440	F	3840	F	240	F	15

Exemple : Conversion de 70A8 (Hexadécimal) en décimal

Colonne 1	8 ⟶	8
Colonne 2	A ⟶	160
Colonne 3	0 ⟶	0
Colonne 4	7 ⟶	28672
TOTAL		28840

1.4. Représentation "Décimal Condensé" (Packed decimal). Il s'agit d'une méthode de représentation dans laquelle plusieurs chiffres décimaux sont enregistrés dans un seul multiplet (par exemple 2 chiffres décimaux dans un seul octet).

On parle alors de données "packées" (ou condensées).

1.5. Représentation des nombres en mémoire (IBM 370). On appelle "mot" le plus petit groupe de positions de mémoire adressable dans son ensemble. La taille du mot varie selon les constructeurs. Chez IBM on a :

1/2 mot	=	2 octets
1 mot	=	4 octets
double mot	=	8 octets
quadruple mot	=	16 octets

- Représentation des nombres entiers : les nombres entiers (c'est-à-dire en virgule fixe) occupent en mémoire une zone de longueur fixe qui peut être un mot ou un demi-mot. Dans les 2 cas le premier bit (bit de gauche) est toujours le bit de signe.

  Les nombres positifs sont codés dans leur représentation binaire vraie avec le bit signe égal à 0.

  Les nombres négatifs sont représentés par leur complément à deux avec le bit signe égal à 1.
  Le complément à 2 s'obtient en inversant chaque bit du nombre et en ajoutant 1 au bit de droite.

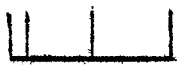
  exemple : Dans un format binaire d'un demi-mot (INTEGER*2)
  + 1 et - 1 s'écrivent respectivement :

  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

  Les nombres + 28_{10} et - 28_{10} s'écrivent :

  0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0
  1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0

  La valeur du plus grand nombre positif contenu dans un mot est $2^{31} - 1$.
  La valeur du plus petit nombre négatif contenu dans un mot est -2^{31}

REPRESENTATION D'UN NOMBRE ENTIER (INTEGER)		
Longueur en octets	Représentation interne	Plage des valeurs possibles
INTEGER 4	S  S = 0 Signe positif S = 1 Signe négatif	$-2147483648 \leqslant V \leqslant +2147483647$
INTEGER 2	S 	$-32768 \leqslant V \leqslant +32768$

- Représentation des nombres en virgule flottante : les nombres réels peuvent être représentés en virgule flottante hexadécimale simple précision, double précision ou quadruple précision.

  Le principe de la virgule flottante consiste à représenter un nombre réel (R) par le produit d'un nombre (M) compris entre 0 et 1 et d'une puissance de 10 (C) dans le système décimal et une puissance de 16 dans le système hexadécimal : R = M x C.

par exemple : $125,33_{10} = 0,12533 \times 10^3$

Un nombre flottant est donc représenté par

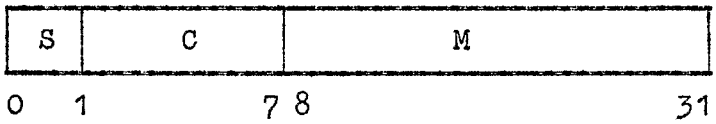
- sa mantisse M
- sa caractéristique C

En simple précision (REAL*4) un nombre flottant est représenté sur un mot de la façon suivante :

- Signe : bit numéro 0
  S = 0 pour les nombres positifs
  S = 1 pour les nombres négatifs.

- Caractéristique : bits 1 à 7

- Mantisse : bits 8 à 31



La caractéristique C qui est exprimée sur les 7 bits 1 à 7 a donc pour valeurs minimales et maximales : 0 et 127.
Cet intervalle est à diviser en deux pour représenter des exposants positifs et négatifs. Par convention on a :

Caractéristique	0	1	63	64	65	127
Exposant réel	-64	-63	-1	0	+1	+63

Donc l'exposant réel (E) est compris entre -64 et +63. Pour l'obtenir il suffit de retrancher 64 à la caractéristique :

par exemple : C = 1 ⟶ E = -63
C = 63 ⟶ E = -1

Pour représenter un nombre négatif on a 1 dans le bit signe, on ajoute donc 128 à la caractéristique. Par exemple un nombre négatif avec un exposant réel de +2 a pour caractéristique :

$$C = 2 + 64 + 128 = 194$$

Un nombre flottant simple précision a une représentation hexadécimale de 8 caractères hexadécimaux, les 6 caractères de droite représentant la mantisse et les 2 caractères de gauche la caractéristique.

En double précision (REAL * 8) ou quadruple précision (REAL * 16) les nombres en virgule flottante conservent la même configuration qu'en

simple précision mais avec une mantisse plus longue :

Soit 56 bits dans le cas où le nombre occupe 2 mots (double précision)
Soit 120 bits dans le cas où le nombre occupe 4 mots (quadruple précision).

Le signe est toujours dans le premier bit et la caractéristique est exprimée dans les 7 bits (1 à 7) suivant le signe.

Les valeurs extrêmes autorisées pour un nombre flottant sont de :

16^{-64} (environ 10^{-78}) et 16^{63} (environ 10^{75})

La précision est de :

* 6 chiffres hexadécimaux (environ 7,2 chiffres décimaux) pour 4 positions de mémoire (simple précision),

* 14 chiffres hexadécimaux (environ 16,8 chiffres décimaux) pour 8 positions de mémoire (double précision),

* 28 chiffres hexadécimaux (environ 35 chiffres décimaux) pour 16 positions de mémoire (quadruple précision).

