



N° D'ORDRE

T H E S E

présentée à

LA FACULTE DES SCIENCES DE L'UNIVERSITE DE GRENOBLE

Pour Obtenir

LE GRADE DE DOCTEUR DE TROISIEME CYCLE
"Mathématiques Appliquées"

par

Georges BEAUME

PROJET SOCRATE

(2-2) Gestion des Mémoires

Thèse soutenue le 22 Décembre 1970 devant la commission d'examen

Monsieur J. KUNTZMANN	Président
Monsieur L. BOLLIET	Examineurs
Monsieur N. GASTINEL	
Monsieur J.C. BOUSSARD	
Monsieur F. SALLE	

L I S T E D E S P R O F E S S E U R S

Doyen honoraire : Monsieur M. MORET
Doyen : Monsieur E. BONNIER

PROFESSEURS TITULAIRES

MM.	NEEL Louis	Physique Expérimentale
	KRAVTCHENKO Julien	Mécanique Rationnelle
	CHABAUTY Claude	Calcul différentiel et intégral
	BENOIT Jean	Radioélectricité
	CHENE Marcel	Chimie Papetière
	FELICI Noël	Electrostatique
	KUNTZMANN Jean	Mathématiques Appliquées
	BARBIER Reynold	Géologie Appliquée
	SANTON Lucien	Mécanique des Fluides
	OZENDA Paul	Botanique
	FALLOT Maurice	Physique Industrielle
	KOSZUL Jean-Louis	Mathématiques
	GALVANI Octave	Mathématiques
	MOUSSA André	Chimie Nucléaire
	TRAYNARD Philippe	Chimie Générale
	SOUTIF Michel	Physique Générale
	CRAYA Antoine	Hydrodynamique
	REULOS René	Théorie des Champs
	BESSON Jean	Chimie Minérale
	AYANT Yves	Physique Approfondie
	GALLISSOT François	Mathématiques
Melle.	LUTZ Elisabeth	Mathématiques
MM.	BLAMBERT Maurice	Mathématiques
	BOUCHEZ Robert	Physique Nucléaire
	LLIBOUTRY Louis	Géophysique
	MICHEL Robert	Minéralogie et pétrographie
	BONNIER Etienne	Electrochimie et Electrometallurgie
	DESSAUX Georges	Physiologie animale
	PILLET Emile	Physique Industrielle-Electrotechnique
	YOCCOZ Jean	Physique Nucléaire théorique
	DEBELMAS Jacques	Géologie Générale
	GERBER Robert	Mathématiques
	PAUTHENET René	Electrotechnique
	MALGRANGE Bernard	Mathématiques Pures
	VAUQUOIS Bernard	Calcul Electronique
	BARJON Robert	Physique Nucléaire

MM.	BARBIER Jean-Claude	Physique
	SILBER Robert	Mécanique des Fluides
	BUYLE-BODIN Maurice	Electronique
	DREYFUS Bernard	Thermodynamique
	KLEIN Joseph	Mathématiques
	VAILLANT François	Zoologie et Hydrobiologie
	ARNAUD Paul	Chimie
	SENGEL Philippe	Zoologie
	BARNOUD Fernand	Biosynthèse de la Cellulose
	BRISSONNEAU Pierre	Physique
	GAGNAIRE Didier	Chimie Physique
Mme.	KOFLER Lucie	Botanique
MM.	DEGRANGE Charles	Zoologie
	PEBAY-PEROULA Jean-Claude	Physique
	RASSAT André	Chimie Systématique
	DUCROS Pierre	Cristallographie Physique
	DODU Jacques	Mécanique Appliquée I. U. T.
	ANGLES D'AURIAC Paul	Mécanique des Fluides
	LACAZE Albert	Thermodynamique
	GASTINEL Noël	Analyse numérique
	GIRAUD Pierre	Géologie
	PERRET René	Servo-mécanisme
	PAYAN Jean-Jacques	Mathématiques Pures

PROFESSEURS SANS CHAIRE

MM.	GIDON Paul	Géologie
Mme.	BARBIER Marie-Jeanne	Electrochimie
Mme.	SOUTIF Jeanne	Physique
	COHEN Joseph	Electrotechnique
	DEPASSEL R.	Mécanique des Fluides
	GLENAT René	Chimie
	BARRA Jean	Mathématiques Appliquées
	COUMES André	Electronique
	PERRIAUX Jacques	Géologie et Minéralogie
	ROBERT André	Chimie Papetière
	BIARRELL Jean	Mécanique Physique
	BONNET Georges	Electronique
	CAUQUIS Georges	Chimie Générale
	BONNETAIN Lucien	Chimie Minérale
	DEPOMIER Pierre	Physique Nucléaire-Génie Atomique
	HACQUES Gérard	Calcul numérique
	POLOUJADOFF Michel	Electrotechnique
Mme.	KAHANE Josette	Physique
Mme.	BONNIER Jane	Chimie
MM.	VALENTIN Jacques	Physique
	REBECQ Jacques	Biologie
	DEPORTES Charles	Chimie
	SARROT-REYNAULD Jean	Géologie
	BERTRANDIAS Jean-Paul	Mathématiques Appliquées
	AUBERT Guy	Physique

PROFESSEURS ASSOCIES

MM. RODRIGUES Alexandre
MORITA Susumu
RADHAKRISHNA
Mathématiques Pures
Physique Nucléaire
Thermodynamique

MAITRES DE CONFERENCES

MM. LANCIA Roland
Mme. BOUCHE Liane
MM. KAHANE André
DOLIQUE Jean Michel
BRIERE Georges
DESRE Georges
LAJZEHOWICZ Joseph
LAURENT Pierre
Mme. BERTRANDIAS Françoise
MM. LONGEQUEUE Jean-Pierre
SOHM Jean-Claude
ZADWORNÝ François
DURAND Francis
CARLIER Georges
PFISTER Jean-Claude
CHIBON Pierre
IDELMAN Simon
BLOCH Daniel
MARTIN-BOUYER Michel
SIBILLE Robert
BRUGEL Lucien
BOUVARD Maurice
RICHARD Lucien
PELMONT Jean
BOUSSARD Jean-Claude
MOREAU René
ARMAND Yves
BOLLIET Louis
KUHNS Gérard
PEFFEN René
GERMAIN Jean-Pierre
JOLY Jean-René
Melle. PIERY Yvette
BERNARD Alain
MOHSEN Tahsin
CONTE René
LE JUNTER Noël
LE ROY Philippe
ROMIER Guy
VIALON Pierre
BENZAKEN Claude
MAYNARD Roger
Physique Atomique
Mathématiques
Physique Générale
Electronique
Physique
Chimie
Physique
Mathématiques Appliquées
Mathématiques Pures
Physique
Electrochimie
Electronique
Chimie Physique
Biologie végétale
Physique
Biologie animale
Physiologie animale
Electrotechnique I. P.
Chimie (C. S. U. Chambéry)
Construction mécanique (I. U. T.)
Energétique I. U. T.
Hydrologie
Botanique
Physiologie animale
Mathématiques Appliquées (I. P. G.)
Hydraulique I. P. G.
Chimie I. U. T.
Informatique I. U. T.
Energétique I. U. T.
Chimie I. U. T.
Mécanique
Mathématiques Pures
Biologie animale
Mathématiques Pures
Biologie (C. S. U. Chambéry)
Mesures Physiques I. U. T.
Génie Electrique Electronique I. U. T.
Génie Mécanique I. U. T.
Techniques Statistiques quantitatives
I. U. T.
Géologie
Mathématiques Appliquées
Physique

MM.	DUSSAUD René	Mathématiques (C. S. U. Chambéry)
	BELORIZKY Elie	Physique (C. S. U. Chambéry)
Mme.	LAJZEROWICZ Jeannine	Physique (C. S. U. Chambéry)
M.	JULLIEN Pierre	Mathématiques Pures
Mme.	RINAUDO Marguerite	Chimie
MM.	BLIMAN Samuel	E. I. E.
	BEGUIN Claude	Chimie Organique
	NEGRE Robert	I. U. T.