REPRESENTATION D'UN NOMBRE REEL (REAL)			
Longueur en octets	Représentation interne	Valeurs possibles	Précision en chiffres décimaux
REAL 4	S C ← mantisse → C : caractéristique S : signe	$10^{-78} \leq \|x\| < 10^{+75}$	7 chiffres décimaux
REAL 8	La mantisse occupe 7 octets	$10^{-78} \leq \|x\| < 10^{75}$	16 chiffres décimaux
REAL 16	La mantisse occupe 15 octets	$10^{-78} \leq \|x\| < 10^{75}$	35 chiffres décimaux

1.6. Définition des variables en langage FORTRAN :

Une variable FORTRAN est la représentation symbolique d'une quantité placée dans une zone de mémoire.
La valeur indiquée par le nom est toujours la valeur qui se trouve dans cette mémoire à ce moment là.

Par convention en FORTRAN toutes les variables sont de type REAL et de longueur 4 octets sauf si le nom de la variable commence par I J K L M ou N auquel cas ces variables sont considérées comme étant de type INTEGER et de longueur 4 octets.

On peut cependant redéfinir la longueur et le type des variables :

- soit par l'instruction de spécification IMPLICIT

exemple IMPLICIT { INTEGER / REAL / COMPLEX / LOGICAL } * { 2 / 8-16 / 16-32 / 1 }

- soit par des instructions de spécification explicites

exemple : REAL*16A,IB*8(10)/3*1D0,5*0D0/

Remarque : Dans un programme l'instruction IMPLICIT doit être unique et placée en tête du programme principal. Dans un sous-programme cette instruction doit être unique et suivre la carte SUBROUTINE.

2 - DEFINITION DE QUELQUES TERMES UTILISES EN INFORMATIQUE.

Accès sélectif ou accès direct :

recherche d'un enregistrement dans un volume en fonction de son adresse et non en fonction des données recherchées ou enregistrées auparavant.

Adresse :
- indicatif désignant l'emplacement qu'occupe une information mise en mémoire ou qu'occupera une information qui doit être mise en mémoire.
- indicatif désignant une unité d'un ordinateur dans les instructions d'un programme.

Alphanumérique :

concerne une information dont la représentation est définie :
- soit par des caractères alphabétiques
- soit par des caractères numériques
- soit par des caractères spéciaux
- soit par un mélange de 2 ou de 3 de ces groupes de caractères.

Batch (lot) : ensemble des données destinées à être traitées en différé.

Bit : Elément constitutif d'une donnée pouvant représenter l'une ou l'autre de 2 valeurs ou états distincts.
Binary digit (=bit) chiffre représentant l'un des nombres entiers 0 ou 1 en numération binaire (voir paragraphe 1 de l'annexe 1).

Blanc : partie d'un support d'information ne portant pas de caractères.

Bloc (block) :

- en programmation groupe de caractères, de mots ou d'articles manipulé comme un tout
- sur une bande, désigne l'enregistrement physique compris entre 2 intervalles.

Buffer (mémoire tampon) :

mémoire utilisée pour compenser une différence de débit des informations lorsque ces informations sont transmises d'un organe à un autre.

Byte = octet (voir octet)

Canal (channel) :

organe qui connecte des unités entre elles et plus particulièrement des unités d'entrée - sortie à l'unité centrale.

Caractère (digit) : élément d'un ensemble employé conventionnellement pour constituer ou représenter des données. Des caractères peuvent être des lettres, des chiffres des signes de ponctuation ou d'autres symboles

Charger (to load) :

placer en mémoire des informations à partir d'un milieu externe à cette mémoire.
Dans le système d'exploitation 370, mettre en place, c'est-à-dire lire un module de chargement en mémoire principale en vue de son exécution.

Chargeur - éditeur de liens (linkage loader) :

programme du système d'exploitation 370 ayant les mêmes fonctions que l'éditeur de liens mais permettant en outre de regrouper, en une même étape de travail, l'édition des modules de chargement et leur exécution, sans nécessiter l'intervention du programme de contrôle des travaux.

Clé : un ou plusieurs caractères faisant partie d'un enregistrement et permettant de l'identifier.

Compilateur :

programme permettant de traduire un langage évolué en un programme en langage machine directement exécutable.

Concaténation : enchaînement.

Conversion (de données) :

transformation de données par changement de leur représentation sans modification des informations elles-mêmes.

Data (donnée) :

représentation conventionnelle d'une information sous une forme physique convenant à son traitement par des moyens automatiques.

Data set : ensemble de données.

Debug (to) :

mettre au point un programme.

DCB (code) :

Abréviation de "Décimal Codé Binaire" codage en 4 bits d'un chiffre décimal (voir paragr. 1 de l'annexe 1).

Décimal condensé (packed decimal) :

méthode de représentation dans laquelle plusieurs chiffres décimaux sont enregistrés dans un seul multiplet. (par exemple 2 chiffres décimaux dans un octet).

Deck (of cards) :

jeu de cartes formant un tout.

Délimiteur : caractère qui limite une suite de caractères et qui n'est pas membre de cette suite.

Densité d'enregistrement (packing density) :

nombre d'unités d'information enregistrées par unité de dimension sur un support donné.

Dispac (disk pack) :

marque déposée IBM désignant le chargeur de disques amovibles.

Dummy (factice) :

qualifie une adresse, une instruction ou un enregistrement sans valeur opératoire réelle.

Dump : image - mémoire.

To dump = faire une analyse mémoire. Enregistrer (imprimer) sur un support le contenu d'une mémoire et éventuellement les informations complémentaires associées.
exemple : dump d'une bande magnétique = analyse du contenu de la bande.

EBCDIC contraction de "Extended Binary Coded Decimal Interchange Code". (voir paragraphe 1 de l'annexe 1).

Editer : préparer l'impression afin d'obtenir une présentation satisfaisante

Editeur de liens (linkage editor) :

programme qui produit un module de chargement. Pour cela, il transforme des modules résultants en un format tel qu'ils puissent être recherchés et mis en place. Il créé un seul module de chargement en reliant les modules résultants produits séparement et les modules de chargement déjà obtenus.

Enregistrement (record) :

groupe d'informations constituant une unité pour le traitement. Dans le cas d'informations inscrites sur un support continu, on distingue :

- l'enregistrement logique : ensemble des informations ayant trait à un élément du fichier, et constituant l'unité logique de traitement pour le programme.
- l'enregistrement physique ou bloc : ensemble des informations comprises entre 2 limites et constituant l'unité technologique de transmission entre mémoire externe et mémoire interne.