MAITRES DE CONFERENCES ASSOCIES

MM.	YAMADA Osamu	Physique du Solide
	NAGAO Makoto	Mathématiques Appliquées
	MAREZIO Massimo	Physique du Solide
	CHEECKE John	Thermodynamique
	BOUDOURIS Georges	Radioélectricité
	ROZMARIN Georges	Chimie Papetière

A l'heure des remerciements, c'est d'abord à l'équipe que je pense ...

à Jean-Raymond ABRIAL qui a donné à cette équipe un esprit et une méthode. Il a su nous communiquer son enthousiasme et son dynamisme pour le projet SOCRATE. C'est en toute amitié que je lui exprime ma reconnaissance pour les deux années passionnantes passées à travailler avec lui.

à Robert MORIN et Georges VIGLIANO avec qui je passe cette thèse. Robert est un travailleur acharné et infatigable, d'une efficacité redoutable, il n'est pas toujours facile de suivre son rythme ! Georges est l'homme des situations difficiles : aucun "bug" PL/1, aucun algorithme vicieux ne lui résiste ... Il est toujours prêt à rendre service avec une patience exemplaire.

à Gérard HENNERON, précis et méticuleux, il a un grand souci du travail bien fait et il est certainement le plus ordonné de nous tous.

à Jacques BAS arrivé après nous, il a su s'intégrer à notre groupe.

Pour décrire l'ambiance de cette équipe, je dirai simplement que nous avons passé des nuits entières sur l'ordinateur sans nous "manger le nez" ! Pour qui connaît la tension survoltée qui s'établit pendant de telles nuits, il est clair que de solides liens d'amitié existent entre nous ...

Je tiens maintenant à remercier les membres de notre Jury :

Monsieur Jean KUNTZMANN qui a bien voulu en accepter la présidence,

Monsieur Louis BOLLIET qui a toujours manifesté son intérêt et sa confiance dans le projet SOCRATE.

Messieurs Noël GASTINEL et Jean-Claude BOUSSARD qui ont accepté d'y participer.

Monsieur François SALLÉ qui vient spécialement sur notre invitation.

./.

Enfin, il me faut évoquer l'environnement et les circonstances de ce travail.

Pour ses soins affectueux, son soutien moral (et ses petits plats mijotés !), je remercie de tout coeur Tante NANO. Elle a supporté ma vie d'étudiant et toutes mes frasques de jeunesse avec beaucoup de patience ...

Je n'oublie pas François PECCOUD qui m'a orienté vers la gestion des fichiers, et Gilles DAGAND avec qui j'ai fait mes premières armes.

Il me reste à remercier Madame TREVISAN. Elle a réalisé l'impression de ce travail avec une minutie et un soin tout particuliers. J'espère qu'elle ne m'en veut pas pour les travaux amenés parfois un peu tardivement ... !

A MES PARENTS,

L'exposé qui suit est un complément aux spécifications générales du projet SOCRATE.

Les définitions données dans les spécifications générales sont supposées connues.

Les références à cet ouvrage sont notées [1].

Les autres références du projet SOCRATE sont notées :

- [2.1.] Langage de Requêtes*
- [2.2.] Gestion des mémoires*
- [2.3.] Interprétation et Compilation.*

L'ensemble de ces quatre ouvrages constitue une thèse de groupe.

oOoOoOoOoOoOoOoOo

S O M M A I R E

INTRODUCTION -----	1
CHAPITRE 1 - ESPACE VIRTUEL GENERALISE -----	3
1.1. Les structures internes -----	4
1.1.1. - Description d'une structure interne	4
1.1.2. - Espace virtuel associé -----	5
1.1.3. - Conclusion -----	6
1.2. Les Données -----	6
1.3. Les outils utilisateurs -----	7
1.3.1. Les Macros -----	8
1.3.2. Les fonctions utilisateurs -----	9
1.4. Les fichiers de travail -----	10
1.4.1. Fichiers séquentiels -----	10
1.4.1.1. Fichiers "Imprimante-Virtuelle"	10
1.4.1.2. Fichiers "Type-Cobol" -----	10
1.4.1.3. Fichiers "Création-Batch" ---	11
1.4.1.4. Virtualisation des fichiers séquentiels -----	11
1.4.2. Fichiers directs -----	12
1.5. Conclusions -----	13
CHAPITRE 2 - PASSAGE DE L'ESPACE VIRTUEL A L'ESPACE REEL --	15
2.1. Rappels -----	16
2.2. Définition des fonctions de base -----	18
2.3. Etude des fonctions(II) et (III) -----	20
2.3.1. Définitions des registres -----	20
2.4. Etude de la fonction (I) -----	22
2.4.1. Format d'une page -----	22
2.4.2. Gestion des demandes de pages en mémoi- re centrale -----	23
2.4.3. Fonction (I) -----	24
2.5. Optimisation des fonctions de base -----	26
2.5.1. Recherche directe de la sous-page ----	26
2.5.2. Micro-programmation -----	27
2.5.3. Mesures -----	27
CHAPITRE 3 - ORGANISATION DU SYSTEME -----	28

I N T R O D U C T I O N

Le concept de mémoire virtuelle est maintenant classique et trouve son application dans un grand nombre de systèmes d'exploitation en temps partagé.

Il permet d'utiliser des hiérarchies de mémoires (par exemple : mémoire centrale, tambour, disques, ...), les différents niveaux de mémoire restant transparents du point de vue de l'utilisateur qui croit travailler uniquement sur le premier niveau. Les transferts d'informations entre les différents niveaux de mémoires sont réalisés de façon automatique soit par des dispositifs hardware, soit par des micro-programmes, soit par des programmes.

Les mémoires virtuelles sont utilisées pour les programmes et possèdent une image de même taille dans la mémoire de plus bas niveau.

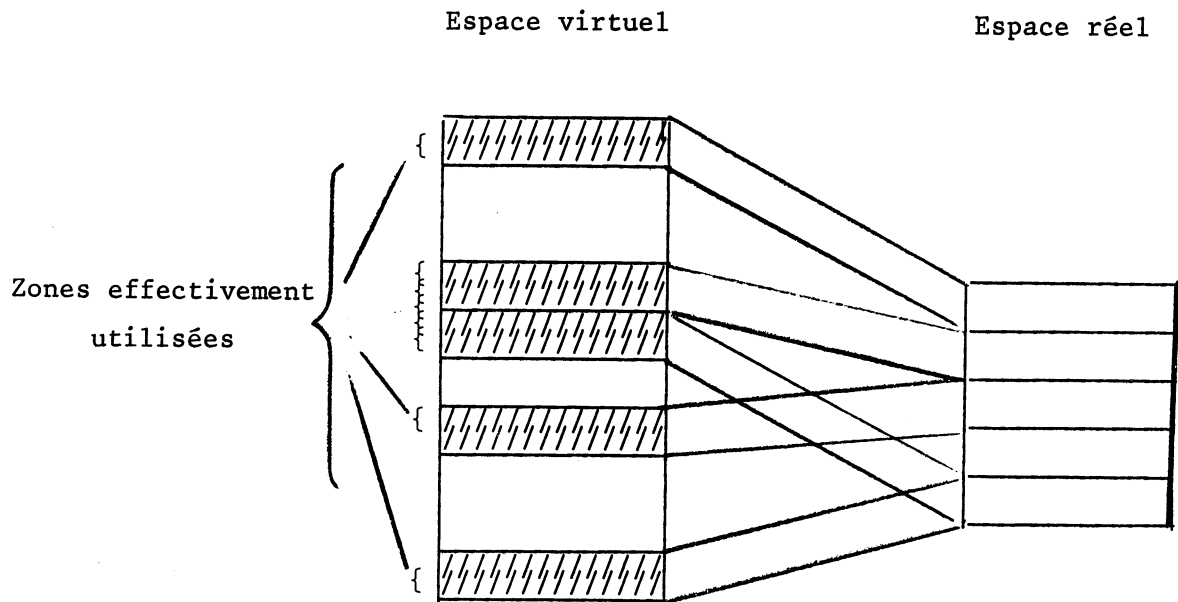
Dans le projet SOCRATE, la notion d'espace virtuel est utilisée au niveau des données ([1] § 3.3.).

Ceci permet, comme dans les systèmes classiques, une indépendance complète entre les programmes et l'organisation physique des données.

L'unité atomique choisie pour cet espace est le mot de 32 bits. Sa taille est gigantesque : 2^{32} mots.

Cette taille permet de faire une allocation statique de l'espace virtuel : les informations sont retrouvées par simple calcul d'adresse.

Cet espace virtuel est projeté dans un espace réel dont la taille dépend du volume des informations qui seront créées.



Le premier chapitre sera consacré à l'étude de l'organisation des informations dans l'espace virtuel.

Le second chapitre décrit les mécanismes de transfert des informations et la transformation d'une adresse virtuelle en adresse réelle.

Le troisième chapitre traite de l'organisation générale du système dans un contexte de "multi-tasking".

CHAPITRE 1

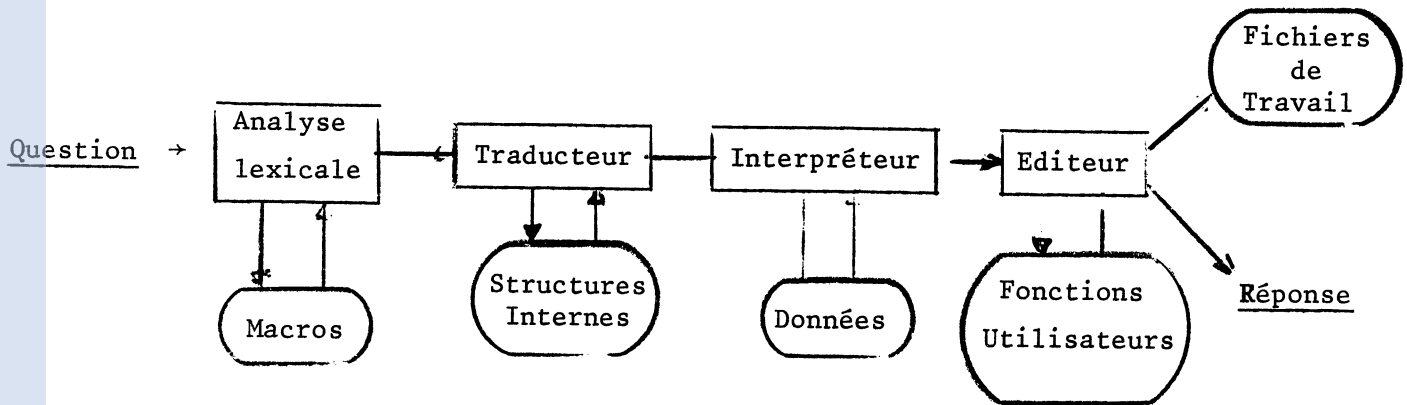
ESPACE VIRTUEL GENERALISE

L'espace virtuel généralisé est l'espace permettant d'enregistrer toutes les informations relatives à une ou plusieurs applications.

Ces informations peuvent se diviser en quatre classes :

- les structures internes,
- les données (les fichiers),
- les outils de travail (macros et fonctions utilisateurs),
- les fichiers de travail.

L'utilisation de ces informations peut se représenter de la façon suivante :



tel-00282344, version 1 - 27 May 2008

Ce schéma n'a pas pour but de montrer le fonctionnement réel du système de banque de données. Il permet simplement de situer les différents niveaux où sont généralement utilisées les informations que nous allons décrire.

1.1. LES STRUCTURES INTERNES

La structure interne est le résultat de la traduction du langage de définition ([1] § 4.1.).

C'est elle qui détermine toutes les actions sur le fichier en permettant de retrouver l'adresse d'une donnée, et en indiquant les renseignements nécessaires au traitement de cette donnée.

1.1.1. Description d'une structure interne

En utilisant le langage de définition, une structure interne peut se décrire de la façon suivante ([1] § 4.3.).

STRUCTURE-INTERNE

entité CARACTERISTIQUE

début

NOM mot

C de 0 à 2^{32}

Q de 0 à 2^{32}

M de 0 à 2^{32}

ED de 0 à 2^{32}

TYPE (MOT TEXTE LIVAL VALNUM SI IDEM REFERENCE ...)

FRERE référence CARACTERISTIQUE

REQUETE référence LISTE-REQUETES

si TYPE = 'LIVAL' alors FILS référence LISTE-VALEURS

sinon si TYPE = 'VALNUM' alors FILS référence VALEUR-NUMERIQUE

sinon si TYPE = 'REFERENCE' alors CITATION texte

./.

```
        sinon si TYPE = 'SI' alors CONDITION texte
        sinon FILS référence CARACTERISTIQUE
    fin
entité LISTE-VALEURS
    début
        NOM mot
        entité VALEURS mot
    fin
entité VALEUR-NUMERIQUE
    début
        NOM mot
        BORNE-INF de  $-2^{31}$  à  $+2^{31}$ 
        BORNE-SUP de  $-2^{31}$  à  $+2^{31}$ 
        PRECISION de 0 à  $2^{32}$ 
        UNITE mot
    fin
entité LISTE-REQUETES
    début
        NOM mot
        entité REQUETE texte
    fin
fin
```

1.1.2. Espace virtuel associé

Toutes les structures internes se présentant sous la même forme, il semble naturel de les intégrer dans un même espace virtuel.

Cette solution présente deux avantages :

- a) - Possibilité d'adapter la taille de sous-page à la forme particulière d'une structure interne. Rappelons que pour optimiser les allocations d'espaces réels en mémoire secondaire la taille des sous-pages doit être déterminée en fonction de l'application ([1] § 3.3.4.2.).

./.

- b) - Possibilité de choisir un support physique particulier, Les structures internes peuvent ainsi être mises sur un support à accès très rapide (par exemple, un tambour).

1.1.3. Conclusion

Le temps de traitement d'une question est pratiquement ramené au temps d'interprétation sur le fichier, la traduction et la compilation étant effectuées très rapidement.

Dans un système multi-tâches cette solution peut permettre d'avoir une seule activation du compilateur : ceci évite l'emploi et la gestion de mécanismes complexes qui deviendraient nécessaires si le compilateur était partagé entre plusieurs questions.

1.2. LES DONNEES

L'espace virtuel associé aux données à été largement décrit dans les spécifications générales du projet SOCRATE ([1] § 3.3.2.).

L'espace réel alloué est décrit par une carte de la mémoire secondaire :

CARTE-MEMOIRE

début

entité ALLOCATION

début

NOM mot

TAILLE de 0 à XX

entité ADRESSES

début

NUM-PAGE de 0 à XX

NOM-VOLUM (VOL-1 ... VOL-N)

NUM-CYLINDRE de 0 à 199

NUM-PISTE de 0 à 19

fin

fin

fin

NOM : nom de l'application

TAILLE : nombre de pages allouées

NUM-PAGE : indique le plus grand numéro de page adressable à partir de l'adresse disque (NUM-CYLINDRE, NUM-PISTE) sur le volume spécifié (NOM-VOLUME).

1.3. LES OUTILS UTILISATEURS

Ils sont créés pour faciliter les communications de l'utilisateur avec ses données :

- au niveau des questions, par l'emploi de macros,
- au niveau de la présentation, par l'emploi de fonctions spéciales.

1.3.1. Les Macros ([1] § 2.4.)

Une macro est définie par son nom et par son expansion. L'ensemble des macros peut donc se décrire de la façon suivante :

```
entité MACRO
  début
    NOM mot
    EXPANSION texte
  fin
```

L'appel d'une macro est détecté par l'analyseur lexical qui doit tester pour chaque nom (non reconnu comme mot du langage par l'analyseur) s'il s'agit d'un nom de macro.

Dans le cas d'une réponse affirmative, le contrôle est donné au macro-générateur qui récupère les paramètres éventuels et remplace l'appel de la macro par son expansion.

La reconnaissance d'un nom de macro est donc assez complexe, mais permet une grande souplesse d'emploi : les macros restent transparentes pour l'utilisateur.