Exécuter (to execute) :

accomplir la fonction demandée par une instruction ou une routine.

Facteur de groupage (blocking factor) :

nombre d'enregistrements logiques par enregistrement physique défini ou choisi par l'utilisateur.

Fichier : - ensemble d'enregistrements apparentés traités comme un tout
- ensemble organisé de documents différents ayant un objet commun.

Format : sur un support d'informations, décrit la représentation matérielle des informations et leur juxtaposition, indépendamment de leur représentation codée.

Fusionner (to Merge) :

combiner en un seul fichier deux ou plusieurs fichiers selon un ordre donné et défini à l'avance.

Gap : (=entre enregistrement) :

intervalle séparant 2 enregistrements sur un support continu.

Hardware (matériel de traitement de l'information) :

Ensemble des machines de traitement de l'information ou de leurs parties constituantes, par opposition aux programmes et autres moyens abstraits d'emploi de ces machines.

Hexadécimale (représentation) :

représentation des nombres dans un mode de numération de base 16. (voir paragraphe 1. Annexe 1).

Identificateur :

en programmation nom symbolique désignant une variable, un sous-programme, une procédure, un fichier ...

Image : copie d'informations sur un autre support.
exemple : image de carte = représentation d'une carte perforée

Interface :
plan de jonction conventionnel entre deux ordinateurs ou organes limités à des matériels permettant les échanges d'information suivant des règles déterminées.

IØCS : abréviation de "Input / Output Control System".
Ensemble de routines pouvant être incorporées dans un programme quelconque afin d'y gérer tous les problèmes généraux d'entrée - sortie

(lecture, écriture, contrôles ...)

Label : groupe de caractères servant à identifier et décrire un volume ou un fichier

Liaison (Link) :

séquence d'instruction réalisant le liaison avec une autre partie du programme.

Linkage editor (=éditeur de liens) :

voir éditeur de liens.

Linkage loader (=chargeur-éditeur de liens) :

voir chargeur - éditeur de liens.

Lister : imprimer tout ou partie des informations traitées article par article

Load and go (=chargement et exécution) :

technique d'exploitation permettant de convertir directement un langage spécifique en langage machine et d'en assurer l'exécution immédiate.

Load module (=module de chargement) :

résultat du traitement de l'éditeur de liens.
Le programme peut alors être chargé en mémoire principale pour être exécuté.

Loader : programme de chargement (permettant le chargement automatique d'autres programmes).

Marque de bande (=Tape mark) :

caractère conventionnel représenté par le symbole TM. Signale la fin de la portion de bande considérée.

Module : ce qui est traité comme en tout par un assembleur, un compilateur ou un éditeur de liens, ou le résultat de ce traitement.
exemple : load module, object module.

Module de chargement (=load module) voir load module.

Module résultant (=object module) voir object module.

Mot : plus petit groupe de positions de mémoire adressable dans son ensemble.

Multiplet (=byte) ensemble de bits consécutifs traité comme un tout.

MVT : abréviation de Multiprogramming With a Variable number of Tasks Multiprogrammation avec un nombre variable de taches. Exécution simultanée de plusieurs travaux avec affectation de la mémoire à chaque travail en fonction de la place demandée (région).

Multitraitement (multiprocessing) :

technique dans laquelle plusieurs organes d'exécution travaillent simultanément sur un ou plusieurs programmes en ayant accès à des mémoires ou à des organes communs.

Object module (=module résultant) résultat d'un seul assemblage ou d'une seule compilation, qui constitue une entrée de l'éditeur de liens.

Octet (=8 - bit byte) multiplet de 8 bits (=byte).

Off-line : non connecté à l'unité de traitement ou à l'ordinateur central

On - line : connecté à l'unité de traitement ou à l'ordinateur central.

Opérande : toute quantité entrant dans une opération mathématique.

Opérateur : symbole définissant un traitement mathématique à effectuer sur les quantités qu'il associe.

Piste (=track) :

partie linéaire d'un support mobile d'information accessible par une station de lecture ou d'écriture.

Région : - la mémoire centrale d'un ordinateur gérée par un système d'exploitation avec un nombre variable de taches (MVT) est découpée d'une façon dynamique en plusieurs régions dont la taille a été choisie par l'utilisateur.

Routine : suite ordonnée d'instruction qui peut avoir un emploi général ou répété.

Software (=logiciel) :

ensemble des programmes, procédés et règles destinés à commander le fonctionnement d'un ordinateur.

Spool (abréviation de "Simultaneous Peripheral Operations On-line") :

en multiprogrammation, méthode de travail permettant d'exécuter 1 ou plusieurs programmes de service, ou annexes, en même temps qu'un ou plusieurs programmes de traitement.

Table des matières du Volume (=VTOC Volume Table of Contents) fichier associé à un volume de mémoire à accès sélectif et contenant les labels des différents fichiers de ce volume.

Tache (Task) : élément de travail formant un tout pour des programmes de contrôle et considéré comme tel par l'unité de traitement. (élément de base de la multiprogrammation).

Tape-mark (=marque de bande) voir marque de bande.

Unité logique : symbole utilisé par le programmeur et désignant un type d'unité physique.

Volume : désignation générale d'une mémoire externe identifiée d'une façon unique. Désigne toute la partie d'une mémoire accessible par un seul mécanisme de lecture - écriture.

VTOC : abréviation de "Volume Table of Contents" voir table des matières du Volume.

ANNEXE - 2 -

UTILISATION PRATIQUE DU 370/168-168

Nous donnons ici quelques renseignements pratiques qui facifiteront l'emploi de l'ensemble matériel-logiciel, actuellement disponible au CIRCE. Rapellons que cet ensemble est pour l'instant constitué d'un ordinateur IBM 370/168-168 exploité sous les systèmes ØS - MVT version 21-6 et ASP version 3. Nous indiquerons tout d'abord quelques restrictions et contraintes diverses propres à ce type d'installation et nous donnerons ensuite quelques "recettes" permettant de rechercher et de corriger quelques unes des erreurs les plus courantes à partir des messages fournis par le système.
Enfin, nous signalerons quelques ressources supplémentaires mises à la disposition de l'utilisateur pour la mise au point des programmes.