Lorsque l'analyseur rencontre une chaîne ALFA, il doit se poser la question suivante :

existe-t-il une MACRO AYANT NOM = 'ALFA' ; ?

Il existe donc une analogie très étroite entre la gestion des macros et la gestion des données. Il est naturel de considérer que les macros forment un fichier, dont la description a été donnée ci-dessus, avec recherche rapide sur le nom ([1] § 3.3.6.2.).

L'espace virtuel alloué pour les macros peut être pris :

- soit dans l'espace virtuel des données,
- soit dans l'espace virtuel des structures,
- soit dans un espace virtuel spécialisé.

Là critère de temps que nous avons appliqué pour la structure interne reste vrai au niveau des macros : les deux dernières solutions sont préférables si l'on dispose d'un support à accès très rapide.

Les macros correspondant à différentes applications se trouvent toutes dans un même espace virtuel ; il se pose alors un problème d'homonymie. Cet inconvénient peut être évité :

- soit en découpant l'espace virtuel par application,
- soit en faisant le hashcode sur :

NOM APPLICATION|| NOM MACRO

(concaténation).

1.3.2. Les fonctions utilisateurs

Elles se présentent sous la forme d'un nom et d'un module chargeable :

entité FONCTION

début

NOM mot

PROGRAMME texte

fin

Le problème est le même que pour les macros : il faut pouvoir retrouver rapidement un programme par son nom. Le programme est amené en mémoire centrale par l'interpréteur. Celui-ci passe alors le contrôle à un chargeur qui résoud les références externes et donne à son tour le contrôle au programme.

Ces fonctions sont utilisées au même niveau que les données : l'espace virtuel alloué sera donc pris dans l'espace des données.

1.4. LES FICHIERS DE TRAVAIL

Ils sont générés comme réponse éventuelle à une question et peuvent être séquentiels ou directs ([1] § 5.5.2.).

1.4.1. Fichiers séquentiels

Ils sont caractérisés par l'usage auquel ils sont destinés.

1.4.1.1. Fichiers "Imprimante-Virtuelle"

Ces fichiers sont créés pour réaliser l'édition en différé par exemple sur une imprimante rapide.

La taille des enregistrements logiques est fonction du dispositif utilisé.

1.4.1.2. Fichiers "Type-Cobol"

Il s'agit ici de résultats qui seront exploités ultérieurement en batch par des programmes de gestion classique, écrits en langage de programmation.

La taille de l'enregistrement dépend du programme qui doit utiliser ce fichier. Il faut que l'utilisateur puisse indiquer le format "Cobol" de son fichier.

(Remarque: Le terme "Cobol" est utilisé ici dans un sens large, et fait appel à la notion classique de fichier).

1.4.1.3. Fichiers "Création-Batch"

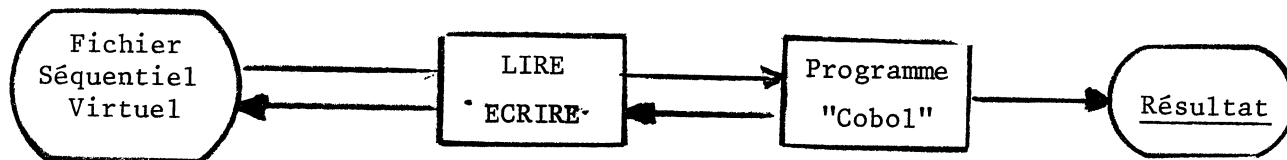
Ces fichiers se présentent comme des fichiers stream, leur format interne est indifférent. Ils sont utilisés par le programme de création pour générer des données sur une certaine structure ([1] § 6.3.5.1.).

1.4.1.4. Virtualisation des fichiers séquentiels

Deux solutions sont envisageables :

- i) - les fichiers sont séquentiels dans l'espace virtuel, mais pas dans l'espace réel.
- ii) - les fichiers sont séquentiels dans l'espace réel (et à fortiori dans l'espace virtuel).

Dans le premier cas, il est nécessaire (si le fichier sert en entrée d'un programme "Cobol") de réutiliser les fonctions LIRE et ECRIRE qui permettent de passer d'une adresse virtuelle à une adresse réelle :



En fait, le choix dépend de la politique suivie lors de l'allocation de la mémoire secondaire pour chaque application.

Si l'espace est alloué à priori pour une application, les zones de travail devant être prises dans cet espace, il faut choisir la première solution. (Ce type de gestion est utilisé par le système "CP" sur IBM 360/67). Chaque utilisateur est responsable de son propre espace.

Une autre technique consiste à allouer temporairement une partie banalisée de la mémoire secondaire pour les fichiers de travail. Cette méthode impose une gestion très dynamique de cette mémoire.

Dans ce dernier cas, les fichiers peuvent être créés de façon séquentielle dans l'espace réel.

Enfin, s'il est possible d'utiliser des bandes magnétiques, la gestion de la mémoire secondaire à accès direct en sera grandement simplifiée !

1.4.2. Fichiers directs

Ce sont des fichiers "PAR" ([1] § 5.2.2.2.4.) et des tableaux. Dans le cas d'un fichier "PAR", l'information élémentaire (appelée BOITE) est un compteur, dans le cas des tableaux il s'agit d'une valeur donnée par l'interpréteur.

Ces fichiers peuvent se décrire à l'aide du langage de définition :

FICHER-PAR

début

PARAMETRES

début

NOM-APPLICATION mot

⋮

entité CARACT-CROISEES mot

fin

entité BOITE de 0 à XXX

fin

PARAMETRES : Ensemble des renseignements nécessaires au programme d'édition.

./.

Le nombre maximum de réalisations de l'entité BOITE est simple à calculer.

$$\text{CARD}(P\{(x_1, \dots, x_n) \mid x_i \in X_i\})$$

n = nombre de caractéristiques croisées

X_i = une caractéristique du croisement

x_i = une valeur de X_i .

L'espace réel alloué est pris dans la partie banalisée de la mémoire secondaire à accès direct.