1 - RESTRICTIONS ET CONTRAINTES.

1.1. Programmation FØRTRAN sur ordinateur IBM.

- Codage du Format :

Dans le codage du format de lecture ou écriture d'an enregistrement, on ne doit pas utiliser de termes supérieures à 255

Ainsi par exemple :

FØRMAT(I8,F7.2,45ØI5) n'est pas autorisé

Il faut écrire : FØRMAT(I8,F7.2,2 (225I5))

- Spécification Iw de l'instruction FORMAT :

Pour les ordinateurs IBM, une variable entière a une longueur standard de 4 octets et doit donc être comprise entre $-2^{31} = -2147483648$ et $2^{31} - 1 = 2147483647$ (voir Annexe 1).

En conséquence pour un format de lecture, seuls sont autorisés les spécifications I1 à I9 et éventuellement I10 dans la mesure où l'entier lu ne dépasse pas les valeurs limites indiquées ci-dessus.

- Constante réelle :

Une constante réelle, ne faisant l'objet d'aucun ordre de définition explicite en double ou quadruple précision, est considérée comme REAL*4 quel que soit le nombre de chiffres après la virgule.

exemple :

A = 21.98753829457168 est en fait équivalent à

A = 21.987538 et n'occupe que 4 octets (1 mot) en mémoire.

- Double ou quadruple précision :

Il faut obligatoirement spécifier la lettre D ou Q lorsqu'on affecte une valeur numérique à une variable déclarée en double ou quadruple précision.

exemple : Si l'on écrit :

```
REAL*8  A

A = 5.843216734
```

La valeur donnée à A est tronquée et ne sera prise qu'en simple précision c'est-à-dire dans cet exemple A = 5.843216

L'écriture correcte est :

```
REAL*8  A

A = 5.843216734D+00
```

- Limites d'un tableau :

Un tableau peut avoir jusqu'à 7 dimensions sur l'ordinateur IBM 370.

1.2. Instructions de contrôle

- Valeurs limites des paramètres LRECL et BLKSIZE de la carte DD.

valeur théorique minimale admise 18 octets
valeur maximale admise 32760 octets sur bande
13030 octets sur disque sans spécification spéciale.

On peut cependant augmenter ces valeurs maximales :

- sur disque par l'emploi de l'option "TRACK ØVERFLOW" (T) du paramètre RECFM exemple : RECFM = VBT

- sur bande en utilisant l'option "SPANNED" (S). du paramètre RECFM exemple : RECFM = VBS

- Sortie sur imprimante.

Actuellement au CIRCE, les fichiers du type SYSØUT = A (c'est-à-dire pour sortir sur imprimante) ont un paramètre BLKSIZE limité à 2020 octets.

Ainsi, par exemple pour imprimer des lignes en 132 positions avec le minimum d'opérations E/S on pourra effectuer cette impression par groupe de 14 lignes en codant le LRECL et le BLKSIZE de la façon suivante :

RECFM = VBA
LRECL = 132 + 4 + 1 = 137 (4 octets pour le compteur d'enregistrement)
1 octet pour le caractère de contrôle de saut ASA)
BLKSIZE = (137*14) + 4 (4 octets pour le compteur de bloc)

soit DCB = (RECFM = VBA, LRECL = 137, BLKSIZE = 1922)

- Nombre maximal de cartes de contrôle :

Ce nombre est limité théoriquement à 255 par étape.

- Nombre maximal d'unités logiques utilisables pour les fichiers :

99 en FØRTRAN IBM
20 en WATFIV

- Nombre maximal de lignes en sortie sur imprimante :

option par défaut : 2000 lignes
avec carte MAIN pas de limite théorique.

- Nombre maximal de cartes en sortie :

option par défaut : 200 cartes
avec carte MAIN pas de limite théorique

- Entrée / sortie de fichiers cartes :

Ces fichiers ne peuvent pas être bloqués. Ils sont donc obligatoirement du type :

DCB = (RECFM = F, LRECL = 80)

Rappelons qu'il est inutile de coder le DCB pour les fichiers
du type SYSOUT = B (sortie sur perforatrice)
ou du type DD* (entrée de fichiers cartes).

- Concaténation :

- la limite maximale de fichiers que l'on peut concaténer est de :

255 pour les fichiers séquentiels
16 pour les fichiers partitionnés

- le compilateur FØRTRAN H n'admet pas la concaténation de fichiers résidant sur des supports différents.

- Chaîne de programmes dans un même JOB :

* Lorsqu'un JOB contient différentes étapes avec exécution de plusieurs programmes au cours de ces étapes, il est nécessaire de préciser le fichier LKED.SYSLMØD par adjonction d'une carte DD supplémentaire à partir du deuxième programme.

Ceci permet au système de distinguer les fichiers constitués par les différents "LOAD MODULE" (programmes compilés et linkés), même si ceux-ci se trouvent sur le même support disque.

- L'instruction DD à ajouter est de la forme :

```
//LKED.SYSLMØDbDDbDSN=&&ET1(RUN),DISP=(NEW,PASS),
//   UNIT=SYSDA,SPACE=(CYL,(3,1,1),,CØNTIG)
```

ET1 étant le nom de l'étape (DDNAME).
Cette carte doit être placée avant les cartes DD relatives à l'étape GØ.