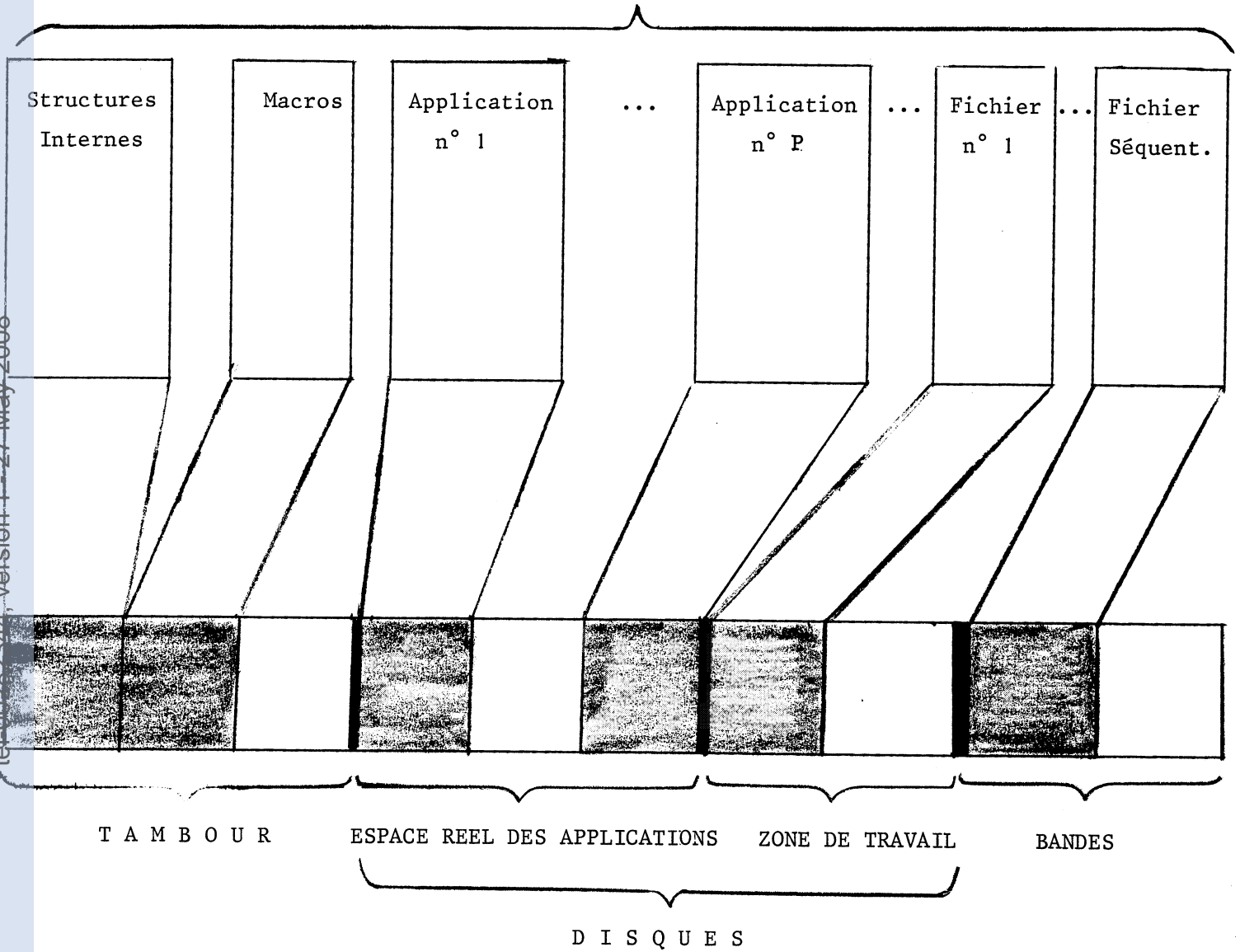
1.5. CONCLUSIONS

L'analyse que nous venons de faire des différentes classes d'informations montre que l'on a intérêt à spécialiser les espaces virtuels.

Les programmes d'accès à ces informations sont les mêmes quelque soit le niveau de l'information (niveau structurel, niveau de donnée ou autre).

L'ensemble étant ainsi très homogène, le coût (en "taille programme") de la gestion de toutes ces informations est minimum.

E S P A C E S V I R T U E L S



CARTE DE L'ALLOCATION DES ESPACES REELS DANS L'ENSEMBLE

DES SUPPORTS DISPONIBLES

C H A P I T R E 2

PASSAGE DE L'ESPACE VIRTUEL A L'ESPACE REEL

Ce chapitre présente les fonctions de base permettant d'accéder à une information.

De la même façon que dans les mécanismes classiques de pagination, il est possible d'envisager différents niveaux de réalisations :

- hardware,
- microprogrammation,
- programmes.

L'étude portera principalement sur la réalisation au niveau programme, et essaiera par analogie avec les systèmes classiques (Atlas, Multics, IBM 360/67, ...), d'aborder les autres niveaux de réalisation.

2.1. RAPPELS

L'espace virtuel est projeté dans l'espace réel par blocs de 2^k mots consécutifs appelés sous-pages. Les sous-pages sont regroupées en pages dans l'espace réel ([1] § 3.3.4.2.).

La taille d'une page dépend de contraintes physiques liées à l'installation. La taille d'une sous-page dépend d'une application.

Dans l'espace réel, il existe deux types de sous-pages :

- les sous-pages libres : elles sont doublement chaînées entre elles à l'intérieur d'une page. Chaque page possède en tête un pointeur sur cette liste.
- les sous-pages occupées : les sous-pages faisant collision sont chaînées entre elles en anneau.

Mot de contrôle d'une sous-page occupée :

IND'	SQ	S	CAP	CASP	NI
------	----	---	-----	------	----

IND' : indicatif
SQ : indique si la sous-page est un squatter
(S, CAP): chaînage des pages
(CASP) : chaînage des sous-pages
NI : nombre de lignes occupées dans la page.

Mot de contrôle d'une sous-page libre



CAM : chaînage amont

CAV : chaînage aval

Décomposition d'une adresse X



IND : indicatif

AP : n° de page réelle

ASP : n° de sous-page dans la page AP

LIGNE : déplacement dans la sous-page ASP

Le couple (AP, ASP) détermine une sous-page de l'espace réel.
Quatre cas peuvent se présenter :

- a - il s'agit d'une sous-page libre
- b - il s'agit d'une sous-page squatter (SQ = 1)
- c - c'est la bonne sous-page (SQ = 0, IND = IND')
- d - il y a collision (SQ = 0, IND ≠ IND').

Dans les deux cas a et b, la sous-page correspondant à X n'existe pas.

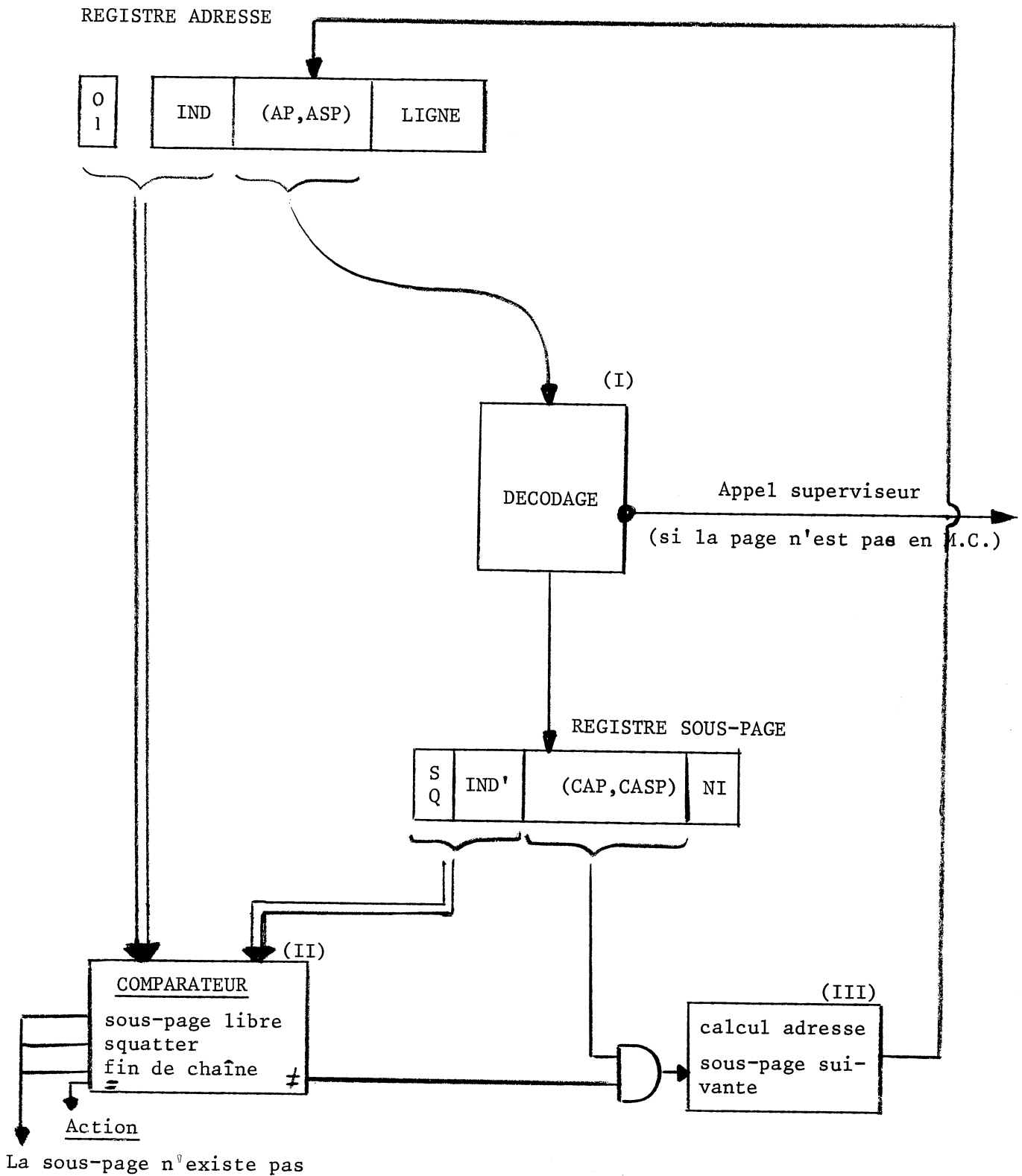
Dans le cas d, il faut parcourir l'anneau des sous-pages :

- si on trouve une sous-page telle que (IND,1) = (IND',SQ) c'est la bonne sous-page.
- dans le cas contraire, il n'y a pas de sous-page correspondant à X.