Exemple : éxécution en série de 3 programmes CØBØL dans le même JØB.

```
//carte JØB
//PR1bEXECbCBACLG
//CØB.SYSINbDDb*
    | programme CØBØL n° 1 sur cartes
/*
//GØ.SYSOO4bDDbUNIT= ...
  .........
```

```
/*
//PR2bEXECbCBACLG,REGIØN=90K
//CØB.SYSINbDDb*
    | programme CØBØL n° 2 sur cartes
/*
//LKED.SYSLMØDbDDbDSN=&&PR2(RUN),DISP=(NEW,PASS),
//   UNIT=SYSDA,SPACE=(CYL,(3,1,1),,CØNTIG)
//GØ.SYSOO3bDDbUNIT= ...
     ..............
/*
//PR3bEXECbCBACLG
//CØB.SYSINbDDb*
    | programme CØBØL n° 3 sur cartes
/*
//LKED.SYSLMØDbDDbDSN=&&PR3(RUN),DISP=(NEW,PASS),
//   UNIT=SYSDA,SPACE=(CYL,(3,1,1),,CØNTIG)
//GØ.SOO7bDDbUNIT= ...
     ..............
/*
//
```

* Si dans la chaîne de programmes du même JOB, un programme CØBØL suit un programme FØRTRAN, il faut également préciser le paramètre :
DISP = (NEW,PASS) du fichier SYSLIN de la procédure CBACLG.

On ajoute donc dans ce cas une carte de la forme :

```
//CØB.SYSLINbDDbDISP=(NEW,PASS)
```

Pour respecter les règles de remplacement des cartes DD des procédures cataloguées, il faut placer cette carte entre la carte EXEC et la carte CØB.SYSIN DD.

exemple :

```
// carte JØB
//LEC 1bEXECbFTXCLG
//FØRT.SYSINbDDb*
     programme FØRTRAN sur cartes
```

```
/*
//GØ.SYSINbDDb*
    | données sur cartes (fichier 5)
/*
//GØ.FT08FØ01bDDbUNIT=3330-1, ...
     ..........
/*
//EXT2bEXECbCBACLG
//CØB.SYSLINbDDbDISP=(NEW,PASS)
//CØB.SYSINbDDb*
    | programme CØBØL sur cartes
/*
//LKED.SYSLMØDbDDbDSN=&&EXT2(RUN),DISP=(NEW,PASS),
//  UNIT=SYSDA,SPACE=(CYL,(3,1,1),,CØNTIG)
//GØ.SYSØØ5bDDbUNIT= ...
//GØ.SYSØØ6bDDb ........
     ..............
/*
//
```

- Création et relecture d'un membre d'un fichier partitionné avec un programme FØRTRAN.

La carte DD décrivant le fichier partitionné à relire doit obligatoirement contenir le paramètre

LABEL = (,,,IN)

Si on oublie de spécifier LABEL = (,,,IN) on obtient dans le fichier SYSMSG un message de la forme :

```
IEC225I  04,NØMJ,GØ, FT10FØ01,254,PP3330,  WRR1358.RØCHEMAR.FICH1
```

Code erreur nom du JØB

Nom d'étape

Nom de la carte DD

Nom du volume sur lequel réside le fichier.

Nom du fichier

Notons que la création de membre de fichier partitionné avec un programme FORTRAN exige la spécification du paramètre LABEL = (,,,ØUT) dans la carte DD décrivant le fichier.

2 - ERREURS LES PLUS COURANTES -

- Nous donnons ici quelques erreurs fréquentes avec des explications possibles permettant d'orienter la recherche de l'erreur à partir des messages fournis par le système.

2.1. COMPLETION CODE.

Ce type de diagnostic apparaît dans le fichier SYSMSG avant la liste du programme et le plus souvent à l'étape GØ.
Un complétion code entraîne systématiquement l'abandon de l'exécution en cours par le système.

001 En COBOL, il s'agit en général d'une erreur d' Entrée / Sortie qui provient d'une différence de longueur entre ce qu'il y a de décrit dans le programme (FILE SECTION) et ce qui existe réellement sur le fichier (DCB).

ØCx - En gestion de fichier cela provient en général d'un mauvais JCL.
ØC1: quelques exemples entraînant ce message :

- 1 carte DD manquante
- programme COBOL avec instructions DISPLAY et omission de la carte SYSØUT.
- écriture sur un fichier non ouvert.

ØC4:
- taille du bloc et de l'enregistrement incorrecte pour des enregistrements de longueur variable
- mouvement de données sur un fichier non ouvert
- lecture sur un fichier non ouvert ou sans carte DD.

- En programmation FORTRAN, il s'agit en général d'une instruction entraînant un mauvais positionnement en mémoire centrale (chevauchement des données). ØC1, ØC4, ØC5 :
Cette erreur d'adressage peut avoir des causes diverses - par exemple :

- une variable est non définie ou non initialisée
- indice nul dans une variable indicée
- inversion d'indices d'une variable à plusieurs indices
- tableau dépassant la place réservée dans l'ordre DIMENSION
- arguments de nature différente dans le sous programme et le programme principal :

  exemple : simple - double précision

entier - réel, INTEGER*2 - INTEGER*4 ...

- 2 boucles DØ avec même numéro et même indice (ØC5)

- ect ...

222 Nombre de lignes ou de cartes demandé insuffisant.
Les options par défaut au CIRCE sont de 2000 lignes et 200 cartes.
Pour augmenter ces valeurs, utiliser une carte MAIN.

322 Temps spécifié insuffisant pour l'exécution du JØB.
L'option par défaut au CIRCE est de 30 secondes.
Pour augmenter cette valeur, utiliser le paramètre TIME de la carte JØB.

422 Nombre de cartes système trop élevé.
Ce nombre doit être inférieur à 255.

80A ou 804 Il n'y a pas assez de place en mémoire pour l'étape.
Augmenter le nombre de K dans le paramètre REGION de la carte EXEC.
Option par défaut pour l'étape GØ : 64 K

- Ne pas oublier l'encombrement des buffers d'entrée / sortie pour les fichiers sur support magnétique avec un paramètre BLKSIZE élevé.

B37 Cette erreur apparait à la création d'un fichier.
Ce fichier a besoin d'utiliser l'allocation secondaire pour s'étendre et il n'y a plus de place sur le volume,
solution : recopier le fichier sur un disque qui possède un espace libre suffisant.