./.

2.2. DEFINITION DES FONCTIONS DE BASE

La recherche d'une sous-page peut se représenter de la façon suivante :



Ce schéma montre les trois fonctions de base qu'il faut réaliser pour déterminer une adresse réelle correspondant à une adresse virtuelle X.

(I) Recherche de la sous-page (AP, ASP) :

- i) Si la page AP n'est pas en mémoire centrale, il faut faire une entrée de page,
- ii) Chargement du registre sous-page avec le mot de contrôle de (AP, ASP).

(II) Comparaison

- i) Au premier passage :
 - . entre (IND,0) et (IND',SQ)
 - . entre SQ et 1
 - . test de sous-page vide.
- ii) Aux passages suivants :
 - . entre (IND,1) et (IND',SQ)
 - . test de fin de chaîne.

(III) Calcul de la sous-page suivante

$$(AP,ASP) \leftarrow (AP + CAP,CASP)$$

Ces trois fonctions doivent être réalisées en utilisant deux paramètres (dépendant d'une application donnée):

- le nombre de pages réelles
- la taille des sous-pages.

2.3. ETUDE DES FONCTIONS (II) ET (III)

Nous essayons ici d'expliciter les techniques employées pour réaliser de façon software ces deux fonctions.

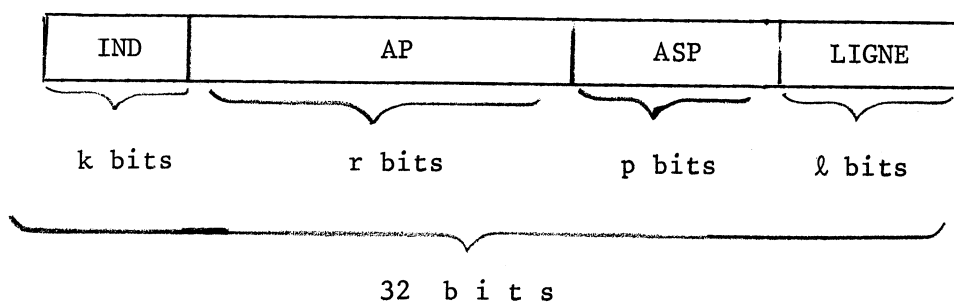
2.3.1. Définitions des registres

Le registre X et le registre sous-page sont réalisés à l'aide des registres généraux du 360.

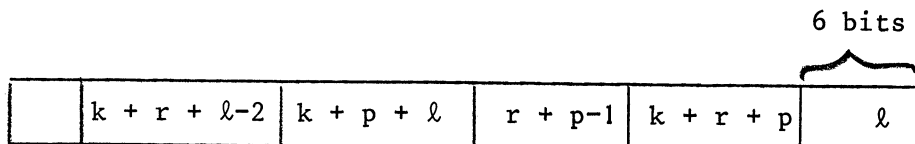
Registre X : R2 = (IND,B), R3 = (AP), R4 = (ASP)

Registre sous-page : R6 = (IND,SQ), R7 = (S,CAP), R5 = (CASP)

Format de X pour une application :



Registre de décalage (REGDEC)



(Rappel : Les instructions de décalage en 360 se présentent sous la forme :

SHIFT R, D(B)

Le second opérande de l'instruction n'est pas une adresse, mais indique (6 bits de droite) la longueur du décalage).

./.

Si le registre R6 contient le mot de contrôle à éclater, la séquence de décomposition d'un mot de contrôle de sous-page est la suivante :

L	R8,REGDEC	R8 ← registre de décalage
SRL	R6,0(R8)	
SRL	R8,12	R8 = R + P-1
LR	R5,R6	
SRDL	R6,0(R8)	$R6 = (IND, SQ)$
SRL	R8,6	R8 = K + P + L
SRA	R7,1(R8)	$R7 = (S, CAP)$
SRL	R8,6	R8 = K + R + L-2
SLL	R5,2(R8)	
SRL	R5,0(R8)	$R5 = (CASP * 2^2)$

La décomposition d'une adresse X se fait de façon similaire.

La fonction (III) est alors réalisée très facilement :

R3 ← R3 + R7
R4 ← R5 (AP,ASP) ← (AP + CAP,CASP)

soit en langage assembleur :

AR R3,R7
LR R4,R5

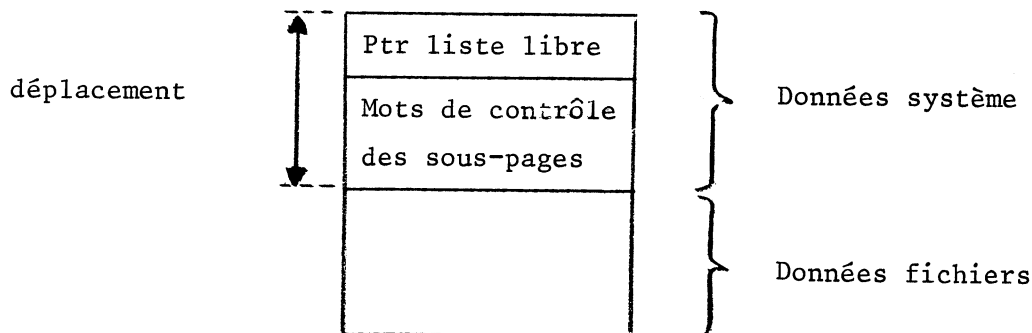
La fonction (II) est réalisée :

- en comparant R2 et R6((IND,1) = (IND',SQ))
- en comparant :
(R3 + R7, R5) avec (AP,ASP) initial
(Test de fin de chaîne)
- le test SQ = 1 se fait à l'aide d'un masque (instruction TM)
- le test de sous-page vide n'est effectué qu'au premier passage dans le cas où NI = 0.
(il faut regarder si la sous-page appartient à la liste des sous-pages vides).

2.4. ETUDE DE LA FONCTION (I)

2.4.1. Format d'une page

Chaque page regroupe en tête l'ensemble des mots de contrôle des sous-pages ainsi que le pointeur sur la liste libre.



Si nous supposons que la taille d'une page est fixée à 2K octets, et que la taille de sous-page est de 8 mots, il y a alors 64 sous-pages dans une page (déplacement = 65 mots).

Chaque page contient un pourcentage d'informations "pures" de $(512-65)/512 = 0,87$. Pour une taille de sous-page supérieure, ce rapport est évidemment amélioré.

Si le registre R a pour valeur l'adresse en mémoire centrale du cadre contenant la page AP, et si R' contient le numéro de sous-page ASP, on obtient :

- adresse du mot de contrôle de la sous-page ASP : $4(R',R)$
- adresse d'une ligne dans une sous-page :

$$R' \leftarrow R' * 2^{\ell} + \text{LIGNE}$$

l'adresse cherchée est DEPLACEMENT (R',R).

2.4.2. Gestion des demandes de pages en mémoire centrale

La mémoire centrale comporte un ensemble de cadres ([1]§ 7.2.1.2.) pouvant contenir des pages. Le contenu d'un cadre est indiqué dans une table de contrôle.

Plusieurs tâches travaillant sur ces cadres, des ressources permettent de synchroniser le déroulement des tâches et les mouvements de pages.

Cadre libre

Afin d'optimiser les demandes de pages, l'ensemble des cadres comporte un cadre libre. L'adresse de ce cadre est donnée par un pointeur dont l'utilisation par une tâche est contrôlée par une ressource.

Cadre de manoeuvre

Certaines opérations peuvent amener une tâche à travailler simultanément sur deux pages :

a) Mise à jour d'une chaîne de sous-pages occupée

Les deux sous-pages à chaîner peuvent se trouver sur deux pages différentes.

b) Recopie d'une sous-page.

Cette opération est effectuée soit pour déplacer un squatter, soit pour réorganiser une chaîne après récupération d'une sous-page.

Lorsqu'une opération de ce type se présente, la tâche active garde le contrôle et se met en attente pendant les opérations d'entrée - sortie de pages. Le cadre de manoeuvre est utilisé pour la 2ème page nécessaire au traitement.

Cette technique n'utilise pas les possibilités de traitement en parallèle, mais elle permet un fonctionnement plus simple du système.

./.

Dans le cas contraire, il faudrait introduire de nouveaux blocages :

- risque plus grand de dead-lock,
- perte de temps supplémentaire par passage dans le scheduler (choix de tâche, libération et réservation de ressources supplémentaires).

2.4.3. Fonction (I) ([1]§ 7.2.1.2.2.)

fonction (I)

si AP n'est pas en mémoire centrale

alors RESERVE RCL

si pas libre alors sauver registre X

charger adresse de retour

retour au SCHEDULER

Retour : restaurer registre X

I ← CL

CC(I) ← AP, TUC(I) ← 1

mettre CHOIX dans l'état éligible

RESERVE RC(I)

call ENTRE PAGE (AP,I,RC(I))

sinon

si TUC(I) = 0 RESERVE RNC0

TUC(I) ← TUC(I) +1

RESERVE RC(I)

si pas libre sauver registre X

sauver I

charger adresse de retour

retour au SCHEDULER

Retour : restaurer registre X

restaurer I

charger le registre sous-page

Pendant l'exécution de cette fonction, la tâche active peut perdre le contrôle :

- soit parce qu'il n'y a pas de cadre libre (réserve RCL)
- soit parce que la page AP a été demandée, mais ne se trouve pas encore en mémoire centrale (Réserve RC(I)).

CHOIX : RESERVE RNC0

I ← Index de cadre tel que TUC = 0

CL ← I, AP ← CC(I)

CC(I) ← 0

Si MAJ(I) = 1

alors MAJ(I) + 0

call SORT-PAGE (AP,I,RCL)

sinon LIBERE RCL

La ressource RCL est libérée soit par CHOIX, soit par l'interruption de fin de sortie de page.

La ressource RNC0 permet de connaître le nombre de cadres utilisés à un instant donné :

- dans lire ou écrire :

réserve RNC0 : RNC0 ← RNC0 + 1

libère RNC0 : si flag = 1 alors flag ← 0

mettre CHOIX éligible

- dans CHOIX :

réserve RNC0 : RNC0 RNC0 + 1

si RNC0 = (nb de cadres) alors flag ← 1

retour SCHEDULER

ENTREPAGE et SORTPAGE construisent à l'aide des paramètres qui leurs sont fournis le programme canal et lancent le SIO.

Si le SIO ne s'exécute pas (canal occupé ou unité occupée), la tâche canal créée est chaînée aux tâches de même type qui sont en attente.

L'interruption d'entrée-sortie :

- libère la ressource de la tâche correspondant au programme qui vient de s'exécuter (RC(I) ou RCL).
- lance un nouveau SIO pour les tâches canal en attente.

L'algorithme présenté ci-dessus possède les caractéristiques suivantes:

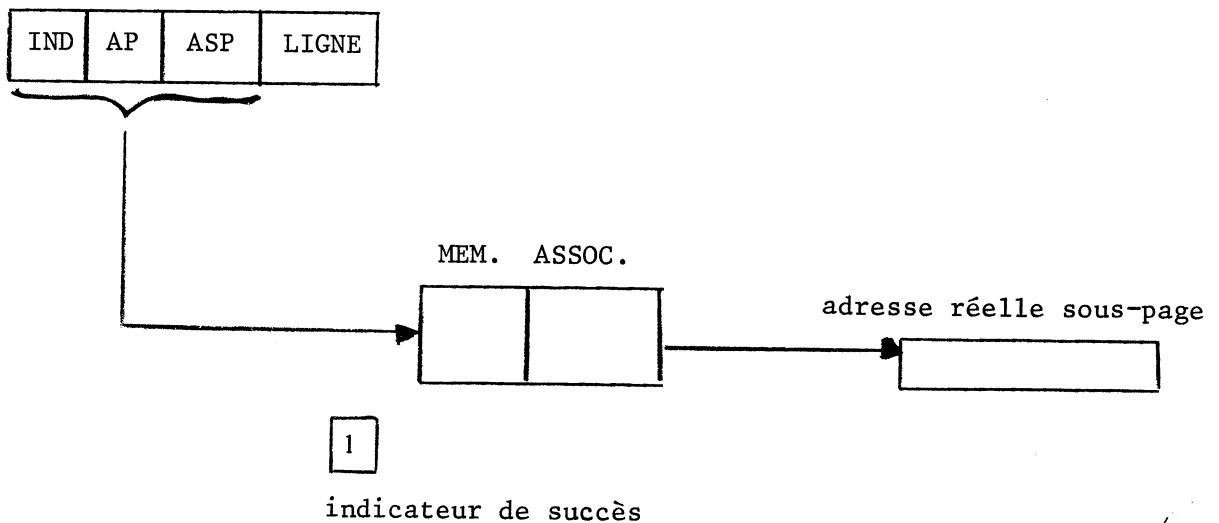
- toutes les tâches en attente d'une même page AP sont chaînées entre elles, et la page n'est demandée qu'une seule fois (grâce à la ressource RC(I)).
- Les programmes canaux sont initialisés aussitôt que possible : l'utilisation du canal est optimale.

2.5. OPTIMISATION DES FONCTIONS DE BASE

2.5.1. Recherche directe de la sous-page

La technique classique consiste à utiliser une mémoire associative :

ADRESSE VIRTUELLE X



Cette technique peut également être utilisée au niveau de la recherche d'une page en mémoire centrale.

2.5.2. Micro-programmation

Les fonctions (II) et (III) dépendant de paramètres (taille, sous-page, et nombre de pages), il semble qu'une solution micros-programmes soit plus adaptée que la réalisation de dispositifs câblés.

2.5.3. Mesures

Indépendamment de la réalisation technique, un tel système doit faire l'objet d'un ensemble de mesures et de calculs permettant d'évaluer ses performances.

En particulier, ce genre de système se dégrade à partir d'un certain taux d'occupation de l'espace réel. Ceci pose donc le problème d'allocation d'espace réel pour une application donnée en fonction des performances (temps de réponse) exigées.

Cette dualité classique "Espace-Temps" se traduit en fait par une politique de facturation :

$$\text{coût} = f(A * (\text{temps de réponse}), B * (\text{taille espace réel})).$$

Au niveau du fonctionnement global du système, il faut déterminer le nombre optimal de cadres en fonction du nombre de tâches.

Le choix du cadre à libérer est également un problème très important.

Ces mesures sont maintenant classiques, il est cependant nécessaire de les adapter au cas particulier d'une banque de données : par exemple, la probabilité de travailler dans une même page n'est pas la même dans une banque de données et dans un système de pagination sur des programmes.

La réalisation et l'interprétation de ces mesures sont en cours d'étude.

C H A P I T R E 3

ORGANISATION DU SYSTEME

Il semble qu'un système de banque de données fonctionnant seul sur un ordinateur soit peu rentable, et n'intéresse que quelques applications particulières.