E37 Toute la place allouée pour le fichier est épuisée, y compris l'allocation secondaire.
Solution : recopier le fichier en demandant plus de place pour le nouveau fichier (utilitaire IEB COPY1).

213 ou 313 Erreur qui se produit si l'on essaie de lire un fichier sur disque non créé ou mal créé.

F13 Erreur à l'exécution d'ouverture d'un fichier.

En cas de création de fichierspartitionnés sur disque, cette erreur se produit lorsque le fichier a été créé par un JØB antérieur mais le membre n'a pas été créé.

Dans ce cas mettre DISP = (MØD, KEEP)

2.2. Autres messages d'erreurs.

Il s'agit des messages de diagnostic consécutifs à des erreurs se produisant :

- soit au niveau de la compilation IFExxxI en FØRTRAN H
- soit au niveau du link-édit IEWxxxx en FØRTRAN H
- soit au niveau de l'exécution IHØxxxI en FØRTRAN H

Pour la plupart des erreurs apparaîssant au niveau de l'exécution, le système effectue une correction standard avec apparition du message suivant :

STANDARD FIXUP TAKEN, EXECUTIØN CØNTINUING

Cette correction standard est repétée éventuellement un certain nombre de fois. Quand le nombre limite de corrections prévu est atteint le système abandonne l'exécution du programme et il y a impression systèmatique dans le fichier SYSMSG au niveau de l'étape GØ du message suivant :

CØMPLETIØN CØDE - SYSTEM = 000 USER = 0240

Enfin, un récapitulatif des erreurs est donné sur le fichier FT06F001

2.2.1. Messages au niveau de la compilation IFExxxI.

Ces messages correspondent à des erreurs à la compilation. Il s'agit de fautes de syntaxe du langage FØRTRAN H.

Ces messages indiquent le numéro de liste (ISNxxxx) ou le numéro d'instruction (LABELxxxx) dans laquelle l'erreur s'est produite.

Ces messages sont suffisamment explicites pour que l'erreur soit immédiatement détectée.

2.2.2. Messages au niveau du link-édit IEWxxxx.

Les erreurs au niveau de l'éditeur de liens sont assez rares.
Nous ne retiendrons que la plus fréquente :

IEW0132 Ce message indique l'utilisation d'une fonction ou d'un sous programme non défini.

2.2.3. Messages au niveau de l'exécution IHØxxxI.

Ces messages apparaissent dans le courant des résultats. Nous n'examinerons ici que les plus fréquents :

IHØ207I OVERFLOW : il y a dépassement de capacité mémoire c'est-à-dire utilisation d'un nombre flottant $> 10^{75}$

On vérifiera en particulier que toutes les variables utilisées dans des opérations sont bien initialisées.

La correction standard permet de continuer l'exécution en prenant le plus grand nombre possible pouvant être représenté.

IHØ208I UNDERFLOW : il y a sous dépassement de capacité c'est-à-dire utilisation d'un nombre flottant $< 10^{-78}$
Même processus que pour l'overflow.

La correction standard permet de continuer l'exécution en remplaçant le nombre en défaut par zéro.

IHØ209I DIVIDE CHECK : division par zéro.

La correction standard permet de continuer l'exécution sans faire la division.

IHØ210I Ce message peut avoir différentes causes :

- une erreur dans les ordre EQUIVALENCE
- une division par zéro pour un entier.

IHØ212I On cherche à lire ou à écrire avec un format qui dépasse la longueur du buffer :

- en cas d'écriture ou de lecture sur disque, vérifier le DEFINE FILE (accès direct)
- en cas de sortie sur imprimante, vérifier que le format ne dépasse pas 132 caractères
- en cas des travaux sur bande, vérifier qu'il n'y a pas de contradiction entre le DCB et l'ordre d'Entrée / Sortie.

IHØ213I On cherche à lire ou à écrire sans format une liste plus grande que l'enregistrement logique.
On fera les mêmes vérifications que pour IHØ212I.

IHØ215I Mauvaise lecture : par exemple :

- on demande de lire du décimal en format I
- ou bien on est positionné sur un caractère invalide.

On vérifiera qu'il ne manque pas de cartes données ou qu'elles ne sont pas mélangées.

On vérifiera qu'il n'y a pas eu une mauvaise reproduction de cartes (caractère décalé par exemple).

Sur bande, on vérifiera que l'on est bien positionné sur le bon fichier.

IHØ217I Rencontre d'une fin de fichier alors que le programme prévoit de lire des données.

- On vérifiera qu'il ne manque pas de cartes données.
- Sur bande, on vérifiera que l'on ne se borne pas à lire le label.

IHØ219I Exécution terminée, due à un trop grand nombre d'erreurs (207I ou 208I ou 209I).

Cette liste n'est pas limitative ; les vérifications indiquées ne sont données que pour une première investigation mais ne sont pas absolues. Il ne faut pas oublier qu'un même diagnostic peut être provoqué par N erreurs.

3 - OUTILS DE MISE AU POINT DES PROGRAMMES FØRTRAN -

3.1. Choix du compilateur.

Pour la mise au point des programmes FØRTRAN, nous conseillons d'utiliser de préférence les compilateurs FØRTRANG1 et WATFIV.

Ces compilateurs fournissent en effet des diagnostics beaucoup plus précis que le compilateur H étendu qui, étant plus performant, sera ensuite utilisé avec les programmes opérationnels.

On remarquera toutefois que le compilateur WATFIV fait l'objet d ' un grand nombre de restrictions et d'incompatibilités par rapport au FØRTRAN IV G ou H dont il faudra tenir compte (voir 3ème partie chapitre 15.3.1.).

Le compilateur FØRTRANG1 est particulièrement adapté à la mise au point des programmes puisqu'il permet l'utilisation d'instructions spéciales pour localiser les erreurs éventuelles (voir paragr. 3.3.).

3.2. Exploitation des messages d'erreurs.

Lorsqu'il y a interruption à l'exécution le code d'abandon du programme par le système est donné dans le fichier FT06F001.

On aura par exemple un message de la forme :

```
IHØ900I EXECUTIØN TERMINATING DUE TØ ERRØR CØUNT FØR ERRØR NUMBER 240

IHØ240I STAE - ABEND CØDE IS :  SYSTEM ØØCS  , USER 0000 .........
                                code d'erreur

                                .... PSW IS FFA5000D  A20A59A6
                                                      adresse d'interruption
                                                      du programme

TRACEBACK ROUTINE CALLED FRØM ISN REG. 14  REG. 15   REG. 0   REG. 1
          MAIN
                                  000123B8 010A5810 FF000018 000BDFD8

ENTRY PØINT = 01CA5810 <--- adresse de
                            chargement du programme
```

Dans cet exemple, le programme s'est terminé anormalement avec un code d'abandon 0C5 dans le programme principal (MAIN).

La différence entre l'adresse d'interruption et l'adresse de chargement du module en erreur donne l'adresse en absolue du point d'arrêt.

Soit dans cet exemple :

Point de chargement 0A5810
Point d'interruption 0A59A6

0A59A6 - 0A5810 = 196 (hexadécimal)

196 est l'adresse de l'instruction qui suit le point où il y a eu erreur.

Pour trouver l'instruction FØRTRAN qui a provoqué l'erreur il faut obtenir une liste du code assembleur généré à la compilation.

Cette liste s'obtient en spécifiant l'option "liste" à la compilation par l'utilisation du paramètre PARM.FØRT de la carte EXEC.

On aura par exemple : //E1bEXECbFTG1CLG,PARM.FØRT=LIST

Dans le cas d'abandon du programme avec des codes d'erreurs portant sur la gestion des fichiers tels que

COMPLETION CODE SYSTEM = B37,D37,E37,113,213,313,413,513, 613,813,913,A13,C13

un message supplémentaire apparaît dans le fichier SYSMSG donnant le nom de la carte DD ou le DSNAME du fichier en erreur

soit par exemple :

```
SYSABEO1  JØB  TØTØ        ABENDED CØDE = D37
              nom du JØB   DDNAME = FTO8FOO1
```

Ou

```
SYSABEO1 JØB   TØTØ         ABENDED  CØDE = 213
               nom du JØB   DSNAME = WRR1358.RØCHEMAR.FICH1
```

Ces renseignements supplémentaires sont très utiles pour la recherche de l'erreur considérée.

3.3. Ressources supplémentaires pour la recherche des erreurs. (instruction DEBUG).

Il existe un module de mise au point des programmes FØRTRAN utilisable seulement en FØRTRAN G.

Ce module est placé soit dans le programme principal soit dans un sous programme FORTRAN. Son effet est alors strictement limité au programme ou sous-programme dans lequel il est placé. Il est défini par l'instruction :

DEBUG UNIT (n), paramètres

n désigne une unité de sortie du programme. Si l'option UNIT est ommise, les sorties de mise au point se font sur l'unité standard (n = 6 imprimante).

Chacun des paramètres définit un sous-programme particulier de mise au point ; Ce sont :

TRACE
SUBTRACE
INIT
SUBCHK

Ces sous-programmes sont appelés par le programme à analyser au moment de son exécution :

- soit automatiquement pour les sous-programmes SUBTRACE, INIT et SUBCHK chaque fois que l'exécution d'une instruction du programme à analyser justifie la sortie d'un message ;

- soit sur commande pour le sous-programme TRACE et l'ensemble des instructions exécutables du module.
  L'instruction AT nnnn placée dans le module (où nnnn est l'étiquette d'une instruction du programme à analyser) définit alors le point d'appel dans le programme à analyser.

Les paramètres d'un instruction DEBUG peuvent être indiqués dans n'importe quel ordre, et doivent être séparés par des virgules :

exemple :

```
DEBUG TRACE, SUBCHK
AT  130
TRACE  ØN
```

demande de la "trace" à partir de l'instruction 130 et analyse des indices (SUBCHK) en séquence.

3.3.1. Sous-programme SUBTRACE

Utilisation :

```
programme FØRTRAN

DEBUG  SUBTRACE
END
```

Ce sous-programme provoque au moment de l'exécution du programme FØRTRAN l'impression du message :

```
SUBTRACE   xxxxxx
```

où xxxxxx est le nom du programme FORTRAN,
et du message :

```
SUBTRACE  * RETURN *
```

au moment où le contrôle est renvoyé à une autre partie du programme (exécution de l'instruction FØRTRAN RETURN).
Ce sous-programme permet donc d'afficher le nom du programme ou sous-programme chaque fois que le contrôle lui est transmis.

3.3.2. Sous-programme SUBCHK

Utilisation :

```
programme FØRTRAN

DEBUG  SUBCHK  (X1,X2, .....Xn)
END
```

Ce sous-programme examine les valeurs prises par les indices des tableaux X1, X2, ... Xn au moment de l'exécution.

En cas de dépassement de la dimension réservée pour le tableau X, il provoque l'impression du message :

SUBCHK X (nnn)

où X est le tableau contrôlé et nnn la valeur de l'indice linéaire correspondant aux valeurs prises par les indices du tableau en dépassement. De la même façon si un indice du tableau X est trouvé égal à zéro on aura sortie du message :

SUBCHK X (0)

Si aucun nom de tableau n'est précisé dans l'option, l'examen a lieu pour tous les tableaux du programme.

3.3.3. Sous-programme INIT.

Utilisation

```
programme FØRTRAN

DEBUG  INIT  (X1, X2, ... Xn)
END
```

Ce sous-programme provoque à chaque transfert d'une valeur dans X1, X2, ... Xn l'impression d'un message du type :

X = nn. nnnn

X1, X2, ... Xn pouvant être des variables simples ou des tableaux.

Si aucun nom de variable n'est précisé dans l'instruction DEBUG INIT l'examen et la sortie des valeurs sont effectués pour toutes les variables du programme.