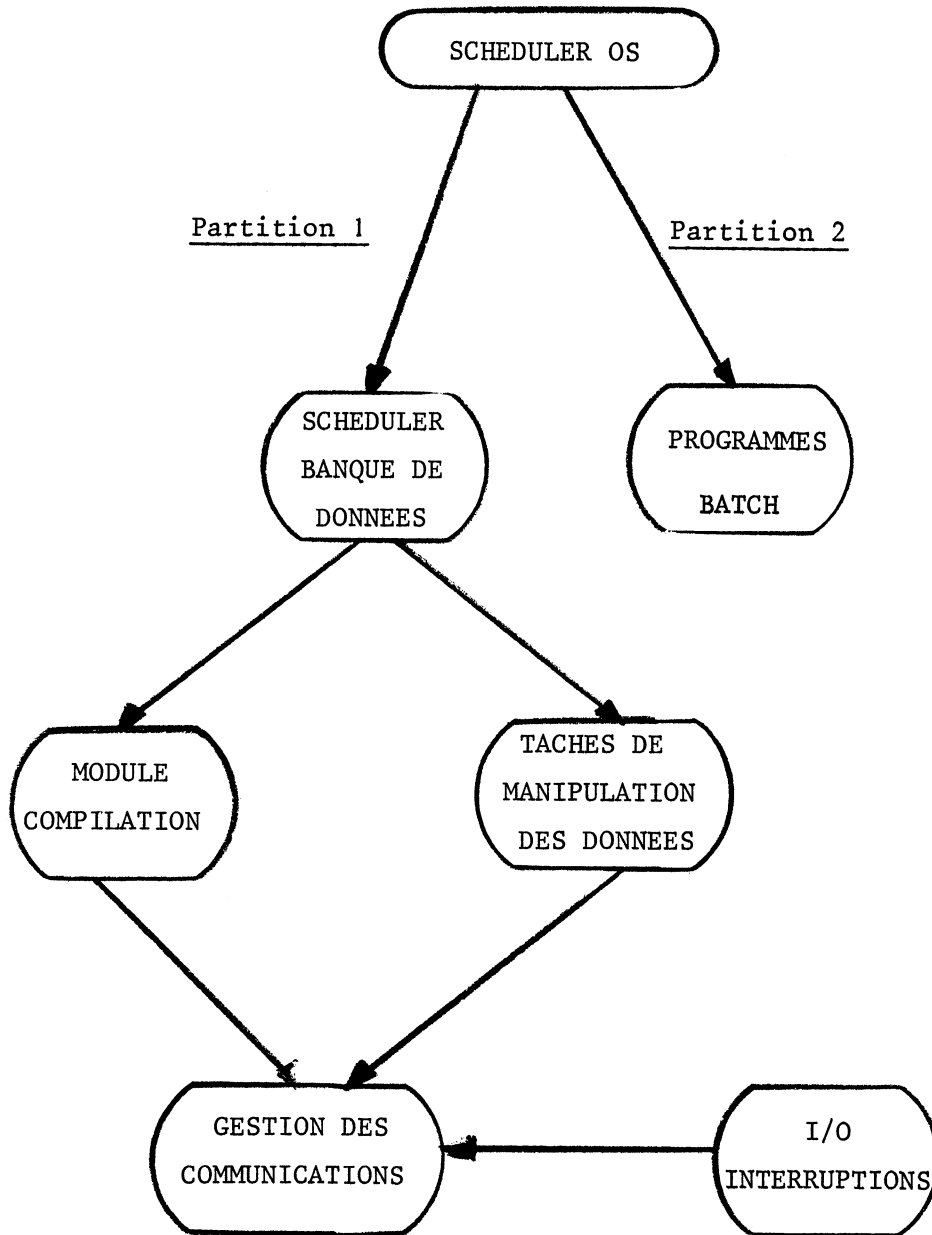
Ce type de fonctionnement peut s'envisager par exemple dans un centre hospitalier. Par contre, dans une entreprise, il serait en général un luxe trop coûteux malgré les services rendus.

Ces considérations nous ont amenés à repenser la banque de données comme un système prioritaire travaillant dans une partition d'un "opérating system" de type au moins "MFT".

La taille d'ordinateur visée correspond au modèle IBM 360/40 avec 128 K octets de mémoire centrale.

Ce mode de fonctionnement impose une réduction de la taille du système qui est réalisée en acceptant éventuellement une baisse de performances.

Lorsque la banque de données est active, le scheduler de la banque ne rend le contrôle au scheduler OS que lorsque toutes les tâches de la banque sont bloquées ou terminées.



Module compilation

C'est le module qui crée dynamiquement les algorithmes des tâches de manipulation des données. Ce module se compose de l'analyseur lexical et du macro-générateur, du traducteur et du compilateur ([4]).

./.

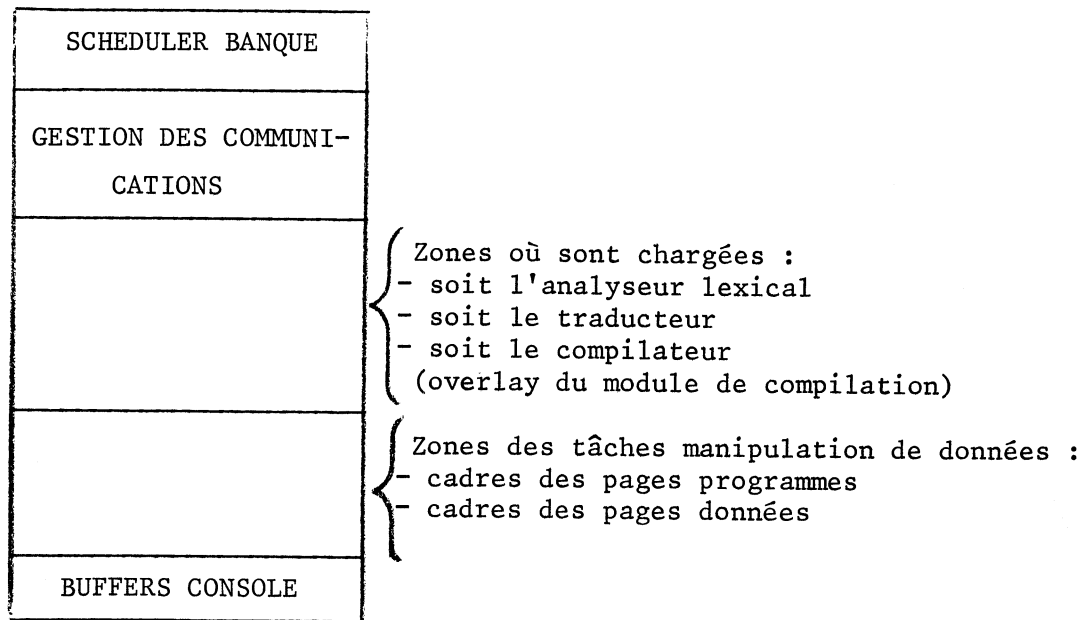
Pour des raisons de simplicité et d'efficacité ces programmes sont en "overlay", le module de compilation n'est pas partageable : les questions sont traitées globalement.

Cette technique n'apporte pas de restriction au fonctionnement du système, car la compilation d'une question est très rapide.

Gestion des communications. C'est le module qui gère toutes les entrées-sorties il est partageable entre le module compilation et les tâches de manipulation de données.

Tâches de manipulation de données. Elles sont créées dynamiquement par module de compilation. Elles peuvent être bloquées soit en attente d'informations (disque ou console), soit en attente sémantique. Ces tâches peuvent également être créées par des actions de mise à jour spontanée ([2] § 3).

En mémoire centrale, pour la partition dans laquelle se trouve la banque, la configuration est la suivante :



Rappelons que l'allocation dans la mémoire centrale est statique. C'est-à-dire que le nombre de tâches coexistantes est fixé à priori, et que les programmes ne sont pas segmentés.

B I B L I O G R A P H I E

[1] JR. ABRIAL, J. BAS, G. BEAUME, G. HENNERON, R. MORIN, G. VIGLIANO

Projet SOCRATE
(I) Spécifications générales

communication I.M.A.G. Août 1970

[2] G. VIGLIANO

Projet SOCRATE
(2.1.) Langage de requêtes

Thèse présentée à l'Université de Grenoble, déc. 1970

[3] G. BEAUME

Projet SOCRATE
(2.2.) Gestion des mémoires

Thèse présentée à l'Université de Grenoble, déc. 1970

[4] R. MORIN

Projet SOCRATE
(2.3.) Compilation et Interprétation

Thèse présentée à l'Université de Grenoble, déc. 1970

VU,

Grenoble, le

Le Président de la Thèse

Vu,

Grenoble, le

Le Doyen de la Faculté des Sciences

Vu, et permis d'imprimer

Le Recteur de l'Académie
de Grenoble