3.3.4. Sous-programme TRACE.

L'appel à ce sous-programme se fait sur commande à l'aide de l'instruction

AT nnn

Où nnn désigne l'étiquette d'une instruction du programme à analyser.

L'exécution des instructions qui suivent AT nnn a lieu immédiatement avant celle de l'instruction référencée nnn.

Lorsque l'option TRACE a été spécifiée dans l'instruction DEBUG, l'exécution du sous-programme correspondant est alors commandée par l'instruction :

TRACE ØN

A l'exécution de chaque instruction du programme à analyser référencée ppp correspondra la sortie du message :

TRACE ppp

ceci à partir de l'instruction référencée nnn de l'instruction AT (et y compris nnn).

Cette analyse peut être arrêtée au moyen d'une instruction.

TRACE ØFF

précédée d'une instruction AT précisant l'étiquette à partir de laquelle cette interruption doit prendre effet.

exemple :

```
.....................
1  READ(5,2)X                    produit :
2  FORMAT(5X,F5.2)                          TRACE  1
   J = 5                                    TRACE  3
3  J = J + 2                                TRACE  3
   IF(J - 13)3,4,4                          TRACE  3
4  STØP                                     TRACE  3
   DEBUG  TRACE
   AT  1
   TRACE  ØN
   AT  4
   TRACE  ØFF
   END
```

Remarque : Un module de mise au point peut contenir toute les instructions d'un programme FØRTRAN ordinaire à l'exception des instructions suivantes :

IMPLICIT

BLØCK DATA

ENTRY

ou une définition arithmétique de fonction.

ANNEXE - 3 -

RECAPITULATIF DESCRIPTIF DES PRINCIPAUX PARAMETRES DE LA CARTE DD.

PARAMETRES	BANDES MAGNETIQUES	DISQUES 3330
DSNAME ou DSN = nom du fichier	quelconque $\leqslant$ 17 caractères, le 1er étant une lettre	WRR1358.RØCHEMAR.non du fichier ... $\leqslant$ 8 caract. $\leqslant$ 8 caract. $\leqslant$ 8 caract. $\leqslant$ 8 caract. (total $\leqslant$ 44 caract) &nom : fichiers temporaires
UNIT = type d'unité	BDE7 bandes 7 canaux 2400-4 bandes 9 canaux. d = 800 bpi BD16 bandes 9 canaux. d = 1600 bpi BD20 bandes 9 canaux. d = 6250 bpi	3330 disques 3330 modèle 1 3330-1 disques 3330 modèle 11
VØLUME ou VØL = { SER = numéro (,RETAIN,SER= numéro). REF = *.nom }	RETAIN employé pour éviter le démontage de la bande en fin d'étape SER = numéro de la bande $\leqslant$ 6 chiffres REF = employé pour faire référence à un volume défini dans une étape précédente (nom)	RETAIN employé pour un volume en Setup. Le volume n'est pas démonté en fin d'étape. SER = nom du volume (6 caractères) : - résident permanent : RES301, RES302, RES303, RES304 (module 11) - privé provisoire : PP3330 (module 11) - Setup : SET001, SET002, ..., SET012
LABEL = (n [,NL / ,SL / ,BLP])	n : rang du fichier sur la bande NL : "NØ LABEL" SL : label standard IBM (option par défaut) BLP : le label n'est pas traité	généralement inutilisé

RECAPITULATIF DESCRIPTIF DES PRINCIPAUX PARAMETRES DE LA CARTE DD (SUITE)

<table>
<tr><th>PARAMETRES</th><th>BANDES MAGNETIQUES</th><th>DISQUES 3330</th></tr>
<tr><td>DISP=([val1],[val2],[val3])
val1 : état du fichier en début d'étape.
val2 : état du fichier en fin d'étape.
val3 : état du fichier en cas de fin anormale d 'étape</td><td colspan="2">← identifique pour bande et disque →
Val 1 : SHR : fichier sur disque consulté par plusieurs utilisateurs (équivalent à ØLD)
NEW : fichier créé dans l'étape
ØLD : fichier existant consulté dans l'étape non partagé
MØD : fichier existant modifié dans l'étape.
Val 2 : DELETE : fichier non conservé en fin d'étape
KEEP : fichier conservé en fin d'étape
PASS : fichier repris dans une étape ultérieure du travail.
Val 3 : DELETE, KEEP } même signification que val 2 en cas de fin anormale d'étape.</td></tr>
<tr><td rowspan="2">DCB = ([LRECL = L]
[,RECFM = val]
[,BLKSIZE = B]
[,DEN = d)]</td><td colspan="2">DEN = 1 pour 556 bpi
2 pour 800 bpi
3 pour 1600 bpi
4 pour 6250 bpi</td></tr>
<tr><td colspan="2">RECFM={[U] / [V] / [F]} [B] [A] [S]
U enregistrement de longueur indéfinie
V enregistrement de longueur variable
F enregistrement de longueur fixe
B enregistrements bloqués
A l'enregistrement contient un caractère de saut ASA
S enregistrements étendus (ou spannés)
LRECL = L : longueur d'un enregistrement logique (en octets) pour RECFM = F ou FB
longueur maximale d'un enregistrement logique + 4 pour RECFM = V ou VB ou VBS
BLKSIZE = B : longueur réelle ou maximale d'un enregistrement physique.</td></tr>
<tr><td>SPACE=({nn / TRK / CYL},{(q1,q2,q3) / (q1,q2)}
[,RLSE])</td><td>Inutilisé</td><td>n : allocation par blocs de n octets
TRK : allocation par piste : 13030 octets / piste
CYL : allocation par cylindre : 19 pistes / cylindre
q1 : allocation primaire
q2 : allocation secondaire (15 fois q2)
q3 : réservation pour le directory pour un fichier partitionné ou pour l'index pour un fichier séquentiel indexé (q3 enregistrement de 256 octets).
RLSE : permet de restituer la place inutilisée (à la création uniquement).</td></tr>
</table>